

# Python Cookbook

Release 3.0.0

çĖŁèĈi

Mar 18, 2018

## Contents

<b>1</b>	<b>Copyright</b>	<b>1</b>
<b>2</b>	<b>ĀĹĖĹĀ</b>	<b>1</b>
2.1	éążçŻöäÿzéąĭ . . . . .	1
2.2	ėŕŚèĀĖçŻĎėŕĭ . . . . .	1
2.3	äĭĬĖĀĖçŻĎėŕĭ . . . . .	2
2.4	èĤŽæĬĥăžęéĀĈăŖĹĹĕŕĀ . . . . .	2
2.5	èĤŽæĬĥăžęäÿĖĀĈăŖĹĹĕŕĀ . . . . .	3
2.6	āĬĬçŻĤçđ' žăĭŊăžççăĀ . . . . .	3
2.7	äĭĤçĤĬçđ' žăĭŊăžççăĀ . . . . .	3
2.8	èĀĤçşżæĹŚăžñ . . . . .	3
2.9	èĠŕ'ėŕć . . . . .	4
<b>3</b>	<b>çñăÿĀçñăĭĭĴæŤŕæĖőçşşæđĎăŖŇçőŬæşŤ</b>	<b>4</b>
3.1	1.1 èğçăŎŇăžŖăĹŬĕŧŊăĀĭĵçŻăđ'ŽăÿĹăŖŸéĠŖ . . . . .	5
3.2	1.2 èğçăŎŇăŖŕĕĤĖăžçăŕžĕşăĕŧŊăĀĭĵçŻăđ'ŽăÿĹăŖŸéĠŖ . . . . .	6
3.3	1.3 äĬĭçŤŽæĬĀăŖŎŊ äÿĹăĖĈçŧ'ă . . . . .	9
3.4	1.4 æşĕæĹĭæĬĀăđ'ğæĹŬæĬĀăŖŕçŻĎŊ äÿĹăĖĈçŧ'ă . . . . .	11
3.5	1.5 āōđçŎŕăÿĀăÿĹăĭĵăĖĹçžęŸşăĹŬ . . . . .	12
3.6	1.6 āĖĀăÿăÿĖçŻĎĕŧŎæŸăăŕĎăđ'ŽăÿĹăĀĭĵ . . . . .	15
3.7	1.7 āĖĀăÿăÿæŎŖăžŖ . . . . .	16
3.8	1.8 āĖĀăÿçŻĎĕŕŕçőŬ . . . . .	17
3.9	1.9 æşĕæĹĭăÿđ'āĖĀăÿçŻĎçŸăŖŇçŖĈż . . . . .	19
3.10	1.10 āĹăĕŽđ'ăžŖăĹŬçŸăŖŇăĖĈçŧ'ăăžŭăĤĬăŖăăžăžŖ . . . . .	20
3.11	1.11 āŖĭăŖăĹĠĠçĹĠĠ . . . . .	22
3.12	1.12 āžŖăĹŬăÿăăĠçŖŎŕăĕŋăæŤŕæĬĀăđ'ŽçŻĎăĖĈçŧ'ă . . . . .	23
3.13	1.13 éĀŽĕĤĠæşŖăÿĹăĖşĕŧŎăĖŎŖăžŖăÿĹăĖĀăÿăĹŬĕăĭ . . . . .	25
3.14	1.14 æŎŖăžŖăÿăæŤŕæŊăăŎŖçŧşăŕŤĕçĈçŻĎăŕžĕşă . . . . .	27
3.15	1.15 éĀŽĕĤĠæşŖăÿĹăĖŕăŕăĖĕŕăĭŤăĹĖççŻĎ . . . . .	28
3.16	1.16 ĕĤĠæžđ'ăžŖăĹŬăĖĈçŧ'ă . . . . .	30
3.17	1.17 äžŎăĖĀăÿăÿăæŖŖăŖŬăŖĕŖĖ . . . . .	32
3.18	1.18 æŸăăŕĎăŖçğŕăĹŕăžŖăĹŬăĖĈçŧ'ă . . . . .	33
3.19	1.19 ĕĭŋăĖăžŭăŖŇăŬĕĕŕăçŕŕăŤŕæĖ . . . . .	35
3.20	1.20 âŖĹăžŭăđ'ŽăÿĹăĖĀăÿăĹŬăŸăăŕĎ . . . . .	37

4.15	2.15	ā■Ūčņēāyšāy■āRŠāĒēāRŸéĢR	63
4.16	2.16	āzēāNĢāōŽāLŪāōjāēaijāijRāNŪā■Ūčņēāyš	65
4.17	2.17	āIJlā■Ūčņēāyšāy■ād'ĎčRĒhtmlāŠNxml	66
4.18	2.18	ā■Ūčņēāyšāzđ'čL'NēgčæđR	68
4.19	2.19	āōđčŌřāyĀāyļčōĀā■TčŽDēĀŠājŠāyNéŽ■āLEæđRāZl	70
4.20	2.20	ā■ŪēLCā■ŪčņēāyšāyLčŽDā■ŪčņēāyšāS■ājIJ	78
<b>5 çññāyL'čñāiijŽæTřā■ŪæŪēæIJšāŠNæŪúéŪt'</b> <b>80</b>			
5.1	3.1	æTřā■ŪčŽDāŽZēL■āžTāĒē	81
5.2	3.2	æL'gēāNčšļčāōčŽDætōčCžæTřēfRčōŪ	82
5.3	3.3	æTřā■ŪčŽDæaijāijRāNŪējŠāGž	84
5.4	3.4	āžNāĒnā■AāĒ■ēfZāLūæTt'æTř	86
5.5	3.5	ā■ŪēLCāLřād'gæTt'æTřčŽDæL'SāNĒāyŌēgčāNĒ	88
5.6	3.6	ād'■æTřčŽDæTřā■ēēfRčōŪ	89
5.7	3.7	æŪāçl'ūād'gāyŌNāN	91
5.8	3.8	āLEæTřēfRčōŪ	93
5.9	3.9	ād'gādNæTřčzDēfRčōŪ	94
5.10	3.10	čšl'ēYtāyŌčžfæĀgāžčæTřēfRčōŪ	97
5.11	3.11	ēŽRæIJžéĀL'æNl'	99
5.12	3.12	āšžæIJñčŽDæŪēæIJšāyŌæŪúéŪt'ējñæ■	101
5.13	3.13	ēōaçōŪæIJāāRŌāyĀāyļāŠlāžTčŽDæŪēæIJš	103
5.14	3.14	ēōaçōŪājŠāL'■æIJLāzjčŽDæŪēæIJšēNČāZt'	105
5.15	3.15	ā■Ūčņēāyšējñæ■cāyžæŪēæIJš	107
5.16	3.16	čzŠāRLæŪūāNžčŽDæŪēæIJšæS■ājIJ	108
<b>6 çññāŽŽčñāiijŽēf■āzčāZlāyŌčTšæL'RāZl</b> <b>110</b>			
6.1	4.1	æL'NāLéA■āŌĒēf■āzčāZl	110
6.2	4.2	āzččRĒēf■āzč	111
6.3	4.3	ājčTlčTšæL'RāZlāLZāžžæŪřčŽDēf■āzčælāaijR	112
6.4	4.4	āōđčŌřēf■āzčāZlā■Rēōō	114
6.5	4.5	āR■āRŠēf■āzč	116
6.6	4.6	āyēæIJL'ād'ŪēČlčLūæĀAçŽDčTšæL'RāZlāGjæTř	117
6.7	4.7	ēf■āzčāZlāLGčL'Ģ	119
6.8	4.8	ēușēfĢāRřēf■āzčāržēsāçŽDāijĀāgNéČlāLE	120
6.9	4.9	æŌŠāLŪčzDāRLčŽDēf■āzč	122
6.10	4.10	āžRāLŪāyLčt'cāijTāĀijēf■āzč	124
6.11	4.11	ārNæŪūēf■āzčād'ŽāyļāžRāLŪ	126
6.12	4.12	āy■ārNéŽĒārLāyLāĒČčt'āçŽDēf■āzč	128
6.13	4.13	āLZāžžæTřæ■ōād'ĎčRĒčōāēAš	129
6.14	4.14	āsTāijĀātNāēŪčŽDāžRāLŪ	132
6.15	4.15	ēāžāžRēf■āzčāRLāžūāRŌčŽDæŌŠāžRēf■āzčāržēsā	133
6.16	4.16	ēf■āzčāZlāžčæŽfwhileæŪāēŽRāļčŌř	134
<b>7 çññāžTčñāiijŽæŪĢāzūāyŌIO</b> <b>136</b>			
7.1	5.1	ērzaĒŽæŪĢæIJñæTřæ■ō	136
7.2	5.2	æL'Sā■rējŠāGžēĢšæŪĢāzūāy■	138
7.3	5.3	ājčTlāĒūāžŪāLEēŽTčņæLŪēāNčžLæ■ēçņæL'Sā■ř	139
7.4	5.4	ērzaĒŽā■ŪēLCæTřæ■ō	140
7.5	5.5	æŪĢāzūāy■ā■YāIJāL'■ēČjāĒŽāĒē	142

7.6	5.6	āŭņēäyšçŽDI/Oæš■ä;IJ	143
7.7	5.7	ērzaĒZāŌŅcijl' æŪGāzū	144
7.8	5.8	āŽžāōZād' ġārRēōrā;TçŽDæŪGāzūēf■āzč	145
7.9	5.9	ērzaRŪāzŅēfZāLūæTṛæ■ōāLrāRrāRYčijšāEšāŅzāy■	146
7.10	5.10	āĒĒā■ŸæŸāārDçŽDāzŅēfZāLūæŪGāzū	148
7.11	5.11	æŪGāzūēūrā;DāR■çŽDæš■ä;IJ	150
7.12	5.12	ætŋNērTṛæŪGāzūæŸrāRēā■ŸāIJl	151
7.13	5.13	ēŌūāRŪæŪGāzūād' zāy■çŽDæŪGāzūāLŪēāl	152
7.14	5.14	āf;çTēæŪGāzūāR■cijŪčāA	154
7.15	5.15	æL'šā■rāy■āRlæšTçŽDæŪGāzūāR■	155
7.16	5.16	ācđāLāæLŪæTzāRŸāūsæL'šāijĀæŪGāzūçŽDçijŪčāA	157
7.17	5.17	ārĒā■ŪēLĀæZāĒēæŪGæIJnæŪGāzū	160
7.18	5.18	ārĒæŪGāzūæRRēfçņēāŅĒēčĒāLræŪGāzūāržēšā	160
7.19	5.19	āL'Zāžzāyŕ' æŪūæŪGāzūāSŅæŪGāzūād' ž	162
7.20	5.20	äyŌäyšēāŅçnrāRççŽDæTṛæ■ōēĀŽāfā	165
7.21	5.21	āžRāLŪāŅŪPythonāržēšā	165
<b>8</b>		<b>çññāĒē■çñāijŽæTṛæ■ōcijŪčāAāSŅād'DçRE</b>	<b>169</b>
8.1	6.1	ērzaĒZCSVæTṛæ■ō	169
8.2	6.2	ērzaĒZJSONæTṛæ■ō	172
8.3	6.3	ēğçæđRçōĀā■TçŽDXMLæTṛæ■ō	177
8.4	6.4	ācđēGRāijRēğçæđRād' ġādNXMLæŪGāzū	180
8.5	6.5	ārĒā■ŪāĒyē;ñæ■cāyžXML	183
8.6	6.6	ēğçæđRāSŅāfōæTzXML	185
8.7	6.7	āLr'çTlāS;āR■çl'žēŪŕ'ēğçæđRXMLæŪGæaç	187
8.8	6.8	äyŌāĒšçšzādNæTṛæ■ōāžšçŽDāzđ' āžš	189
8.9	6.9	çijŪčāAāSŅēğççāAā■AāĒēfZāLūæTṛ	191
8.10	6.10	çijŪčāAēğççāABase64æTṛæ■ō	192
8.11	6.11	ērzaĒZāžNēfZāLūæTṛçžDæTṛæ■ō	193
8.12	6.12	ērzaRŪāŋNāēŪāSŅāRrāRŸēTfāžŅēfZāLūæTṛæ■ō	197
8.13	6.13	æTṛæ■ōçŽDçŕ'fāLāäyŌçžšēōæš■ä;IJ	207
<b>9</b>		<b>çññäyČçñāijŽāG;æTṛ</b>	<b>209</b>
9.1	7.1	ārRræŌēāRŪāzæđRæTṛēGRāRCæTṛçŽDāG;æTṛ	209
9.2	7.2	ārLræŌēāRŪāĒšēTōā■ŪāRCæTṛçŽDāG;æTṛ	210
9.3	7.3	çžZāG;æTṛāRCæTṛācđāLāāĒČāfāæAr	212
9.4	7.4	ēfTāžđād' ŽāyſāĀijçŽDāG;æTṛ	212
9.5	7.5	āōŽāzL'æIJL'ēžŸēōd' āRCæTṛçŽDāG;æTṛ	213
9.6	7.6	āōŽāzL'āŅfāR■æLŪāĒĒēĀTāG;æTṛ	216
9.7	7.7	āŅfāR■āG;æTṛæ■TēŌūāRŸēGRāĀij	217
9.8	7.8	āGRārSārRrērČçTlāržēšāçŽDāRCæTṛāyſæTṛ	219
9.9	7.9	ārĒā■TṛæŪzæšTçŽDçšzē;ñæ■cāyžāG;æTṛ	222
9.10	7.10	āyēēčlād' ŪçLūæĀAāfāæArçžDāZđērČāG;æTṛ	223
9.11	7.11	āĒĒēĀTāZđērČāG;æTṛ	226
9.12	7.12	ēōfēŪōēŪ■āŅĒäy■āōŽāzL'çŽDāRŸēGR	228
<b>10</b>		<b>çññāĒē■çñāijŽçšzāyŌāržēšā</b>	<b>231</b>
10.1	8.1	æTzāRŸāržēšāçŽDā■ŪņēäyšæŸçđ'ž	231
10.2	8.2	ēGſāōŽāzL'ā■ŪņēäyšçŽDæāijāijRāŅŪ	233

10.3	8.3	èol' áržèsæŕŕæŇAäyŁäyŇæŮĠçõaçŔĖāŕĕõõ	234
10.4	8.4	ālŽāzzād' gēĠŕáržèsæŭŮēŁĆĲIAāĖĖāŕæŮzæsŦ	236
10.5	8.5	āĲĲčšzäyŕAēcĖĖāšdæĀgāŕ	237
10.6	8.6	ālŽāzzāŕŕçõaçŔĖçŽĎāšdæĀg	239
10.7	8.7	ērČçŦĲĲčšzæŮzæsŦ	243
10.8	8.8	āŕŕçšzäyæL'ŕāsŦproperty	247
10.9	8.9	ālŽāzzæŮŕçŽĎçšzæŁŮāōđäĲŇāšdæĀg	251
10.108.10		äĲçŦĲāzŭēŕšēõaçõŮāšdæĀg	254
10.118.11		çõĀāŇŮæŦŕæŕçzšæđĎçŽĎĀĲāgŇāŇŮ	257
10.128.12		āōŽāzŁ æŖēāŕçæŁŮēĀĖæĲçsāšžçšz	261
10.138.13		āōđçŖŕæŦŕæŕæĲāđŇçŽĎçšzādŇçžæĲš	263
10.148.14		āōđçŖŕēĠāōŽāzŁ āōžāzĲ	268
10.158.15		āšdæĀgçŽĎāzççŔĖēōŕēŮō	271
10.168.16		āĲĲčšzäyāōŽāzŁ āđŽāyĲæđĎĖĀāzĲ	276
10.178.17		ālŽāzzäyērČçŦĲĲnitæŮzæsŦçŽĎāōđäĲŇ	277
10.188.18		ālŦçŦĲĲMixinsæL'ŕāsŦçšzāŁšēČĲ	278
10.198.19		āōđçŖŕçŁŮæĀĀŕžèsæŁŮēĀĖçŁŮæĀĀæĲž	281
10.208.20		ēĀžēŕĠāŭçŇçäyšērČçŦĲŕžèsæŮzæsŦ	284
10.218.21		āōđçŖŕēōŕēŮōēĀĖāĲāĲŕ	286
10.228.22		äyŕŦĲēĀšāšāōđçŖŕēōŕēŮōēĀĖāĲāĲŕ	289
10.238.23		āĲçŖŖāĲŦçŦĲæŦŕæŕçzšæđĎçŽĎĖĖāŕçõaçŔĖ	293
10.248.24		èol'çšzæŦŕæŇAæŕŦēĲçæšāĲĲ	296
10.258.25		ālŽāzzçĲĲšāŕāōđäĲŇ	298

## 11 çŇŇāzĲçŇāĲĲžĀĖČçĲĲŮĲŇ 302

11.1	9.1	āĲĲāĠçĲæŦŕäyŁæŭzāŁāāŇĖēçĖĀzĲ	303
11.2	9.2	ālŽāzzèçĖĖēŕāzĲæŮŭāĲçŦŽāĠçĲŦŕāĖČçĲæĲæĲŕ	304
11.3	9.3	ēğçēzđ' äyĀäyĲēçĖĖēŕāzĲ	306
11.4	9.4	āōŽāzŁ äyĀäyĲäyēāŕçæŦŕçŽĎēçĖĖēŕāzĲ	308
11.5	9.5	āŕŕēĠāōŽāzŁ āšdæĀgçŽĎēçĖĖēŕāzĲ	309
11.6	9.6	äyēāŕŕēĀŁ āŕçæŦŕçŽĎēçĖĖēŕāzĲ	312
11.7	9.7	ālŦçŦĲēçĖĖēŕāzĲāĲzāŁŮāĠçĲŦŕäyŁçŽĎçšzādŇæçĀæšē	314
11.8	9.8	āŕĖççĖĖēŕāzĲāōŽāzŁ äyžçšççŽĎäyĀēČĲāĲĖ	317
11.9	9.9	āŕĖççĖĖēŕāzĲāōŽāzŁ äyžçšz	319
11.109.10		äyžçšzāšŇēĲæĀĀæŮzæsŦŕēŕäĲžēçĖĖēŕāzĲ	322
11.119.11		ēçĖĖēŕāzĲäyžēçŇāŇĖēçĖĀĠçĲŕāçđāŁāāŕçæŦŕ	324
11.129.12		äĲçŦĲēçĖĖēŕāzĲæL'ŕāĖĖçšççŽĎāŁšēČĲ	327
11.139.13		äĲçŦĲāĖČçšzæŖġāŁŮāōđäĲŇçŽĎāŁŽāzž	328
11.149.14		æŦŦēŖçšçççŽĎāšdæĀgāōŽāzŁ ēāžāžŕ	331
11.159.15		āōŽāzŁ æĲĲ'āŕŕēĀŁ āŕçæŦŕçŽĎāĖČçšz	334
11.169.16		*argsāšŇ**kwargççŽĎāĲzāŁŮāŕçæŦŕçĲāŕ	336
11.179.17		āĲĲçšzäyĲāĲzāŁŮāĲçŦĲçĲĲŮĲŇēğĎçžē	339
11.189.18		āžēçĲĲŮĲŇæŮzāĲŕāōŽāzŁ çšz	342
11.199.19		āĲĲāōŽāzŁ çŽĎæŮŭāĀzāĲāgŇāŇŮçšçççŽĎāĲŕāšŦŮ	345
11.209.20		ālŦçŦĲāĠçĲæŦŕæšĲēğçāōđçŖŕæŮzæsŦŦēĠēĲ	347
11.219.21		ēĀŕāĖēĠāđŕçŽĎāšdæĀgæŮzæsŦ	353
11.229.22		āōŽāzŁ äyŁäyŇæŮĠçõaçŔĖāzĲçŽĎçõĀāŦæŮzæsŦ	355
11.239.23		āĲĲāšĀēČĲāŕŮēĠŕāššäyæL'gēāŇāzççāĲ	357

11.249.24	èğçæđŘäyŎáĽEæđŘPythonæžŘçāA	359
11.259.25	æŇEèğçPythonā■ŮèĽČçāA	363
<b>12</b>	<b>çññā■AçñāijŽælaaiŮäyŎāŇĚ</b>	<b>366</b>
12.1	10.1 æđĐāžžäyÄäyĽælaaiŮçŽĐāsĆçžgāŇĚ	366
12.2	10.2 æŎgāĽŮælaaiŮècñāĚĚĆĽārijāĚĚçŽĐāĚĚāōž	367
12.3	10.3 ä;ĲçŦĲçŽyāržeŮrā;ĐāŘ■ārijāĚĚāŇĚäy■ā■ŘælaaiŮ	368
12.4	10.4 ārĚælaaiŮāĽEāĽ'sæĽŘād'ŽäyĽæŮĠgāžŮ	369
12.5	10.5 āĽĲçŦĽāS;āŘ■çĲ'žéŮr'ārrijāĚĚçŽōā;ŦāĽEæŦççŽĐāžççāA	371
12.6	10.6 éĠ■æŮrāĽæ;ĲælaaiŮ	373
12.7	10.7 èĲŘēāŇçŽōā;ŦæĽŮāŎŇçijĲ'æŮĠgāžŮ	374
12.8	10.8 èřžāŘŮā;■äžŎāŇĚäy■çŽĐæŦřæ■ōæŮĠgāžŮ	375
12.9	10.9 ārĚæŮĠgāžŮād'žāĽāāĚĚāĽrsys.path	376
12.10	10.10 éĀŽèĲĠā■ŮçñçäyšāŘ■ārijāĚĚælaaiŮ	377
12.11	10.11 éĀŽèĲĠéŚĲ'ā■ŘèĲĲçĽĽāĽæ;ĲælaaiŮ	378
12.12	10.12 ārijāĚĚælaaiŮçŽĐāŘŇæŮŮāĲōæŦžælaaiŮ	393
12.13	10.13 āōĽ'èçĚçgAæĲĲçŽĐāŇĚ	396
12.14	10.14 āĽŽāžžæŮřçŽĐPythonçŎřāçČ	396
12.15	10.15 āĽEāŘŚāŇĚ	398
<b>13</b>	<b>çññā■AäyÄçñāijŽç;ŚçzĲJäyŎWebçijŮçĽĽ</b>	<b>399</b>
13.1	11.1 ä;ĲJäyžāōçæĽŮçñrāyŎHTTPæĲ■āĽāžd'äžŠ	399
13.2	11.2 āĽŽāžžTCPæĲ■āĽāžĽ	404
13.3	11.3 āĽŽāžžUDPæĲ■āĽāžĽ	407
13.4	11.4 éĀŽèĲĠCIDRāĲřāĽĲçŦšæĽŘāřžāžŦçŽĐĲPāĲřāĽĲéŽE	409
13.5	11.5 āĽŽāžžäyÄäyĽçōĀā■ŦçŽĐRESTæŎĚāŘç	411
13.6	11.6 éĀŽèĲĠGXML-RPCāōđçŎřçōĀā■ŦçŽĐèĲĲçĽĽNèřČçŦĽ	415
13.7	11.7 āĲJäy■āŘŇçŽĐPythonèğçéĠĽāžĽāžŇéŮr'äžd'äžŠ	418
13.8	11.8 āōđçŎřèĲĲçĽĽNæŮžæşŦèřČçŦĽ	419
13.9	11.9 çōĀā■ŦçŽĐāōçæĽŮçñrèōd'èřA	423
13.10	11.10 āĲĲç;ŚçzĲJæĲ■āĽäy■āĽāāĚĚSSL	425
13.11	11.11 èĲŽçĽĽNèŮr'äijäéĀSSocketæŮĠgāžŮāŘŘèĲřçñç	431
13.12	11.12 çŘEèğçāžŇāžŮēĲ'sāĽĲçŽĐIO	436
13.13	11.13 āŘŚéĀAäyŎæŎæŦŮād'gādŇæŦřçžĐ	441
<b>14</b>	<b>çññā■AäžŇçñāijŽāžŮāŘŚçijŮçĽĽ</b>	<b>443</b>
14.1	12.1 āŘrāĽĽäyŎāĲJæ■ççžçĽĽ	444
14.2	12.2 āĽd'æŮ■çžçĽĽNæŮřāŘēāŮşçžŘāŘrāĽĽ	446
14.3	12.3 çžçĽĽNèŮr'èĀŽāĲā	449
14.4	12.4 çžŽāĚşéŦōéĽĽāĽEāĽæĲA	454
14.5	12.5 éŮşæ■çæ■zéŦAçŽĐāĽæĲAæĲžāĽŮ	456
14.6	12.6 äĲĽā■ŮçžçĽĽNçŽĐçĽŮæĀAäĲāæĲr	460
14.7	12.7 āĽŽāžžäyÄäyĽçžçĽĽNæšā	461
14.8	12.8 çōĀā■ŦçŽĐāžŮēāŇçijŮçĽĽ	465
14.9	12.9 PythonçŽĐāĚĽāsĀéŦAéŮōéçŮ	469
14.10	12.10 āōŽāžĽĽäyÄäyĽActorāžžāĽā	471
14.11	12.11 āōđçŎřæŮĽæĲāŘŚāyČ/èōçéŮĚæĽāādŇ	475
14.12	12.12 ä;ĲçŦĲçŦšæĽŘāžĽāžçæŽççççĽĽ	478
14.13	12.13 ād'ŽäyĽçžçĽĽNèŮşāĽŮè;ōèřç	486



14.1412.14	āIĪJUnixçşzçzşşäyŁÉİcāRřāŁáōŁæŁd'ēfZçlŃ	489
<b>15</b>	<b>çññā■AäyŁçñāijŽēDŽæIJñcijÚçlŃäyŌçşzçzşçōaçŘE</b>	<b>492</b>
15.1	13.1 éĀŽēfGéG■āōŽāRŠ/çōaqAŞ/æŪGüzūāŌēāRŪē;ŞāĒē	493
15.2	13.2 çZŁæ■ççlŃāžRāzūçZŽāGžēŤŽēřřāŁæAr	494
15.3	13.3 èğçæđŘāŚ;äzd'ēāNēĀL'ēāž	494
15.4	13.4 èŁŘēāNæŪūāijzāGžāřEçāAē;ŞāĒēæRŘçd'ž	497
15.5	13.5 èŌūāRŪçZŁçñřçŽDād'gārR	498
15.6	13.6 æL'gēāNād'ŪēClāŚ;äzd'āžūēŌūāRŪāōÇçŽDē;ŞāGž	499
15.7	13.7 ād'■āLūæLŪēĀĒçgžāLīæŪGüzūāŠNçŽōā;T	501
15.8	13.8 āLŽāzžāŠNēgçāŌNā;ŞæaçæŪGüzū	503
15.9	13.9 éĀŽēfGæŪGüzūāR■æšæL;æŪGüzū	503
15.10	13.10 èřzāRŪēĒ■ç;ōæŪGüzū	505
15.11	13.11 çZŽçōĀā■TēDŽæIJñāçdāŁāæŪēāfŪāŁšèÇ;	508
15.12	13.12 çZŽāG;æTřāžŞāçdāŁāæŪēāfŪāŁšèÇ;	511
15.13	13.13 āōđçŌřāyĀäyŁēōaqŪūāZl	512
15.14	13.14 éŽŘāLūāEĒ■YāŠNCPUçŽDā;fçŤlēGR	514
15.15	13.15 āŘřāLīāyĀäyIWEBætŘēgŁāZl	515
<b>16</b>	<b>çññā■AāZŽçñāijŽætNērŤāĀAērČērŤāŠNāijČāyŷ</b>	<b>516</b>
16.1	14.1 æŤNērŤstdoutē;ŞāGž	516
16.2	14.2 āIĪJā■ŤāĒČætNērŤāy■çZŽāřžēşaqæL'ŞēaqēyA	518
16.3	14.3 āIĪJā■ŤāĒČætNērŤāy■ætNērŤāijČāyŷāČĒāEŤ	521
16.4	14.4 ārEætNērŤē;ŞāGžçŤlāŪēāfŪēōřā;ŤāLřæŪGüzūāy■	523
16.5	14.5 āŁ;çŤēæLŪæIJşæIJŽætNērŤād'set'ē	524
16.6	14.6 ād'ĐçŘEād'ŽāyŁāijČāyŷ	525
16.7	14.7 æ■TēŌūæL'ĀæIJL'āijČāyŷ	527
16.8	14.8 āLŽāzžēGĥāōŽāzL'āijČāyŷ	529
16.9	14.9 æ■TēŌūāijČāyŷāŘŌæLŽāGžāRēād'ŪçŽDāijČāyŷ	531
16.10	14.10 éG■æŪřæLŽāGžèçnā■TēŌūçŽDāijČāyŷ	533
16.11	14.11 ē;ŞāGžē■ēāŚLāŁæAr	534
16.12	14.12 èřČērŤāşžæIJñçŽDçlŃāžRāt'l'æžČēŤŽēřř	535
16.13	14.13 çZŽā;äçŽDçlŃāžRāAŽæĀgèÇ;ætNērŤ	538
16.14	14.14 āLāēĀşçlŃāžRēfŘēāN	541
<b>17</b>	<b>çññā■AāžŤçñāijŽČēr■ēlĀæL'l'āsŤ</b>	<b>545</b>
17.1	15.1 ā;fçŤlctypesēōfēŪōCāzççāA	547
17.2	15.2 çōĀā■ŤçŽDČæL'l'āsŤæĪāāIŪ	553
17.3	15.3 çijŪāEŽæL'l'āsŤāG;æTřæŞ■ā;IJæTřçžD	557
17.4	15.4 āIĪJČæL'l'āsŤæĪāāIŪāy■æŞ■ā;IJēŽŘā;çæNĠēŚL	559
17.5	15.5 āžŌæL'l'āsŤæĪāāIŪāy■āōŽāzL'āŠNārijāGžCçŽDAPI	562
17.6	15.6 āžŌČēr■ēlĀāy■ērČçŤlPythonāžççāA	566
17.7	15.7 āžŌČæL'l'āsŤāy■ēGŁæŤ;āĒlāsĀēŤA	571
17.8	15.8 ČāŠNPythonāy■çŽDçžfçlŃæūūçŤl	572
17.9	15.9 çŤlWSIGāNĒèçĒCāzççāA	573
17.10	15.10 çŤlCythonāNĒèçĒCāzççāA	578
17.11	15.11 çŤlCythonāEŽēnYāĀgèÇ;çŽDæTřçžDæŞ■ā;IJ	585
17.12	15.12 ārEāG;æTřæNĠēŚLē;ñæ■cāyžāRřērČçŤlāřžēşā	589
17.13	15.13 āijāēĀŠNULLçZŞāř;çŽDā■ŪçñēyşçZCāG;æTřāžŞ	590

17.14 15.14	äijäéĂŠUnicodeā■ŮčņēäyšçzŽCāĠæTřāžŠ . . . . .	594
17.15 15.15	Cā■Ůčņēäyšē;ñæ■cäyžPythonā■Ůčņēäyš . . . . .	599
17.16 15.16	äy■çāōāōŽçijŮçāAæäijäijRçŽĐCā■Ůčņēäyš . . . . .	600
17.17 15.17	äijäéĂŠæŮĠäzūāR■çzŽCæL'ġāsT . . . . .	603
17.18 15.18	äijäéĂŠāūsæL'SäijĀçŽĐæŮĠäzūçzŽCæL'ġāsT . . . . .	604
17.19 15.19	äzŮCēr■ēĀäy■ērzāRŮçszæŮĠäzūāržēsā . . . . .	605
17.20 15.20	ād'DçRĒCēr■ēĀäy■çŽĐāRrēf■äzçāržēsā . . . . .	608
17.21 15.21	ērŁæŮ■āŁEæōtēTŽēr . . . . .	609
<b>18</b>	<b>éŽĐā;TA</b>	<b>610</b>
18.1	āĬĬçžfētĐæžR . . . . .	610
18.2	Pythonā■ēāzāžēçs■ . . . . .	610
18.3	ēñŸçžgāžēçs■ . . . . .	611
<b>19</b>	<b>āĒšāžŮērSèĂĒ</b>	<b>611</b>
<b>20</b>	<b>Roadmap</b>	<b>611</b>

---

Contents:

# 1 Copyright

äžēāR■iijŽ āĂŁPython CookbookāĂŃ3rd Edition

äĬĬèĂĒiijŽ David Beazley, Brian K. Jones

ērSèĂĒiijŽ çĒŁèČĬ

çL'ŁæĬiijŽ çññ3çL'Ł

āĠžçL'Łçd' ĬiijŽ ŌāĂŽReilly Media, Inc.

āĠžçL'ŁæŮčæĬšīiijŽ 2013āžt'5æĬĬ08æŮč

Copyright Âġ 2013 David Beazley and Brian Jones. All rights reserved.

æŽt' ād' ŽāRŠāyČäŁæAfrēuāRČèĂČ

<http://oreilly.com/catalog/errata.csp?isbn=9781449340377>

## 2 aL'■èlĀ

### 2.1 éazçŽöäyžéaṭ

<https://github.com/yidao620c/python3-cookbook>

### 2.2 èrSèĀĖçŽĎèrl

äzçŦšèÑeçš■iijNæŁŚçŦl PythoniijA

èrSèĀĖäyĀçŽŦ' aiZæNĀä;ŁçŦl Python 3iijNāZāyžāōČāzčēalāžE Python çŽĎæIJĥælēāĀČēZ;çDūāRŚāRŌāĖijāōžæYŦāōČçŽĎçañaijd' iijNā;EæYŦēŁZāyĦāsĀēlčēŁšæŬl' aiijZæŦžāRŶçŽ èĀNāyŦ Python 3 çŽĎæIJĥælēĖIJĀēēAæŦRāyĦāzçŽĎāyōāŁl' āŠNæŦŦŦæNĀāĀČ çŽōāŁ'■āyČēlčāyŁçŽĎæŦŦçlNāzēçš■iijNç;ŚāyŁçŽĎæŁNāĖNāđ' gēČlāŁEāšžæIJñēČ;æYŦ 2.x çšžāŁŬçŽĎiijNāyŠēŬlāšžāžŌ 3.x çšžāŁŬçŽĎāzēçš■ārŚçŽĎāRŦæĀIJāĀČ

æIJĀēŁŚçIJNāŁŦŦāyĀæIJñāĀŁPython CookbookāĀN3rd Edi- tioniijNāōNāĖlāšžāžŌ Python 3iijNāĖZçŽĎāzšā;Łāy■ēŦžāĀČ äyžāžE Python 3 çŽĎæZōāRĤiijNæŁŚāzšāy■ēGlēGRāŁZiijNæČšāĀZçČžāžĀāžĤāžNæČĖāĀČāžŌæYŦāžŌiijNāršæIJL'āžEçŁ ēŁZāy■æYŦāyĀéāžē;žæĤçŽĎāūēā;IJiijNā■Ŧ æYŦāyĀāžūāĀijā;ŬāĀZçŽĎāūēā;IJiijZāy■āžĖæŬžā;ŁāžEāŁnā

èrSèĀĖäijZāiZæNĀāržēGĦāūsæŦRāyĀāRēçŽĎçŁžērŠēr' šēr' çiiijNāŁZæšČénYēr' léGRāĀČā;EāRŬēČ;āŁ āēČæđIJērSæŬGāy■æIJL'āžĀāžĤēŦZæijRçŽĎāIJŦæŬžērūāđ' gāōūēgAērEiijNāžšæñčēŁŌāđ' gāōūēŽRæŬūæN yidao620@gmail.com

### 2.3 ä;IJēĀĖçŽĎèrl

ēGĦāžŌ 2008 āžŦ'āžēælēiijNPython 3 æĦçŦ'žāGžāyŬāžūæĖčæĖčēŁZāNŬāĀČPython 3 çŽĎæŦAēāNāyĀçŽŦ' ēčñēōđ' äyžēIJĀēēAā;ŁēŦēāyĀæōŦæŬūēŬŦ' āĀČ āžNāōđāyŁiijNāŁŦŦæŁŚāĖZēŁZæIJñāzēçŽĎ 2013 āžŦ' iijNçžĦāđ' gēČlāŁEçŽĎ Python çlNāžRāŚYāž■çDūāIJçŦšāžgçŌŦāčČāy■ā;ŁçŦlçŽĎæYŦçŁŁæIJñ 2 çšžāŁŬiijN æIJĀāyžēēAæYŦāZāāyž Python 3 äy■ārŚāRŌāĖijāōžāĀČæŦŦæŬāçŬŠēŬōiijNāržāžŌāūēā;IJāIJléAŬçŦžāžçç ā;EæYŦæŦçIJiijæIJĥælēiijNā;āāršāijZāRŚçŌŦ Python 3 çžZā;āāyēælēāy■āyĀæāūçŽĎæČŁāŬIJāĀČ

æ■čāēČ Python 3 āžčēalæIJĥælēāyĀæāūiijNæŬŦçŽĎāĀŁPython Cook- bookāĀNçŁŁæIJñçŽyæŦē;ČāžNāŁ■çŽĎçŁŁæIJñæIJL'āžEāyĀāyĦāĖlæŬŦçŽĎæŦžāRŶāĀČ ēēŬāĖlŦiijNāžšæYŦæIJĀēČ■ēēAçŽĎiijNēŁZæĎŦāŚçlĀæIJñāzēæYŦāyĀæIJñēlđāyŦāŁ■æšŁçŽĎāRČēĀČāž Python 3.3 çŁŁæIJñāyNēlčçijŬāĖZāŠNæŦNērŦçŽĎiijN āžūæšāæIJL'ēĀČēZŠāžNāŁ■ēĀAçŁŁæIJñçŽĎāĖijā ā;EæYŦæŁŚāžñæIJĀçžŁçŽĎçŽōçŽĎæYŦāĖZāyĀæIJñāōNāĖlāšžāžŌçŌŦāžčāūēāĖūāŠNēr■ēlĀçŽĎāzēçš■āĀ æŁŚāžñāyNæIJZæIJñāzēēČ;āđ' šæNĠāŦijāžžāžñā;ŁçŦl Python 3 çijŬāĖZæŬŦçŽĎāzççāĀæŁŬēĀĖā■GçžgāžNāŁ■çŽĎēAŬçŦžāžççāĀāĀČ

æŦŦæŬāçŬŠēŬōiijNçijŬāĖZāyĀæIJñēŁZæāūçŽĎāzēççžçijŬē;Śāūēā;IJāyēælēāyĀāōžçŽĎæNŠæŁYāĀ Python çğYçš■çŽĎèrlŦiijNāijZāIJlēŦyāēČ ActiveStateāĀZs Python recipes æŁŬēĀĖ Stack Overflow çŽĎç;ŠçñZāyŁæRĤĤāŁŦæŦŦāžēā■ČēōāçŽĎæIJL'çŦlçŽĎçğYçš■iijNā;EæYŦāĖŬāy■çžĦāđ' gēČlāŁEē ēŁZāžZçğYçš■ēZđ' āžEæYŦāšžāžŌ Python 2 çijŬāĖZāžNāđ' ŬiijNārŦēČ;ēŁYæIJL'ā;Łāđ' ŽēgčāĖšæŬžæāŁāŁ iijŁæŦŦāēČ 2.3 āŠN 2.4 çŁŁæIJñiijL'āĀČ āŦēāđ' ŬiijNāōČāžñēŁYāijZçžRāyŦā;ŁçŦlāyĀāžZēŁGæŬūçŽĎæŁ



## 2.4 è£ŽæIŋäzééĂĆăŘĹěřĄ

æIJL'äyÄäZæZt'äŁäénŸçžgčŽDçgŸçs■ijNäçĆäđIjèÄŘäřČčÉYÈèrziiNäřEæIJL'äŁ'äžŌçŘEègč  
Python äžTäsČčŽDäüëä;IJäŌšçŘEäĀĆ äžŌäy■ä;äärEä■äŁräyÄäZæŸčŽDæŁÄäügäšNäŁÄäIJfijNäzüäž

## 2.5 èŁŻæłJňäzéäÿ■éĂĆăŘĹěřĄ

## 2.6 ĄǃıçžŁčd'żăȚNăzčçăĄ

`æIJñäzəǾGăázŎæl'ĂæIJL'æžŘăžčĉAǟİǾRăřăžəǟIJİ` [http://github.com/dabeaz/  
python-cookbook](http://github.com/dabeaz/python-cookbook) `äyŁéİcæl'ġăĹrăĂĆ` `ä;IJeĂEænćełŎăŘDă;■érzèĂĖăfđœ■ć`  
`bugiijNæṬzēfZăžčĉAǟŠNērĐèőžăĂĆ`

åIJl YouTube äÿLèġĆçIJNæĹŚäzniiJŽ<http://www.youtube.com/oreillymedia>

## 2.9 èGt'èrc

æŁŚäznèaũåŁÇæĎŝèrcæIJñäzèçŽĎæŁÄæIJfæääåöäžžåŠŸ Jake VanderplasiiĴRobert Kern åŠŇ Andrea Crotti éÍđäyÿæIJL'çŦłçŽĎèfĎèöžåŠŇäzžèöőiiĴN èŁŸæIJL' Python çĎ'çåŇžçŽĎäyöåŁ'åŠŇéijŝåŁsāĀÇæŁŚäznāŖŇæåũæĎŝèrcäyŁäyĀäyłçŁ'ŁæIJñçŽĎçijŮèçŚ Alex MartelliĴiiĴNAnna Ravenscroft åŠŇ David AscherāĀĆ ārçöæŁŽäyłçŁ'ŁæIJñæŸfæŮŕåŁŽäĴIJçŽĎiiĴNäĴEæŸfåŁ■äyĀäyłçŁ'ŁæIJñäyžæIJñäzèæŖŖäçŽäžEäyĀäyłæŇæIJĀāŖŖŌäžŝæŸfæIJĀéÇ■èçAçŽĎiiĴNæŁŚäznèçAæĎŝèrcæŁ'ĀæIJL'æŮ'æIJŝéçĎèĝŁçŁ'ŁæIJñçŽĎèfžèĀĒiiĴ

## 3 çññäyĀçñäiiĴŽæŦŕæ■óçžŝæĎĎåŠŇçóŮæŝŦ

Python æŖŖäçŽäžEäĎ'ĝéĠŖçŽĎåĒĒç;őæŦŕæ■óçžŝæĎĎiiĴNāŇĒæŇñåŁŮèåĴiiĴNéŽEāŖŁäzèāŖŁā■ŮāĒ äĴEæŸfiiĴNæŁŚäznäzŝäijŽçžŖäyÿçċŕåŁŕåŁŖŕeryæÇæŝèèrciiĴNæŖŝäžŖåŠŇèŁĠæzd'ç■Łç■Ł'èŁŽäžŽæŽóéA■åŽäæ■Ď'iiĴNèŁŽäyĀçñäçŽĎçŽóçŽĎŕŝæŸfèöŕèöžèŁŽäžŽæŖŦèçÇäyÿèĝAçŽĎéŮŖéçŸāŠŇçóŮæŝŦāĀĆ āŖæāĎ'ŮiiĴNæŁŚäznäzŝäijŽçžŽāĠžāIJléŽEāŖŁæĴāāĴŮ collections āĴŝäy■æŝ■āĴIJèŁŽäžŽæŦŕæ■óçžŝæĎĎçŽĎæŮžæŝŦāĀĆ

### 3.1 1.1 èĝçåŖŇäžŖåŁŮèŦNāĀijçžŽāĎ'ŽäyłåŖŸéĠŖ

#### éŮŖéçŸ

çŖŕåIJĴæIJL'äyĀäyłåŇĒåŖŇN äyłåĒĒçŦ'äçŽĎåĒĒçžĎæŁŮèĀĒæŸfäžŖŖåŁŮiiĴNæĀŖæåũāŖEāöČéĠŇéĴN äyłåŖŸéĠŖiiĴŝ

#### èĝçåŖŝæŮžæāŁ

äzžäĴçŽĎäžŖåŁŮiiĴLæŁŮèĀĒæŸfåŖŖŕèŁ■äžçāŖžèŝäiiĴL'āŖŖäzèéĀŽèŁĠäyĀäyłçóĀā■ŦçŽĎèŦNāĀijèŖ■āŖŖäyĀçŽĎåŁ■æŖŖŕŕŝæŸfåŖŖŸéĠŖçŽĎæŦŖŕéĠŖŕåŁÉéäžèũŝäžŖåŁŮāĒĒçŦ'äçŽĎæŦŖŕéĠŖæŸfäyĀæåũçŽĎäŖäzççāAçĎ'žäçŇiiĴŽ

```
>>> p = (4, 5)
>>> x, y = p
>>> x
4
>>> y
5
>>>
>>> data = [ 'ACME', 50, 91.1, (2012, 12, 21) ]
>>> name, shares, price, date = data
>>> name
'ACME'
>>> date
(2012, 12, 21)
>>> name, shares, price, (year, mon, day) = data
```

```
>>> name
'ACME'
>>> year
2012
>>> mon
12
>>> day
21
>>>
```

æCædIJæRÿéGRäyæTṙăŠŇăžRăĹŮăĚČt'ăçŽDäyæTṙäy■ăNzéĚ■iijŇaijŽăžğçTšăyĂăyĹaijCăyŷăĂĆ  
 äžççăAçd'žăĹŇiijŽ

```
>>> p = (4, 5)
>>> x, y, z = p
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: need more than 2 values to unpack
>>>
```

## èõlèõž

ăôdéŽĚäyĹiijŇeĹŽçğ■ğçăŎŇetŇăĀijăRřăžèçTĹăIJăžzä;TăRřeĹ■ăžčăržèšăyĹéĹciijŇeĂŇăy■ăžĚăžĚă  
 ăŇĚăŇăă■ŮçŇăyšriijŇeŮĜăžŮăržèšăiijŇeĹ■ăžčăŽĹăŠŇçTšæĹRăŽĹăĂĆ  
 äžççăAçd'žăĹŇiijŽ

```
>>> s = 'Hello'
>>> a, b, c, d, e = s
>>> a
'H'
>>> b
'e'
>>> e
'o'
>>>
```

æIJĹæŮŮăĂŽiijŇă;ăăRřèČ;ăRĹæČşèğçăŎŇăyĂéČĹăĹEiijŇăyćaijČăĚŮăžŮçŽDăĀijăĂĆăržăžŎèĹŽçğ■ă  
 Python äžŮăşăæIJĹæRŘăĹŽçĹ'žăôĹçŽDèr■ăşTăĂĆ ä;ĒæŸřă;ăăRřăžăä;ĹçTĹăžzæĎRăRÿéGRăR■ăŎžă■ăä;  
 äžççăAçd'žăĹŇiijŽ

```
>>> data = [ 'ACME', 50, 91.1, (2012, 12, 21) ]
>>> _, shares, price, _ = data
>>> shares
50
>>> price
91.1
>>>
```

ä;äâfĖëäzäflëfAä;äéĀLçTlçZĎĈcāzZā■ä;■āRŸéGRāR■āIJlāĖūāzŪāIJræŪzæšæcñä;£çTlāLřāĀC

## 3.2 1.2 èğçâŌNāRrè£■äzčāržèšæĭNāĀijçzŽad'ŽäyĭāRŸéGR

### éUőéčŸ

æÇædIJäyĀäyĭāRrè£■äzčāržèšæçŽĎāĖĈçt'äyĭāTřèüĖēfGāRŸéGRäyĭāTřæŪūĭijNāijŽæLZāGžäyĀäyĭ  
ValueError āĀC éCčāzLæĀŌæūāL■ēČ;äzŌēfŽäyĭāRrè£■äzčāržèšæäy■èğçâŌNāGž N  
äyĭāĖĈçt'āāGžæĭēĭijš

### èğçâEşæŪzæqĹ

Python çŽĎæŸšāRūēāĭē;āijRāRřāzēçTlāĭēèğçâEşēfŽäyĭāUőéčŸāĀCærTāēČĭijNā;āāIJlā■ēāzāyĀéŪ  
ä;äæČšçzšēōāyNāōūāz■ā;IJäyŽçŽĎāzšāĭGāĹRçzĭĭijNā;EæŸræŌšÉŽd'æŌL'çññäyĀäyĭāSñæIJāāRŌäyĀā  
ä;EāēÇædIJæIJL 24 äyĭāSçĭijšēfŽæŪūāĀZæŸšāRūēāĭē;āijRāřæt'çäyLçTlāIJžāzEĭijŽ

```
def drop_first_last(grades):  
    first, *middle, last = grades  
    return avg(middle)
```

āRēād'ŪäyĀçğ■æČĖāEĭijNāAĞēō;ä;äçŌřāIJlæIJLäyĀāzŽçTlāĹūçŽĎēōřā;TāĹŪēāĭĭijNærRāĭæēōřā;T  
ä;äāRřāzēāČRäyNēĭcēfZæūāĹEğçēfZāzŽēōřā;TĭijŽ

```
>>> record = ('Dave', 'dave@example.com', '773-555-1212', '847-555-  
→1212')  
>>> name, email, *phone_numbers = record  
>>> name  
'Dave'  
>>> email  
'dave@example.com'  
>>> phone_numbers  
['773-555-1212', '847-555-1212']  
>>>
```

āĀijā;ŪæşĭæĎRçŽĎæŸrāyĹēĭcēğçâŌNāGžçŽĎ phone\_numbers  
āRŸéGRærŸēfIJēČ;æŸrāĹŪēāĭçszādNĭijNāy■çōāèğçâŌNçŽĎçTřēřlāRūçāAæTřēGRæŸrād'ŽārSĭijLāNĖæN  
0 äyĭijL'āĀC æĹ'ĀāzēĭijNāzžā;Tā;£çTlāĹř phone\_numbers  
āRŸéGRçŽĎāzççāAāršäy■ēIJĀēēAāAžād'Žä;ŽçŽĎçszādNæçĀæšēāŌzçāōēōd'āōČæŸrāRēæŸrāĹŪēāĭçszād

æŸšāRūēāĭē;āijRāzšēČ;çTlāIJlāĹŪēāĭçŽĎāijĀāğNēČĭāĹēāĀCærTāēČĭijNā;äæIJL'äyĀäyĭāĖñāRyāL■  
8 äyĭāIJLēTāĀTōæTřæ■ōçŽĎāzRāĹŪūĭijN ā;EæŸrā;äæČšçIJNāyNāēIJĀēfSäyĀäyĭāIJLæTřæ■ōāSñāL■ēĭc  
7 äyĭāIJLçŽĎāzšāĭGāĀijçŽĎāržærTāĀČä;āāRřāzēēfZæūāĀŽĭijŽ

```
*trailing_qtrs, current_qtr = sales_record  
trailing_avg = sum(trailing_qtrs) / len(trailing_qtrs)  
return avg_comparison(trailing_avg, current_qtr)
```

äyNēĭcæŸrāIJĭ Python èğçéGĹāŽĭäy■æĹ'ğēāNçŽĎçzšædIJĭijŽ



```
>>> *trailing, current = [10, 8, 7, 1, 9, 5, 10, 3]
>>> trailing
[10, 8, 7, 1, 9, 5, 10]
>>> current
3
```

## èóìèőž

æL'ſ'āsTçŽDēf■āzčēgčāŌŇēr■æſTæYřāyŠēŮlāyžēgčāŌŇāy■čāōāōŽāyſæTřæLŮāzzæDŘāyſæTřāĚČčt'ā  
 éĚŽāyſſijNēfZāžZāRřēf■āzčāržēsāçŽDāĚČčt'āçzŠædDæIJL'čāōāōŽçŽDēgDāLŽſijLæfTæČčññ  
 1 āyſāĚČčt'āāRŌēlčēČ;æYřçTřērſāRūçāAſijLſijN æYšāRūēāſē;āijRēōſ'āijĀāRŠāžžāŠYāRřāžēā;LāōžæYŠç  
 èĀŇāy■æYřēĀŽēfGāyĀāžZæfTē;Čād'■ælČçŽDæL'NæōſāŌžēŌūāRŮēfZāžZāĚšēAſTçŽDāĚČčt'āāĀijāĀČ  
 āĀijā;ŮāſſæDŘçŽDæYřſijNæYšāRūēāſē;āijRāIJſēf■āzčāĚČčt'āāyžāRřāRŸēTfāĚČčzDçŽDāžRāLŮā  
 æfTæČčſijNāyNēlčæYřāyĀāyſāyçæIJL'æāGç;çŽDāĚČčzDāžRāLŮſijŽ

```
records = [
    ('foo', 1, 2),
    ('bar', 'hello'),
    ('foo', 3, 4),
]

def do_foo(x, y):
    print('foo', x, y)

def do_bar(s):
    print('bar', s)

for tag, *args in records:
    if tag == 'foo':
        do_foo(*args)
    elif tag == 'bar':
        do_bar(*args)
```

æYšāRūēgčāŌŇēr■æſTāIJſā■ŮçņēāyšæŠ■ā;IJçŽDæŮūāĀŽāžšāijŽā;LæIJL'çTřſijNæfTæČā■Ůçņēāyšç  
 āzččāAçd'žā;NſijŽ

```
>>> line = 'nobody::-2:-2:Unprivileged User:/var/empty:/usr/bin/
↳false'
>>> uname, *fields, homedir, sh = line.split(':')
>>> uname
'nobody'
>>> homedir
'/var/empty'
>>> sh
'/usr/bin/false'
>>>
```

æIJL'æUûāĀŽiijNā;āæČšèġcāŌNäyĀāzŽāĒČčt'āāRŌäyćaijČāŏČāzñiijNā;āāy■èČ;ćŏĀā■Tāřsā;ĤčTĪ  
\* iijN ā;ĤæYřā;āāRřāzēā;ĤčTĪāyĀāyĭæŽŏéĀŽčŽDāžšāijČāR■čġiijNāēřTāēČ \_ æLŪēĀĒ  
ign iijLignoreiijLāĀČ

āžččāAčd'žā;NīijŽ

```
>>> record = ('ACME', 50, 123.45, (12, 18, 2012))
>>> name, *_ , (*_ , year) = record
>>> name
'ACME'
>>> year
2012
>>>
```

āIJlā;Ĺād'ŽāĠ;æTřāijRēř■ēĹĀāy■iijNæYšāRūèġcāŌNēr■æšTēušāLŪēāĹād'ĎčŘĖæIJL'èŏyād'ŽčŽyāijija  
ā;āāRřāzēā;ĹāŏzæYščŽDāřĖāŏČāĹĖāL'sāĹRāL■āRŌäyđ'ēČĹāĹĖiijŽ

```
>>> items = [1, 10, 7, 4, 5, 9]
>>> head, *tail = items
>>> head
1
>>> tail
[10, 7, 4, 5, 9]
>>>
```

āēČāēdIJā;āād'šèAĤæYŌčŽDērIiijNēĤYēČ;čTĪēĤŽčġ■āĹĖāL'sēr■æšTāŌzāūġāēŽčŽDāŏđčŌřéĀšā;ŠčŏŪ

```
>>> def sum(items):
...     head, *tail = items
...     return head + sum(tail) if tail else head
...
>>> sum(items)
36
>>>
```

čDūāRŌiijNčT'sāzŌēr■ēĹĀāsČēĹččŽDēŽRāĹŪiijNēĀšā;Šāzūāy■æYř Python  
æŠĖēTĤčŽDāĀČ āŽāæ■điijNæIJāāRŌēČčāyĭēĀšā;ŠāijTčđ'žāzĒāzĒæYřāyĭāē;āēĠčŽDæŌččt'ćč;ćāžĖiijNā

### 3.3 1.3 āĤiçTŽæIJĀāRŌ N āyĭāĒČčt'ā

éUŏéčY

āIJĹē■āžčæš■ā;IJæLŪēĀĒāĒūāzŪæš■ā;IJčŽDæUûāĀŽiijNæĀŌæūāRĹāĤĤiçTŽæIJĀāRŌæIJL'ēŽRāĠā

èġcāĖšæŪzæāĹ

āĤiçTŽæIJL'ēŽRāŌĖāRšēŏřā;Tæ■čæYř collections.deque  
ād'ġæY;èžnæL'NčŽDæUûāĀŽāĀČæřTāēČiijNāyNēĹččŽDāžččāAāIJĹād'ŽēāNāyĹēĹcāĀŽčŏĀā■TčŽDæŪĠæ  
āzūēĤTāZđāNzéĒ■æL'ĀāIJĹēāNčŽDæIJĀāRŌNēāNīijŽ

```

from collections import deque

def search(lines, pattern, history=5):
    previous_lines = deque(maxlen=history)
    for line in lines:
        if pattern in line:
            yield line, previous_lines
        previous_lines.append(line)

# Example use on a file
if __name__ == '__main__':
    with open(r'../../cookbook/somefile.txt') as f:
        for line, prevlines in search(f, 'python', 5):
            for pline in prevlines:
                print(pline, end='')
            print(line, end='')
            print('-' * 20)

```

## èõìèõž

æŁŚăžňăĬĬăĚŽæšëèrcăĖĈĈťăçŽĎăžččăĀæŮüijŇěĂŽăyyăijŽă;ĤçŤlăŇěăŔŋ yield  
 èăĹè;ăijŔçŽĎĈŤšæĹŔăŽĹăĜ;æŤŕijŇăžšăŕšæŸŕæĹŚăžňăyĹéĹčđ'žăĹŇăžččăĀăy■çŽĎéĈĈăăŭăĂĈ  
 èĤŽăăŭăŔŕăžěăŕĖæŔĬĬť'cèĤĜçĹŇăžččăĀăšŇă;ĤçŤlăŔĬĬť'ćçžšæđĬĬăžččăĀèğçèĂăăĂĈăĈăđĬĬă;ăèĤŸăy■  
 4.3 èĹĈăĂĈ

ă;ĤçŤĬ deque(maxlen=N) æđĎéĂăăĜ;æŤŕăijŽæŮŕăžžăyĂăyĹăŽžăôŽăđ'ğăŕŔçŽĎéŸšăĹŮăĂĈă;šæŮ  
 æĬĬăĖĂĀçŽĎăĖĈĈťăăijŽèĜĹăĹéćŋçğžéŽđ'æŎĹ'ăĂĈ

ăžččăĀçđ'žăĹŇüijŽ

```

>>> q = deque(maxlen=3)
>>> q.append(1)
>>> q.append(2)
>>> q.append(3)
>>> q
deque([1, 2, 3], maxlen=3)
>>> q.append(4)
>>> q
deque([2, 3, 4], maxlen=3)
>>> q.append(5)
>>> q
deque([3, 4, 5], maxlen=3)

```

ăŕ;çôăq;ăăžšăŔŕăžěæĹŇăĹăĬĬăyĂăyĹăĹŮèăĹăyĹăôđçŎŕèĤŽăyĂçŽĎăš■ă;ĬüijĹăŕŤăèĈăćđăĹăăĂĀăĹ  
 æŽťăyĂèĹŋçŽĎüijŇ deque çšžăŔŕăžěèćŋçŤĹăĬĬăžžă;Ťă;ăăŔĹéĬĬăĖĂăyĂăyĹčôĂă■ŤéŸšăĹŮăŤŕæ■ôç  
 âĖĈăđĬĬă;ăăy■èôç;ôæĬĬăđ'ğéŸšăĹŮăđ'ğăŕŔüijŇéĈăžĹăŕšăijŽă;ŮăĹŕăyĂăyĹæŮăéŽŔăđ'ğăŕŔéŸšăĹŮüijŇ  
 äžččăĀçđ'žăĹŇüijŽ

```

>>> q = deque()
>>> q.append(1)
>>> q.append(2)
>>> q.append(3)
>>> q
deque([1, 2, 3])
>>> q.appendleft(4)
>>> q
deque([4, 1, 2, 3])
>>> q.pop()
3
>>> q
deque([4, 1, 2])
>>> q.popleft()
4

```

1.  $O(1)$  for append and pop operations.  
 2.  $O(N)$  for popleft and appendleft operations.

## 3.4 1.4 æšæL'æJÄäð'gæL'UæJÄärRçŽD N äyläEČt'ä

### éUóécY

1.  $O(N)$  for finding the largest and smallest elements.

### èğçâEşæÚzæqL

1.  $O(N)$  for finding the largest and smallest elements.

```

import heapq
nums = [1, 8, 2, 23, 7, -4, 18, 23, 42, 37, 2]
print(heapq.nlargest(3, nums)) # Prints [42, 37, 23]
print(heapq.nsmallest(3, nums)) # Prints [-4, 1, 2]

```

1.  $O(N)$  for finding the largest and smallest elements.

```

portfolio = [
    {'name': 'IBM', 'shares': 100, 'price': 91.1},
    {'name': 'AAPL', 'shares': 50, 'price': 543.22},
    {'name': 'FB', 'shares': 200, 'price': 21.09},
    {'name': 'HPQ', 'shares': 35, 'price': 31.75},
    {'name': 'YHOO', 'shares': 45, 'price': 16.35},
    {'name': 'ACME', 'shares': 75, 'price': 115.65}
]

```

```
cheap = heapq.nsmallest(3, portfolio, key=lambda s: s['price'])
expensive = heapq.nlargest(3, portfolio, key=lambda s: s['price'])
```

erSèĀĒæşlīijŽäyLéİcäzççāAāIJlārzaerRäylāĒĈçt' æēfZèaŊārzaerTçŽDæŪūāĀZīijŊāijŽāzē  
price çŽDāAijēfZèaŊærTēçCāĀĆ

## èóIèőž

æĈCædIJā;āæĈşāIJlāyĀäyIéZEāRLāy■æşæeL'çæIJĀārRæLŪæIJĀād'ğçŽD N  
äyIāĒĈçt' āīijŊāzūāyT N ārRāzŌéZEāRLāĒĈçt' āæTŕeGRīijŊéCçāzLèfZāzŽāĠ;æTŕæRRāçZāzEāçLāēççŽDæ  
āZāyzaIJlāzTāsCāōdçŌrēGŊéİcīijŊēēŪāĒLāijŽāĒLārEçZEāRLæTŕæ■ōēfZèaŊāāEæŌSāzRāRŌāTçāĒēäy

```
>>> nums = [1, 8, 2, 23, 7, -4, 18, 23, 42, 37, 2]
>>> import heapq
>>> heap = list(nums)
>>> heapq.heapify(heap)
>>> heap
[-4, 2, 1, 23, 7, 2, 18, 23, 42, 37, 8]
>>>
```

āāEæTŕæ■ōçzŞædDæIJĀéG■ēæAçŽDçL'zāçAæYŕ heap[0]  
ærŷèfIJæYŕæIJĀārRçŽDāĒĈçt' āāĀCāzūāyTāLŕ'äçŽDāĒĈçt' āāRfrazēāçLāōzæYŞçŽDēĀZēfĠerCçTī  
heapq.heappop() æŪzæşTāçŪāLŕīijŊ èrēæŪzæşTāijŽāĒLārEçñnāyĀäyIāĒĈçt' āāijzāĠzæIēīijŊçDūāRŌ  
O(log N)īijŊN æYŕāāEāād'ğārRīijLāĀĆ ærTāeCīijŊāæĈCædIJæĈşēæAæşæeL'çæIJĀārRçŽD 3  
äyIāĒĈçt' āīijŊā;āāRfrazēēfZæāūāĀZīijŽ

```
>>> heapq.heappop(heap)
-4
>>> heapq.heappop(heap)
1
>>> heapq.heappop(heap)
2
```

āçŞēæAæşæeL'ççŽDāĒĈçt' āäyIæTŕçZyārzaerTēçCārRçŽDæŪūāĀZīijŊāĠ;æTŕ  
nlargest() āŞŊ nsmallest() æYŕāçLāRLéĀĆçŽDāĀĆ  
æĈCædIJā;āāzĒāzĒæĈşæşæeL'çāTŕāyĀçŽDæIJĀārRæLŪæIJĀād'ğīijLN=1īijLççŽDāĒĈçt' æçŽDèfīijŊéCçāzL  
min() āŞŊ max() āĠ;æTŕāijŽæZt'āfñāzŽāĀĆ çşzāijijççŽDīijŊāæĈCædIJ N  
ççŽDād'ğārRāŞŊéZEāRLād'ğārRæŌēēfSççŽDæŪūāĀZīijŊéĀZāyŷāĒLæŌSāzRēfZāyIéZEāRLçDūāRŌāE■ā;  
īijL sorted(items)[:N] æLŪēĀĒæYŕ sorted(items)[-N:]  
īijLāĀĆ éIJĀēæAāIJlæ■ççāōāIJzāRLāççTīāĠ;æTŕ nlargest() āŞŊ  
nsmallest() æL■ēççāRŠæŊēāōCāzñççŽDāijYāLæ īijLāæĈCædIJ N  
āfñāŌēēfSéZEāRLād'ğārRāzEīijŊéCçāzLāççTīæŌSāzRæŞ■ā;IJāijŽæZt'æççāzZīijLāĀĆ

ārçōāā;āæşæeIJLāfĒēæAäyĀāōZāççTīlèfZéGŊççŽDæŪzæşTīijŊā;EæYŕāāEæTŕæ■ōçzŞædDççŽDāōdçç  
āşzæIJnāyLāRlèæAæYŕæTŕæ■ōçzŞædDāŞŊçŌŪæşTāzēçş■éGŊéİcéççāijŽæIJLæRRāRLāLŕāĀĆ  
heapq ælāāŪççŽDāōYæŪzæŪGæaçēGŊéİcāzşērēççEççŽDāzŊçz■āzEāāEæTŕæ■ōçzŞædDāzTāsCççŽDāōdçç



## 3.5 1.5 áódçŎřăÿĂăÿłăijŸăĚĹčžġéŸșăĹŮ

### éŮóécŸ

æĀŎæăăăódçŎřăÿĂăÿłæŃĹăijŸăĚĹčžġæŎșăžŔçŽĐéŸșăĹŮiijș  
ăžŮăÿŤăĲĲéŤŽăÿłéŸșăĹŮăÿłéĹăŕŔăŋă pop æșăăĲĲæĂžăŸŕèŤăŽđăijŸăĚĹčžġæĲĂénŸçŽĐéĈăÿłăĚĈç

### èġčăĒșăŮžăăĹ

ăÿŃéĹčçŽĐçșăĹĹčŤĲ heapq æĲăăĲŮăódçŎřăžĒăÿĂăÿłçőĂăăŤçŽĐăijŸăĚĹčžġéŸșăĹŮiijŽ

```
import heapq

class PriorityQueue:
    def __init__(self):
        self._queue = []
        self._index = 0

    def push(self, item, priority):
        heapq.heappush(self._queue, (-priority, self._index, item))
        self._index += 1

    def pop(self):
        return heapq.heappop(self._queue)[-1]
```

ăÿŃéĹăŸŕăőĈçŽĐăĲéŤĲăŮžăijŔiijŽ

```
>>> class Item:
...     def __init__(self, name):
...         self.name = name
...     def __repr__(self):
...         return 'Item({!r})'.format(self.name)
...
>>> q = PriorityQueue()
>>> q.push(Item('foo'), 1)
>>> q.push(Item('bar'), 5)
>>> q.push(Item('spam'), 4)
>>> q.push(Item('grok'), 1)
>>> q.pop()
Item('bar')
>>> q.pop()
Item('spam')
>>> q.pop()
Item('foo')
>>> q.pop()
Item('grok')
>>>
```

ăžŤçžĒèġĈăŕșăŕŕăžăăŔŖçŎřiijŃçŋăăÿĂăÿł pop() æșăăĲĲéŤăŽđăijŸăĚĹčžġæĲĂénŸçŽĐăĚĈçŤăăĂ

âRëåd'ŪæşlæĎRăĹRăęCæđIJăyd'ăylæIJL'çĹĂçŻyăŔŇăijYăĔĹçžğçŽĎăĔÇçť'ăiijĹ foo âŞŇ  
grok ĩijL'ĩijŇpop æŞ■ă;IJæŇL'çĔğăŏCăznëćnæŔŚăĔëăĹŕëYşăĹŪçŽĎéąžăžŔëĤăŽđçŽĎăĂĆ

## èóĹëőž

èĤŽăyĂăŕŔèĹCăĹŚăznăyžèęAăĔşæşĹ heapq æĹăăĹŪçŽĎă;ĤçŦĹăĂĆ  
ăĢ;æŦŕ heapq.heappush() âŞŇ heapq.heappop() âĹĒăĹăĹăĹĹéYşăĹŪ  
\_queue äyĹæŔŚăĔëăŞŇăĹăĔđ'çŇŇăyĂăylăĔÇçť'ăiijŇ âžüăyŦéYşăĹŪ  
\_queue âĤĹërAçŇŇăyĂăylăĔÇçť'ăæŇëæIJL'æIJăénYăijYăĔĹçžğĭijĹ  
1.4 èĹCăüşçžŔëŏĹëőžèĤĢëĤŽăyléŪŏéćYĭijL'ăĂĆ heappop()  
ăĢ;æŦŕæĂzæYŕëĤăŽđăĂĹæIJăŕŔçŽĎăĂĹçŽĎăĔÇçť'ăiijŇëĤŽăŕşæYŕăĤĹërAęYşăĹŪpopæŞ■ă;IJëĤăŽđæŕ  
âRëåd'ŪĭijŇçŦşăžŐ push âŞŇ pop æŞ■ă;IJæŪéŪŕăđ'■ăĹCăžęăyž  
O(log N)ĭijŇăĔŭăy■ N æYŕăăĔçŽĎăđ'ğăŕŔĭijŇăŽăæ■đ'ăŕşçŏŪæYŕ N  
ăĴĹăđ'ğçŽĎăŪăĂŽăŏCăznëĤŔëąŇëĂşăžęăžşă;ĹæŪğăĹĹăĤŇăĂĆ

ăĹĹăyĹĔĹcăžççăAăy■ĭijŇéYşăĹŪăŇĒăŕăžĔăyĂăyl (-priority, index,  
item) çŽĎăĔÇçžĎăĂĆ äijYăĔĹçžğăyžet'şæŦŕçŽĎçŐçŽĎăYŕă;ĤăĴŪăĔÇçť'ăæŇL'çĔğăijYăĔĹçžğăžŐénY  
èĤŽăylèŭşæŽŏéĂŽçŽĎăŇL'ăijYăĔĹçžğăžŐă;ŐăĹŕénYăŐŚăžŔçŽĎăăĔĔşăŔæAŕăŭğçŽyăŔ■ăĂĆ

index âŔYéĢŔçŽĎă;IJçŦĹăYŕăĤĹërAăŔŇç■Ĺ'ăijYăĔĹçžğăĔÇçť'ăçŽĎă■ççăŏæŐŚăžŔăĂĆ  
éĂŽëĤĢăĤĹă■ăYăĂăylăy■ăŪ■ăćđăĹăçŽĎ index äyŇăăĢăŔYéĢŔĭijŇăŔŕăžççăŏăĹăĔÇçť'ăæŇL'çĔğăŏCăž  
èĂŇăyŦĭijŇ index âŔYéĢŔăžşăĹĹçŽyăŔŇăijYăĔĹçžğăĔÇçť'ăæŕŦëĴçŽĎăŪăăĂŽëŦăĹŕéĢ■ëęAă;IJçŦĹă

ăyžăžĔęYŔæYŐëĤŽăžŦĭijŇăĔĹăĂĢăŏŽ Item âŏđă;ŇăYŕăy■ăŦŕæŇĂæŐŚăžŔçŽĎĭijŽ

```
>>> a = Item('foo')
>>> b = Item('bar')
>>> a < b
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: unorderable types: Item() < Item()
>>>
```

ăęĆæđIJă;ăă;ĤçŦĹăĔÇçžĎ (priority, item) ĩijŇăŔĹëęAăyd'ăylăĔÇçť'ăçŽĎăijYăĔĹçžğăy■ăŔŇăŕ  
ă;ĔæYŕăęĆæđIJăyd'ăylăĔÇçť'ăăijYăĔĹçžğăyĂæăŭçŽĎërĭijŇéĆćăžĹăŕŦëĴçŞ■ă;IJăŕşăijŽëŭşăžŇăĹ■ăyĂ

```
>>> a = (1, Item('foo'))
>>> b = (5, Item('bar'))
>>> a < b
True
>>> c = (1, Item('grok'))
>>> a < c
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: unorderable types: Item() < Item()
>>>
```

éĂŽëĤĢăijŦăĔëăŔëåd'ŪçŽĎ index âŔYéĢŔçŽĎăĹŔăyL'ăĔÇçžĎ  
(priority, index, item) ĩijŇăŕşëĴăĴĹăë;çŽĎéĂĤăĔăyĹĔĹççŽĎéŦŽërĭijŇ  
ăŽăăyžăy■ăŔŕëĴ;æIJL'ăyd'ăylăĔÇçť'ăæIJL'çŽyăŔŇçŽĎ index âĂĭăĂĆPython

```
>>> a = (1, 0, Item('foo'))
>>> b = (5, 1, Item('bar'))
>>> c = (1, 2, Item('grok'))
>>> a < b
True
>>> a < c
True
>>>
```

heapq æl̥aaiUçŽĐǎoŸæŮzæŮĜæɬæIJL'æŽt'ereçzEçŽĐä;N̥a■ŘçlNăžRăzěaRĹăřzăžŌăăEçŘĚeőžăRĹ

éŮóécŸ

èġčǎẸșæŮźæǻŁ

```
d = {
    'a' : [1, 2, 3],
    'b' : [4, 5]
}

e = {
    'a' : {1, 2, 3},
    'b' : {4, 5}
}
```

ä;ääRräzëä;ŁæŮzä;ŁçŽDä;ŁçŦĬ	collections	æłaaİŮäy■çŽD
defaultdict	ælēæđDéĂăēfŽæăüçŽD■ŮăĚyăĂĈ	defaultdict
çŽDäyĂăyŁçŁ;Ză;AæYřaŎĈaijŽëĠlăLlăLiăĠNăŦŮæřRăyĭ		key
ăĹŽăijĂăgNărzăŹŦçŽDăAijjijNăĹĂăzëä;ääRlëİJĂëëAăĤşæşlăüzăŁăăĚĈçŦ;ăæS■ă;IJăŽEăĂĈæřŦăëĈijŽ		

```
from collections import defaultdict

d = defaultdict(list)
```

```
d['a'].append(1)
d['a'].append(2)
d['b'].append(4)

d = defaultdict(set)
d['a'].add(1)
d['a'].add(2)
d['b'].add(4)
```

éIJĀēęAæşlæĐRçŽĐæŸriijŃ defaultdict äijŽēĠāLlāyžārEēęAēōēUōçŽĐéŤōriijLārşçōUçŽōāL'■  
 āęĆæđIJä;āāzūāy■éIJĀēęAēēZæāūçŽĐçL'zæĀğriijŃä;āāRřæžēāIJlāyĀāyłæŽōéĀŽçŽĐā■UāĚyāyŁä;ęçŤl  
 setdefault() æŮzæşŤælēāzçæŽēāĀĆærŤæęĆriijŽ

```
d = {} # A regular dictionary
d.setdefault('a', []).append(1)
d.setdefault('a', []).append(2)
d.setdefault('b', []).append(4)
```

ä;EæŸrā;Łād'ŽçlŃāzRāŚŸēğL'ā;Ů setdefault() çŤlēūælēæIJL'çCzāLŃæL'■āĀĆāZāyžærŤæñæēŃ  
 [] iijL'āĀĆ

## ēōlēōž

āyĀēLŃælēēōŃriijŃāLZāzžāyĀāyłād'ŽāĀijæŸāārDā■ŮāĚyæŸrā;ŁçōĀā■ŤçŽĐāĀĆä;EæŸriijŃāęĆæđIJä  
 ä;āāRřēČ;äijŽāČRāyŃēlēēZæāūælēāōđçŌriijŽ

```
d = {}
for key, value in pairs:
    if key not in d:
        d[key] = []
    d[key].append(value)
```

āęĆæđIJä;ęçŤl defaultdict çŽĐērlāzççāAārşæŽŤ'āŁāçōĀæt' AāžEriijŽ

```
d = defaultdict(list)
for key, value in pairs:
    d[key].append(value)
```

ēēŽāyĀārRēŁĆæL'ĀēōlēōžçŽĐēŮōēcŸēūşæŤræ■ōād'ĐçŘEäy■çŽĐēōrā;Ťā;ŚçşzéŮōēcŸæIJL'ād'ğçŽĐ  
 1.15 ārRēŁĆçŽĐä;Ńā■RāĀĆ

## 3.7 1.7 ā■ŮāĚyæŌŠāžŤ

### éŮōēcŸ

ä;āæČşāLZāzžāyĀāyłā■ŮāĚyriijŃāzūāyŤāIJlē■āzçæLŮāžRāLŮāŃŮēēZāyłā■ŮāĚyçŽĐæŮūāĀŽēČ;ād

## èġċàEşæŮzæąĹ

äyžāẒEèĈ;æŬğāĹüäyĀäyĹā■ŮāĔyāy■āĔĈĉt'ăçŽĐéążāžŔiijŊā;āāŔřāzēā;£çŦĪ  
collections æĹāāĹŮäy■çŽĐ OrderedDict çśzāĀĆ  
āĪĴĹēf■äzçæŞ■ä;ĪJçŽĐæŮüāĀŽāōĈäijŽăfĹæŊĀāĔĈĉt'ăēcŋæŔŖšāĔĔæŮüçŽĐéążāžŔiijŊçd'žă;ŊāēĆāyŊiijŽ

```
from collections import OrderedDict

d = OrderedDict()
d['foo'] = 1
d['bar'] = 2
d['spam'] = 3
d['grok'] = 4
# Outputs "foo 1", "bar 2", "spam 3", "grok 4"
for key in d:
    print(key, d[key])
```

ā;Şä;āæĈşèēAæđĐāzžāyĀäyĹāŕEæĹēēĪĀēēAāžŔāĹŮāŊŮæĹŮçijŮçăAæĹŔāĔŮāzŮæāijāijŔçŽĐæŸāārĹ  
OrderedDict æŸŕēĪđāyŷæĪĴçŦĪçŽĐāĀĆ æŕŦāēĈiijŊā;āæĈşçş;çəðæŬğāĹüäzē  
JSON çijŮçăAāŔŬā■ŮæōŧçŽĐéążāžŔiijŊā;āāŔřāzēāĔĹä;£çŦĪ OrderedDict  
æĹēæđĐāzžē£ŽæăũçŽĐæŦŕæ■ōiijŽ

```
>>> import json
>>> json.dumps(d)
'{"foo": 1, "bar": 2, "spam": 3, "grok": 4}'
>>>
```

## èőĹèőž

OrderedDict āĔĔēĈĹçzt'æĹd'çĪĀäyĀäyĹæāzæ■ōēŦōæŔŖšāĔĔēāzāžŔæŬŖšāzŔçŽĐāŔŊāŔŖšēŞ;ēāĹāĀĆ  
āōĈäijŽēcŋæŦ;āĹŕēŞ;ēāĹçŽĐār;ēĈĹāĀĆārżāžŬäyĀäyĹāũşçžŔā■ŸāĪĴçŽĐēŦōçŽĐēĠ■ād'■ēŦŊāĀijäy■āijŽæŦ  
ēĪĀēēAæşĹæĐŔçŽĐæŸŕiijŊäyĀäyĹ OrderedDict çŽĐād'ğārŔæŸŕäyĀäyĹæŽōēĀŽā■ŮāĔyçŽĐāyđ'ā  
æĹĀāzēāēĆæđĪā;āēēAæđĐāzžāyĀäyĹēĪĀēēAād'ğēĠŔ OrderedDict  
āōđä;ŊçŽĐæŦŕæ■ōçzŞæđĐçŽĐæŮüāĀŽiijĹæŕŦāēĆērżāŔŮ 100,000  
ēāŊ CSV æŦŕæ■ōāĹŕäyĀäyĹ OrderedDict āĹŮēāĹäy■āŬziijĹŕiijŊ  
ēĆĈāzĹä;āārśā;ŮāzŦçzEæĪĈēaaäyĀäyŊæŸŕāŔēä;£çŦĪ OrderedDict  
āyēæĹēçŽĐāē;ād'ĐēēAād'ğē£ĠēćĪād'ŮāĔĔā■ŸæŮĹēĀŮçŽĐā;śāŞ■āĀĆ

## 3.8 1.8 ā■ŮāĔyçŽĐē£ŔçőŮ

### éŮőēćŸ

æĀŬæăũāĪĹæŦŕæ■ōā■ŮāĔyāy■æĹğēāŊäyĀāzŽēōaçőŮæŞ■ä;ĪiijĹæŕŦāēĆæśĆæĪĀārŔāĀijāĀAæĪĀā



## èġċàEşæŮzæąŁ

èĀĈèŽŚăyŊéÍċŻĐèĈăċéĹăŔ■ăŠŊăzŭæăijæŸăârĐă■ŮăËyŋjŽ

```
prices = {  
    'ACME': 45.23,  
    'AAPL': 612.78,  
    'IBM': 205.55,  
    'HPQ': 37.20,  
    'FB': 10.75  
}
```

ăyžăžĒăŕžă■ŮăËyăĀijæŁġèăŊèőăċŮŮăŞ■ăIJijŊéĂŽăyŷéIJĀèċAă;ċŹŦÍ zip()  
ăĠ;æŦŕăĒĹăŕĒċŦőăŠŊăĀijăŔ■ċġġġĠăĹăăĈ æŕŦăċĈijŊăyŊéÍċæŸŕæşċæŁĹæIJĀăŕŔăŠŊæIJĀăđ'ġèĈăċéĹă

```
min_price = min(zip(prices.values(), prices.keys()))  
# min_price is (10.75, 'FB')  
max_price = max(zip(prices.values(), prices.keys()))  
# max_price is (612.78, 'AAPL')
```

ċşzăijijċŽĐijŊăŕŕăžċă;ċŹŦÍ zip()ăŠŊ sorted()  
ăĠ;æŦŕăĹăăŦŦăŮă■ŮăËyăŦŕă■ŋijŽ

```
prices_sorted = sorted(zip(prices.values(), prices.keys()))  
# prices_sorted is [(10.75, 'FB'), (37.2, 'HPQ'),  
#                   (45.23, 'ACME'), (205.55, 'IBM'),  
#                   (612.78, 'AAPL')]
```

æŁġèăŊġġŽăžŽèőăċŮŮăŞĐăŮŭăĂŽijŊéIJĀèċAæşĹăĐŕċŽĐăŸŕ zip()  
ăĠ;æŦŕăĹŽăžžċŽĐăŸŕăyĂăyĹăŔĹċĴċŋċŮăŮăyĂăŋăċŽĐġ■ăzċăŽĹăĈ  
æŕŦăċĈijŊăyŊéÍċċŽĐăžċăĂăŕşăijŽăžġċŦşċŦŽċŕŋijŽ

```
prices_and_names = zip(prices.values(), prices.keys())  
print(min(prices_and_names)) # OK  
print(max(prices_and_names)) # ValueError: max() arg is an empty_  
↪sequence
```

## èőĹéőž

ăċĈăđIJă;ăăIJăyĂăyĹă■ŮăËyăyŁæŁġèăŊăŽőéĂŽċŽĐăŦŕă■ċġġŕċŮŮijŊă;ăăijŽăŕŚċŦŕăőĈăžŋăžĒăž

```
min(prices) # Returns 'AAPL'  
max(prices) # Returns 'IBM'
```

èċŽăyĹċzŞăđIJăžŭăy■æŸŕă;ăæĈşċċĂċŽĐijŊăŽăăyžă;ăæĈşċċĂăIJă■ŮăËyċŽĐăĂijċŽĒăŕĹăyŁæŁġèă  
æĹŮċŮyă;ăăijŽăŕĹċŦċĹĂă;ċŹŦĹă■ŮăËyċŽĐ values() æŮžăşŦŕăĹăèġċăEşċċŽăyĹċŮőċŸijŽ

```
min(prices.values()) # Returns 10.75  
max(prices.values()) # Returns 612.78
```

āy■āzȳčŽĐæYřijNéĀŽāyŷēfZāylčzŠæđIJāRŊæāuāzšāy■æYřā;āæČšēēAçŽĐāĀĆ  
ā;āāRřēČ;ēfYæČšēēAçšēēAšāřzāžTčŽĐēTōčŽĐāfāæAřijLærTāēĆēČčg■ēČačēlāzūæāijæYřæIJāā;ŌčŽĐ  
ā;āāRřāzēāIJl min() āŠŊ max() āĠ;æTřāy■æRŘā;Ž key  
āĠ;æTřāRČæTřālēēŌūāRŮæIJāāRāĀijæLŮæIJāād gāĀijāřzāžTčŽĐēTōčŽĐāfāæAřāĀĆærTāēČijŽ

```
min(prices, key=lambda k: prices[k]) # Returns 'FB'
max(prices, key=lambda k: prices[k]) # Returns 'AAPL'
```

ā;EæYřijNāēČæđIJēfYæČšēēAā;ŮāLřæIJāāRāĀijijNā;āāRlā;ŮæLgēāNāyĀæñæšēæL;æš■ā;IJāĀ

```
min_value = prices[min(prices, key=lambda k: prices[k])]
```

āl■ēlččŽĐ zip() āĠ;æTřæŮzæāLéĀŽēfĠārEā■ŮāĒyāĀlāR■ē;ñāĀlāyž  
(āĀijijNéTō) āĒČčzĐāžRāLŮælēēgčāEšāžEāyLēfřēŮōēčYāĀĆ  
ā;ŠærTē;Čāyđ'āylāēČčzĐčŽĐæŮūāĀŽijNāĀijāijZāĒLēfZēāNærTē;ČijNčĐūāRŌæL■æYřēTōāĀĆ  
ēfZēāūčŽĐērīā;āāřsēČ;ēĀŽēfĠāyĀælāčōĀā■TčŽĐēr■āRēārēČ;ā;Lē;zælččŽĐāōđčŌrāIJlā■ŮāĒyāyLččŽĐ

ēIJāēēAæšlæĐRčŽĐæYřāIJlēōāčōŮæš■ā;IJāy■ā;fčTlāLřāžE (āĀijijNéTō)  
ārzāĀČā;Šād'Žāylāōđā;ŠæNēæIJLčŽyāRŊčŽĐāĀijčŽĐæŮūāĀŽijNéTōāijZāEšāōŽēfTāŽđčzŠæđIJāĀĆ  
ærTāēČijNāIJlæLgēāN min() āŠŊ max() æš■ā;IJčŽĐæŮūāĀŽijNāēČæđIJæAřāūgæIJāāRāLŮæIJāād

```
>>> prices = { 'AAA' : 45.23, 'ZZZ': 45.23 }
>>> min(zip(prices.values(), prices.keys()))
(45.23, 'AAA')
>>> max(zip(prices.values(), prices.keys()))
(45.23, 'ZZZ')
>>>
```

## 3.9 1.9 æšēæL;äyđ'ā■ŮāĒyčŽĐčŽyāRŊčČz

### ēŮōēčY

æĀŌæāūāIJlāyđ'āylā■ŮāĒyāy■ārzāræL;čŽyāRŊčČzřijLærTāēČčŽyāRŊčŽĐēTōāĀçŽyāRŊčŽĐāĀij

### ēgčāEšæŮzæāL

ēĀČēŽšāyNélčāyđ'āylā■ŮāĒyřijŽ

```
a = {
    'x' : 1,
    'y' : 2,
    'z' : 3
}

b = {
    'w' : 10,
    'x' : 11,
```

```
'y' : 2
}
```

äyžāẒẒæſ■ä;IJāzšāRřāzēçTlāzŌāŁōæTzæLŪēĀĒēŁGæzd'ā■ŪāĒyāĒĈçt'āāĀĆ  
keys() æLŪēĀĒ items() æŪzæşTēŁTāZđçzŞæđIJāyŁæLgëąNéZĒĀRLæŞ■ä;IJāĀĆæŕTāçĈiijŽ

```
# Find keys in common
a.keys() & b.keys() # { 'x', 'y' }
# Find keys in a that are not in b
a.keys() - b.keys() # { 'z' }
# Find (key,value) pairs in common
a.items() & b.items() # { ('y', 2) }
```

èŁZāẒZæŞ■ä;IJāzšāRřāzēçTlāzŌāŁōæTzæLŪēĀĒēŁGæzd'ā■ŪāĒyāĒĈçt'āāĀĆ  
æŕTāçĈiijNāAĠāçĀ;āæĈşāzēçŌŕæIJLā■ŪāĒyæđDēĀāyĀāyŁæŌŠéZđ'āGāāyŁæNĠāōZéTōçŽDæŪŕā■ŪāĒ  
āyNéŁcāŁ'çTlā■ŪāĒyæŌlāŕijæŁēāōđçŌŕēŁZæāūçŽDēIJĀæśĈiijŽ

```
# Make a new dictionary with certain keys removed
c = {key:a[key] for key in a.keys() - {'z', 'w'}}
# c is {'x': 1, 'y': 2}
```

## ëöłëőż

äyĀāyŁa■ŪāĒyāŕsæYřayĀāyŁēTōēZEāRLāyŌāĀijēZEāRLçŽDæYāārDāĒşçşzāĀĆ  
ā■ŪāĒyçŽD keys() æŪzæşTēŁTāZđayĀāyŁāsTçŌŕēTōēZEāRLçŽDēTōēgĒāZ;āŕzēsāāĀĆ  
éTōēgĒāZçŽDāyĀāyŁāŁārŠēcñāzĒēgççŽDçL'zæĀgāŕsæYřāōČāznāzşæTŕæNĀéZEāRLæŞ■ä;IJiijNærTāçĈ  
æL'ĀāzēŕijNāçĀēđIJā;āæĈşāŕzéZEāRLçŽDēTōæL'gëąNāyĀāzZæZōēĀZçŽDēZEāRLæŞ■ä;IJiijNāRřāzēçZt'  
setāĀĆ

ā■ŪāĒyçŽD items() æŪzæşTēŁTāZđayĀāyŁāNĒāRń (éTōiijNāĀij)  
āŕzçŽDāĒĈçt'æēgĒāZ;āŕzēsāāĀĆ èŁZāyŁārzēsāāRŊæāūāzşæTŕæNĀéZEāRLæŞ■ä;IJiijNāzūāyTāRřāzēēčnçT

ārçōā■ŪāĒyçŽD values() æŪzæşTāzşæYřçşzāijijijNā;ĒæYřāōČāzūāy■æTŕæNĀēŁZéĠNāzNçz■ç  
æŞŔçg■çĠNāzçāyŁæYřāZāyzaĀijēgĒāZ;āy■ēČ;āŁēŕĀæL'ĀæIJLçŽDāĀijāzŞāy■çZyāRŊiijNēŁZæāūāijZār  
āy■ēŁĠiijNāçĀēđIJā;āçāñēçĀāIJlāĀijāyŁēŁcāL'gëąNēŁZāzZēZEāRLæŞ■ä;IJçŽDēŕliijNā;āāRřāzēāĒLārĒāĀ  
setiijNçDūāRŌāĒ■æL'gëąNéZEāRLēŁŕççŌŪārśēąNāzĒāĀĆ

## 3.10 1.10 āŁāēZđ'āžRāLŪçŽyāRŊāĒĈçt'āāzūāŁiæNĀéąžāžŔ

### éŪōécŸ

æĀŌæāūāIJlāyĀāyŁāžRāLŪāyŁēŁcāŁiæNĀāĒĈçt'āéąžāžŔçŽDāRŊæŪūæŪŁéZđ'éĠāđ'■çŽDāĀijijş

### èğçāĒşæŪzæąŁ

āçĀēđIJāžRāLŪāyŁçŽDāĀijēČ;æYřhashable çşzādŊiijNéĈçāzŁāRřāzēāŁçōĀā■TçŽDāŁ'çTlēZEā

```
def dedupe(items):
    seen = set()
    for item in items:
        if item not in seen:
            yield item
            seen.add(item)
```

äyÑéÍcæYřä;ŁçŤlăyLèŁřăĜ;æŤřçŽDăĹNă■ŘiiJŽ

```
>>> a = [1, 5, 2, 1, 9, 1, 5, 10]
>>> list(dedupe(a))
[1, 5, 2, 9, 10]
>>>
```

èŁŽăylăŮžæŝŤăžĚăžĚăĹĹăžŔăĹŮăy■ăĚČť'ăăyž hashable  
çŽDăŮŮăĂŽăĹ■çőăŤlăĂĆ äĉCăđĪă;ăăČŝăŮĹéŽď'ăĚČť'ăăy■ăŔřăŚĹăyŇiiJĹăŕŤăĉĆ  
dict çşăđŇiiJĹçŽDăžŔăĹŮăy■éĜăđ'■ăĚČť'ăçŽDěŕĹiiJŇă;ăéĪĂĊăAăŕĚăyLèŁřăžčăăAçĹ■ăġăŕăŤăŕYăy/

```
def dedupe(items, key=None):
    seen = set()
    for item in items:
        val = item if key is None else key(item)
        if val not in seen:
            yield item
            seen.add(val)
```

èŁŽéĜŇçŽDkeyăŔĆăŤŕăŇĜăőŽăžĚăyĂăylăĜ;æŤřiiJŇăŕĚăžŔăĹŮăĚČť'ăĊ;Ňă■ćăĹŔ  
hashable çşăđŇăĂĆăyŇéÍcæYřăőČçŽDçŤlăŝŤçď'žăĹŇiiJŽ

```
>>> a = [ {'x':1, 'y':2}, {'x':1, 'y':3}, {'x':1, 'y':2}, {'x':2, 'y':4}]
>>> list(dedupe(a, key=lambda d: (d['x'],d['y'])))
[{'x': 1, 'y': 2}, {'x': 1, 'y': 3}, {'x': 2, 'y': 4}]
>>> list(dedupe(a, key=lambda d: d['x']))
[{'x': 1, 'y': 2}, {'x': 2, 'y': 4}]
>>>
```

ăĉCăđĪă;ăăČŝăŝžăžŎă■Ťăylă■ŮăőťăĂăăśđăĂĝăĹŮăĚĂăŝŔăylăŽť'ăđ'ĝçŽDăŤŕă■őçžŝăđĐăĹăĊăŮ

èőĹéőž

ăĉCăđĪă;ăăžĚăžĚăŕŝăYřăČŝăŮĹéŽď'éĜăđ'■ăĚČť'ăiiJŇéĂŽăyăăŔăžăčăőĂă■ŤçŽDăđĐăĂăyĂăylă

```
>>> a
[1, 5, 2, 1, 9, 1, 5, 10]
>>> set(a)
{1, 2, 10, 5, 9}
>>>
```

çDŮăĂŇiiJŇĚŁŽçĝăăŮžæŝŤăy■Ċçzt'ăĹď'ăĚČť'ăçŽDăžăžŔiiJŇçŤŝăĹŔçŽDçžŝăđĪăy■çŽDăĚČť'

åIJæIJñèLĆäy■æLŠäzñä;£çTlāžEçTšæLŘāZlāG;æTṛèol'æLŠäzñçŽDāG;æTṛæŽt'āLæĀŽçTlījNäy■āz  
ærTāēĆījNāēCædIJāēCædIJä;āæČšèrZāRŮäyĀäyIæŮGāzūījNæūLÉZd' éG■ād' ■èqNījNä;āāRfāzēā;LāōZæY

```
with open(somefile, 'r') as f:
for line in dedupe(f):
    ...
```

äyLèfṛkeyāG;æTṛāRĆæTṛæIāzēfāžE sorted() , min() āŠN max()  
ç■L'āEĖç;ōāG;æTṛçŽDçŽyāījāLšèC;āĀC āRfāzēāRĆèĀC 1.8 āŠN 1.13  
ārRèLĆāžEèğçæŽt'ād'ŽāĀC

## 3.11 1.11 āŚ;āŘ■āLĜçL'Ĝ

### éUóécŸ

ä;āçŽDćlNāžRāušçzRāGžçŌřāyĀad' gāāEāušæŮāæşTçŽt'èğEçŽDçañçijŮçāAāLĜçL'ĜāyNæāGījNçDū

### èğçāEşæŮzæāL

āAĜāōZā;āæIJL'äyĀæōtāzççāAèçAāzŌäyĀäyIèōřā;Tā■Ůçñçäyşäy■āGāäyIāZžāōZā;■ç;ōæRRāRŮāGžç

```
#####
↪0123456789012345678901234567890123456789012345678901234567890'
record = '.....100 .....513.25 ..... '
cost = int(record[20:23]) * float(record[31:37])
```

äyŌāĖŮéCçæāūāEŽījNäyžāzĀāZLäy■æČšèfZæāūāŚ;āŘ■āLĜçL'ĜāŚćījŽ

```
SHARES = slice(20, 23)
PRICE = slice(31, 37)
cost = int(record[SHARES]) * float(record[PRICE])
```

çñnāžNçğ■çL'ĹæIJñäy■ījNä;æEĀfāĖ■āžEād' gēGRæŮāæşTçŘEèğççŽDçañçijŮçāAäyNæāGījNä;fā;Ů

### èóIèōž

äyĀèLñæIèèōšījNāzççāAäy■āēCædIJāGžçŌřad' gēGRçŽDçañçijŮçāAäyNæāGāĀījāījZā;fā;ŮāRfèřæ  
ærTāēĆījNāēCædIJä;āāZðēfGæIèçIJNçIJNäyĀāzt'āL■ā;āāEŽçŽDāzççāAījNä;āāījZæSýçIĀēDŠèçNæČšéC  
èfZéGŃçŽDèğçāEşæŮzæāLæYřāyĀäyIā;LçōĀā■TçŽDæŮzæşTèol'ä;āæŽt'āLāæyĖæŽřçŽDèāIè;āzççāAāL

āEĖç;ōçŽD slice() āG;æTṛāLZāzžāžEäyĀäyIāLĜçL'ĜārZèşāījNāRfāzèèçñçTlāIJlāzžā;TāLĜçL'ĜāĖ

```
>>> items = [0, 1, 2, 3, 4, 5, 6]
>>> a = slice(2, 4)
>>> items[2:4]
[2, 3]
>>> items[a]
```



```
[2, 3]
>>> items[a] = [10, 11]
>>> items
[0, 1, 10, 11, 4, 5, 6]
>>> del items[a]
>>> items
[0, 1, 4, 5, 6]
```

æĊædIJä;äæIJL'äyÄäyläLGçL'GärzèsaaijNä;äâRfäzèäLEäLnèrĊçTláoĊçŽD a.start, a.stop, a.step äsdæÄgælēèÖuäRŮæZt'äd'ŽçŽDäæAřäÄĊærTæĊriiŽ

```
>>> a = slice(5, 50, 2)
>>> a.start
5
>>> a.stop
50
>>> a.step
2
>>>
```

âRèad'ŮriiNä;æēYèĊ;éÄŽēfGèrĊçTláoLGçL'GçŽD indices(size)  
æŮzæşTärEáoĊæYäârDäLräyÄäylçaðáoŽad'gärRçŽDäžRäLŮäyLriiN  
èfZäylæŮzæşTèfTäZđäyÄäyläyL'äĊçžD (start, stop, step)  
riiNæL'ÄæIJL'äÄijéĊ;äijŽècnäRĹéÄĊçŽDçijl'ârRäzèæzaèúşè;žçTŹéŽRäLŮriiN  
äzÖèÄNä;ççTlçŽDæŮüäÄŽéAřäÄĊæGžçŮ IndexError äijĊäyYäÄĊærTæĊriiŽ

```
>>> s = 'HelloWorld'
>>> a.indices(len(s))
(5, 10, 2)
>>> for i in range(*a.indices(len(s))):
...     print(s[i])
...
W
r
d
>>>
```

## 3.12 1.12 äžRäLŮäy■äGžçŮræñqæTřæIJÄäd'ŽçŽDäĊçT'ä

### éŮóécŸ

æÄŮæäüæL;äGžäyÄäyläžRäLŮäy■äGžçŮræñqæTřæIJÄäd'ŽçŽDäĊçT'ääŚćiiş

### èğċâEşæŮzæaL

collections.Counter çşzârşæYřäyŞéŮlâyžèfŽçşzéŮóécŸèÄŹèöçŽDriiN  
áoĊçTŽèGşæIJL'äyÄäylæIJL'çTlçŽD most\_common() æŮzæşTçZt'æŮèçzŽäžEä;ăç■TæaLäĊ

äyžžEæijTçd' ziiijNăĚĹăAĜeōĭä;ăæIJL'äyÄäyĹă■Tēr■ăĹŪeăĹăzūäyTæČşæL'ĭăĜžăŞĹăyĹă■Tēr■ăĜžčŎřé

```
words = [
    'look', 'into', 'my', 'eyes', 'look', 'into', 'my', 'eyes',
    'the', 'eyes', 'the', 'eyes', 'the', 'eyes', 'not', 'around',
    ↪ 'the',
    'eyes', "don't", 'look', 'around', 'the', 'eyes', 'look', 'into
    ↪ ',
    'my', 'eyes', "you're", 'under'
]
from collections import Counter
word_counts = Counter(words)
# äĜžčŎřéčŚçŎĜæIJĀĕñŸçŽĎ3äyĹă■Tēr■
top_three = word_counts.most_common(3)
print(top_three)
# Outputs [('eyes', 8), ('the', 5), ('look', 4)]
```

## èóĹèőž

äĭIJäyžèĭŞăĔĕiijN Counter äržèşăăRřäzèæŎĕăRŪăzzæĎRçŽĎçTşăRřăŞĹăyNiiijLhashableiijLăĔĈ  
ăIJĹăžTşăCăōđçŎřăyĹiiijNăyÄäyĹ Counter äržèşăărsæŸřăyÄäyĹă■ŪăĔyiiijNăřĒăĔĈçt'ăæŸăăřĎăĹăőCăĜžç

```
>>> word_counts['not']
1
>>> word_counts['eyes']
8
>>>
```

ăĕĆăđIJă;ăæČşæL'NăĹăćđăĹăeōăæTĭiiijNăRřäzèçčŎĂă■TçŽĎçTĹăĹăæşTĭijŽ

```
>>> morewords = ['why', 'are', 'you', 'not', 'looking', 'in', 'my', 'eyes']
>>> for word in morewords:
...     word_counts[word] += 1
...
>>> word_counts['eyes']
9
>>>
```

ăĹŪeĂĔă;ăăRřäzèăĭĕçTĭ update() æŪžæşTĭijŽ

```
>>> word_counts.update(morewords)
>>>
```

CounterăōđăĭNăyÄäyĹéśIJäyžăžžçşĕçŽĎçL'zæĂĝæŸřăőCăžňăRřäzèăĭĹăőžæŸşçŽĎeüşæTřă■ĕĕĹRç

```
>>> a = Counter(words)
>>> b = Counter(morewords)
>>> a
Counter({'eyes': 8, 'the': 5, 'look': 4, 'into': 3, 'my': 3, 'around
    ↪ ': 2,
```

```

"you're": 1, "don't": 1, 'under': 1, 'not': 1})
>>> b
Counter({'eyes': 1, 'looking': 1, 'are': 1, 'in': 1, 'not': 1, 'you': 1, 'my': 1, 'why': 1})
>>> # Combine counts
>>> c = a + b
>>> c
Counter({'eyes': 9, 'the': 5, 'look': 4, 'my': 4, 'into': 3, 'not': 2, 'around': 2, 'you're': 1, 'don't': 1, 'in': 1, 'why': 1, 'looking': 1, 'are': 1, 'under': 1, 'you': 1})
>>> # Subtract counts
>>> d = a - b
>>> d
Counter({'eyes': 7, 'the': 5, 'look': 4, 'into': 3, 'my': 2, 'around': 2, 'you're': 1, 'don't': 1, 'under': 1})
>>>

```

ærnæUäçŮséŮöiijŇ Counter ářžèśaǎIJlǎGǎázŎæL'ĂæIJL'éIJĀèçAǎLúèaǎæLŮèĂĚèóæTǝæTǝæ■óçŽlǎIJlègčǎEşèŁŻçśzéŮóécŸçŽDæŮüǎĀZǎjǎázTèřèaijŸǎĚŁéĀL'æŇl'ǎóČiijŇèĀŇäy■æŸræL'ŇǎĚŁçŽDǎL'çTlǎ

### 3.13 1.13 éĂŽèŁGæşŘäyǎĚşéTóǎ■ŮæŎŞǎžŘäyĀäyǎ■ŮǎĚyǎĹŮèǎĹ

#### éŮóécŸ

äjǎæIJL'äyĀäyǎ■ŮǎĚyǎĹŮèǎĹiijŇǎjǎæČşæǎžæ■óæşŘäyǎĚŮæşŘǎGǎäyǎ■ŮǎĚyǎ■ŮæóŝæĬæŎŞǎžŘè

#### ègčǎEşşæŮzæǎĹ

éĂŽèŁGǎjŁçTlǎ operator æǎǎǎŮçŽDǎ itemgetter  
 ǎĜjǎTǝiijŇǎRřǎžéĬđǎyǎǎóžæŸşçŽDæŎŞǎžŘèŁZæǎüçŽDæTǝæ■óçžşæđDǎĂČ  
 ǎĀĜèőçäjǎázŎæTǝæ■óǎžşäy■æčĂçt'čǎĜžæĬèçjşçŇZǎijŽǎSŸǎǎæAǎǎĹŮèǎĹiijŇǎžüäyTǎžèäyŇǎĹŮçŽDæTǝæ

```

rows = [
    {'fname': 'Brian', 'lname': 'Jones', 'uid': 1003},
    {'fname': 'David', 'lname': 'Beazley', 'uid': 1002},
    {'fname': 'John', 'lname': 'Cleese', 'uid': 1001},
    {'fname': 'Big', 'lname': 'Jones', 'uid': 1004}
]

```

æǎžæ■óǎžæĐRçŽDǎ■ŮǎĚyǎ■ŮæóŝæĬæŎŞǎžŘèçşǎĚèçžşæđIJèǎŇæŸrǎĹLǎóžæŸşǎóđçŎřçŽDiiijŇǎžç

```

from operator import itemgetter
rows_by_fname = sorted(rows, key=itemgetter('fname'))
rows_by_uid = sorted(rows, key=itemgetter('uid'))

```

```
print(rows_by_fname)
print(rows_by_uid)
```

äzčçăAçŽĎè;ŞăĞžăęĆäÿŊiijŽ

```
[{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'},
{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'}]
[{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'},
{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'},
{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'}]
```

```
rows_by_lfname = sorted(rows, key=itemgetter('lname', 'fname'))
print(rows_by_lfname)
```

äijŽăžğçŦşăęĆăyŦçŽĎè;ŞăĞžiiŦ

```
[{'fname': 'David', 'uid': 1002, 'lname': 'Beazley'},
{'fname': 'John', 'uid': 1001, 'lname': 'Cleese'},
{'fname': 'Big', 'uid': 1004, 'lname': 'Jones'},
{'fname': 'Brian', 'uid': 1003, 'lname': 'Jones'}]
```

èóíèőž

```

    aIJaYLeIcā.NāRāyN rows ećnāijāeAŠčZæŌēāRŪāyĀāyIāEšēTōāŪāRĆæTṛčZD
sorted() aĖĖčōāGjæTṛāĀĆ eŁZāyIāRĆæTṛæYř callable čšzādNijNāžūāyTāzŌ rows
āyāŌēāRŪāyĀāyIāTāyĀāĖČčřārijNčDūāRŌēŁTāZđećnčTlāIēāŌŠāžRčZDāAijaĀĆ
itemgetter() āGjæTṛārsæYřet šet čālZāzzēŁZāyI callable āržešacZDāĀĆ

```

operator.itemgetter()      åĠ;æTṙæIJL'äYÄäyļēcñ      rows

äy■çŽDēōŗā;TçTlāēāēšēāL'ıāAijçŽDçt'cāijTāRĈCæTṙāĈCāRfāzēāYṙāyÄäyļa■ŮāĒYēTōāR■çğriijN

äYÄäyļaTṙ'ā;cāAijæLŮēÄēāzzā;TēÇ;ād'šāijāāĒēāYÄäyļāfzēšçŽD      \_\_getitem\_\_()

æŮzæşTçŽDāAijaĈC      æÇædIJā;āaijāāĒēād'Žäyļçt'cāijTāRĈCæTṙçzŽ      itemgetter()

ijjNāōÇçTšæLRçŽD callableāržēšāaijŽēfTāZdāyÄäyļaNēāRnāL'ÄæIJL'āĒÇçt'āāAijçŽDāĒÇçzDijjN

āzūäyT sorted()      åĠ;æTṙaijZæāzā■ōēfŽäyļaĒÇçzDäy■āĒÇçt'āēāzāzRāŌzæŌšāzRāĈC

ä;Eā;āæÇšēçAāRñæŮūāIJlāGäyļa■ŮāōtäyLēlçēfZēāNæŌšāzRijjLæṙTæCéÄŽēfGāgŞāŠñāR■ælēæŌšāz

itemgetter()	æIJL'æUúăĂZăžșăRăřăžčŤí	lambda
èàlè;,:âijRăžčæZfiiĴŋæŕTăçĆiiŽ		

```
rows_by_fname = sorted(rows, key=lambda r: r['fname'])
rows_by_lfname = sorted(rows, key=lambda r: (r['lname'], r['fname']))
```

```

    ĕŁŻçġ■æŨzæłŁäžšăy■éŤZăĂĆă;EæYřijŇă;ŁćŦl                                     itemgetter()
    æŨzâijRaijŽēfRëąŇčŽĐćl■łā;őăŋńçĆzăĂĆăZăæ■đřijŇăęCăđIJă;ăărzæĂğěČ;èęAăśCărŢe;ĆénYčŽĐerlărs
    itemgetter() æŨzâijRăĂĆ

```

`min()` and `max()` can be used with a callable object that takes a single argument and returns a value. For example, to find the minimum and maximum values of the 'uid' attribute in a list of dictionaries, you can use the following code:

```
>>> min(rows, key=itemgetter('uid'))
{'fname': 'John', 'lname': 'Cleese', 'uid': 1001}
>>> max(rows, key=itemgetter('uid'))
{'fname': 'Big', 'lname': 'Jones', 'uid': 1004}
>>>
```

## 3.14 1.14 `sorted()` and `sorted()` with a callable

### `sorted()`

The `sorted()` function can be used to sort a list of objects based on a callable object that takes a single argument and returns a value. For example, to sort a list of dictionaries by the 'uid' attribute, you can use the following code:

### `sorted()`

The `sorted()` function can be used to sort a list of objects based on a callable object that takes a single argument and returns a value. For example, to sort a list of dictionaries by the 'uid' attribute, you can use the following code:

```
class User:
    def __init__(self, user_id):
        self.user_id = user_id

    def __repr__(self):
        return 'User({})'.format(self.user_id)

def sort_notcompare():
    users = [User(23), User(3), User(99)]
    print(users)
    print(sorted(users, key=lambda u: u.user_id))
```

The `sorted()` function can be used to sort a list of objects based on a callable object that takes a single argument and returns a value. For example, to sort a list of dictionaries by the 'uid' attribute, you can use the following code:

```
>>> from operator import attrgetter
>>> sorted(users, key=attrgetter('user_id'))
[User(3), User(23), User(99)]
>>>
```

## ěóĹěőŽ

ěĀĹæŇĹăĴčĹĹ      lambda      âĜĵæŤræĹŮěĂĚæŸĹ      attrgetter()  
ârĹrěČĵârŮâĚşăžŌăŷĹăžžâŮĪĴăěĵăĂĆ      äĵĚæŸĹĵĴŇ      attrgetter()  
âĜĵæŤrěĂžăŷŷăĵĴžěĹŖëăŇčŽĎăĹŇčČžĵĴŇăžŷăŷŤěĹŸěČĵârŇæŮăăĚăěőŷăĎ'ŽăŷĹă■ŮăěőĹěĹžëăŇærŤěĴČăĂ  
ěĹŽăŷĹěűş operator.itemgetter() âĜĵæŤrăĵĴčĹĹăžŌă■ŮăĚŷčşăĎŇăĴĹčşăĵĵĵĵĵĴĹăŖČěĂĆ1.13ârĹ  
ăĴŇăĹčČĵĴŇăĹčČăĎĪ User âőĎăĴŇěĹŸæĪĴăŷĂăŷĹ first\_name âŇŇ last\_name  
âşĎăĂĝĵĴŇěČăžĹĹăŖăžěăŖŖŖŷŇěĪčěĹžăăăăŌŖăžŖĵĴ

```
by_name = sorted(users, key=attrgetter('last_name', 'first_name'))
```

ârŇæăăăĪĂěĹĂăşĹăĎŖčŽĎăŸĹĵĴŇěĹžăŷĂârĹĹčĴĹĹăĹčŽĎăĹăĪŖârŇæăăăĂĆĴĹăžŌăČŖ  
min() âŇŇ max() äžŇčşăžŽĎăĜĵæŤrăĂĆăĹŤăĹčĴĵ

```
>>> min(users, key=attrgetter('user_id'))
User(3)
>>> max(users, key=attrgetter('user_id'))
User(99)
>>>
```

## 3.15 1.15 éĂŽèĹĜæşŖăŷĹă■ŮăěőĹăŖĚěőŖăĵĹăĹĚčžĎ

### éŮěěŸ

ăĵăăĪĴăŷĂăŷĹă■ŮăĚŷæĹŮěĂĚăőĎăĴŇčŽĎăžŖăĹŮĵĴŇčĎăăŖŌăĵăăČşăăžă■ăăşŖăŷĹčĹ'žăőžčžĎă■  
date æĹăăĹĚčžĎĎĎ■ăžčěőĹéŮăăĂĆ

### ěĝčăĚşæŮžæăĴĹ

itertools.groupby() âĜĵæŤrăŖăžăžŌĎĹžăăăžĎăŤræ■ăăĹĚčžĎăş■ăĵĴĹĎăŷŷăőĎčĹĹăĂĆ  
ăŷžăžĚăĵĴĴčĎ'žĵĴŇăĂĜěőĴăĵăăăşčžŖăĪĴăžĚăŷŇăăĹŮčžĎă■ŮăĚŷăĹŮăăĵĴĴ

```
rows = [
    {'address': '5412 N CLARK', 'date': '07/01/2012'},
    {'address': '5148 N CLARK', 'date': '07/04/2012'},
    {'address': '5800 E 58TH', 'date': '07/02/2012'},
    {'address': '2122 N CLARK', 'date': '07/03/2012'},
    {'address': '5645 N RAVENSWOOD', 'date': '07/02/2012'},
    {'address': '1060 W ADDISON', 'date': '07/02/2012'},
    {'address': '4801 N BROADWAY', 'date': '07/01/2012'},
    {'address': '1039 W GRANVILLE', 'date': '07/04/2012'},
]
```

čŖăĪĴăĂĜěőĴăĵăăČşăĪĴăŇĹ date âĹĚčžĎăŖŖčžĎăŤræ■ăăĪŮăŷĹĹěĹžëăŇěĹ■ăžčăĂĆăŷžăžĚăĹžăăăă  
date)ăŖŖăžŖĵĴŇ čĎăăŖŖčĴĴĹ itertools.groupby() âĜĵæŤĵĴĴ

```
from operator import itemgetter
from itertools import groupby

# Sort by the desired field first
rows.sort(key=itemgetter('date'))

# Iterate in groups
for date, items in groupby(rows, key=itemgetter('date')):
    print(date)
    for i in items:
        print(' ', i)
```

è£ŘèąŃçżŞæđIJiijŽ

```
07/01/2012
  {'date': '07/01/2012', 'address': '5412 N CLARK'}
  {'date': '07/01/2012', 'address': '4801 N BROADWAY'}
07/02/2012
  {'date': '07/02/2012', 'address': '5800 E 58TH'}
  {'date': '07/02/2012', 'address': '5645 N RAVENSWOOD'}
  {'date': '07/02/2012', 'address': '1060 W ADDISON'}
07/03/2012
  {'date': '07/03/2012', 'address': '2122 N CLARK'}
07/04/2012
  {'date': '07/04/2012', 'address': '5148 N CLARK'}
  {'date': '07/04/2012', 'address': '1039 W GRANVILLE'}
```

èóìèőž

groupby ( ) áĠæŧŕæĴ'ŋæŔŔæŧŧ'äŷłăŕăĴŰăŷüăŧăššæĴŷ'èĤđçz■çŽŷăŔŇăĀġġġĴăĴŰēĀĖăăžăæ■ő

key áĠæŧŕēĤŧăŹđăĀġçŽŷăŔŇġġĴŷŽđăĖĤçŧ'ăăžŔăĴŰăĀĈ

ăĴĴăĤŔăŋăĤēĤ■ăăžçŽđăŰŷăĀŹġġŇăőĈăġĴēĤŧăŹđăŷăĀŷłăĀĴăŖŇăŷăĀŷłēĤ■ăăžçăŹĴăŕžēšăġġĴŇ

ēĤŹăŷłēĤ■ăăžçăŹĴăŕžēšăăŔăžēĤŧšăĴŔăĖĤçŧ'ăăĀĴăĖĴēĤĴç■ĴăžŌăŷĴēĴēĤĈăŷłăĀġçŽđçŽđăŷ■ăĴŰăĀĖĴĴăŕ

äyÄäyléIdäyyëG■èeAçŽĐāĠEđad' Ġæ■céld' æYřeeAæázæ■ōæŃĠăōŽčŽĐā■UæotǵârEæTřæ■ōæŌšăžRăĂ  
ăZăăyž groupby () äžĚăžĚăcĀăšēēfđcz■čŽĐāĤĈct' āijjNăēĆădIJăžNăĒLăžūăşşaeIJL' ōŌšăžRăőNăĹRç

æCædIJä;äazĖäzĖāRlæYræČšæāzæo date āUæotārEæTṛæoāLĖçzDāLṛäyĀäylād'gçŽDæTṛæoçzS  
 éCčazLāj;æIJĀāē;āj;čTl defaultdict() ælēædDāzzäyĀäylād'ŽāĀijāUāĖyijNāĖšāzŌād'ŽāĀijāUāĖy  
 1.6 āRēLČæIJLēfGēreçzEçŽDāzNçzāĀČærTāēČijŽ

```
from collections import defaultdict
rows_by_date = defaultdict(list)
for row in rows:
    rows_by_date[row['date']].append(row)
```

èfZæàuçŽĐèřĭä; ääRřäzèä; Ĺè: zæĭ; çŽĐäřsèĈ; äřzæřRäyĭæNĜäoŽæŮæĬJšèöfēŮöäřzäŤçŽĐèöřä; TĭĭjŽ

```
>>> for r in rows_by_date['07/01/2012']:
...     print(r)
...
```



```
{ 'date': '07/01/2012', 'address': '5412 N CLARK' }
{ 'date': '07/01/2012', 'address': '4801 N BROADWAY' }
>>>
```

āĲāyŁÉİcēŁZāyŁā; Nā■Rāy■ĲĲNāŁŚāznāšqāĲĲāŁĒēēAāĒŁāřĒēōřā; ĲāŌŠāžRāĀĀCāZāē■d'ĲĲNāēĆāēd  
 ēŁŽčg■āŪžāĲRāĲĲāēřĲāĒŁāŌŠāžRčDūāRŌāĒēĀŽēŁĠ groupby()  
 āĠāēĲřēf■āžččŽDāŪžāĲRēŁRēāNā; ŪāŁnāyĀāžZāĀĆ

## 3.16 1.16 èŁĠæzd'āžRāĲŪāĒĈġt'ā

### éŬóécŸ

ā;āēĲĲ'āyĀāyŁāēĲřāē■ōāžRāĲŪĲĲNāēĆšāĲĲčĲĲāyĀāžZēĠDāĲZāžŌāy■āēRŖāRŪāĠžéĲĲāēēAčŽDāĀĲĲēĲ

### èġčāĒšāēŪžāēāĲ

āĲĲčŌĀā■ĲčŽDēŁĠæzd'āžRāĲŪāĒĈġt'āčŽDāŪžāēšĲřāēŸřā;ŁčĲĲāĲŪēāĲāŌĲāřĲāĀĆāēřĲāēĆĲĲĲ

```
>>> mylist = [1, 4, -5, 10, -7, 2, 3, -1]
>>> [n for n in mylist if n > 0]
[1, 4, 10, 2, 3]
>>> [n for n in mylist if n < 0]
[-5, -7, -1]
>>>
```

ā;ŁčĲĲāĲŪēāĲāŌĲāřĲčŽDāyĀāyŁāēĲĲāĲĲĲĲēēZūāřāēŸřāēĆāēdĲĲē;ŠāĒēēĲāyŸāēd'ġčŽDāŪūāĀŽāĲĲāžġč  
 āēĆāēdĲĲāĲāřāēĒēā■ŸāēřĲē;ĆāēĲRāēDšĲĲĲēĆčāžĲā;āāRřāžēā;ŁčĲĲĲšāēĲRāŽĲēāĲē;āĲRēēf■āžčāžġčĲšēŁĠ

```
>>> pos = (n for n in mylist if n > 0)
>>> pos
<generator object <genexpr> at 0x1006a0eb0>
>>> for x in pos:
...     print(x)
...
1
4
10
2
3
>>>
```

āĲĲāēŪūāĀŽĲĲNēŁĠæzd'èġDāĲZāēřĲē;Ćāēd■āēĲĲĲNāy■ēĆ;čŌĀā■ĲčŽDāĲĲāĲŪēāĲāŌĲāřĲāēĲŪēĀĒē  
 āēřĲāēĆĲĲĲNāĲĲēō;ēŁĠæzd'čŽDāŪūāĀŽēĲĲāēēĲāēd'ĲčRēāyĀāžZāĲĲāyŸāēĲŪēĀĒāēŪūāžŪāēd'■āēĲāēĈēāēĲ  
 čDūāRŌā;ŁčĲĲāēĒēāžččŽD filter() āĠāēĲřāĀĆčēd'žā;NāēĆāyNĲĲĲ

```
values = ['1', '2', '-3', '-', '4', 'N/A', '5']
def is_int(val):
```

```
filter() åĜ;æŦřăŁZăzzăžEăÿĂăÿłē■ăžcăZlíijŃăZăæ■đ'ăēĆăđIJă;ăăČșăĹŮăŁřăÿĂăÿłăĹŮăłçŽĐă
list() åŒžè;ñă■ăĈ
```

ǎĽŮeqlæÓláríjaŠNčTšæŁŘáŽléqle; ;ajjRéĂžăÿyæČěĀEtäyNāYřefĞæzd' æȚræ■ōæIJÄçõÅâ■ȚçŻĐæŬ  
ãEũãođáoČázñèfỲëÇ; ãIJłéfĞæzd' çŻĐæŬűăĂZê; ĩæ■cæȚræ■őăĂĆærTăęCiiJŻ

èĤĜæzd' æŞ■ā;IĴčŽĐäyĀäyĭāRŸčg■ārśæŸřārEäy■čņēāŘĽæIaäzüčŽĐāĀijčŦĭæŸřčŽĐāĀijazčæŽřijNēĀ  
 æřŦæčĈijNāIĴlāyĀāĽŪæŦřæ■ōäy■ā;āāRřēč;äy■āzĚæčŚşēĽ;āĽřæ■čæŦřijNēĀNäyŦēſŸæčŚşārEäy■æŸřæ■  
 éŽžēſĜārEēſĜæzd' æIaäzüāŦ;āĽřæIaäzüēāle;ā;āijRäy■āŌžijNāRřāzēā;ĽāōžæŸŚčŽĐēgčāEşēſŽäyĭēŪōēčŸ

āRēād' ŪāyĀäyłāĀijā; ŪāĒšæşİçŽĐēŁGæzd' āūēāĒūārśæŸr      itertools.  
 compress()    iijN̄    āōCžēāyĀäył    iterable    āržeśqaŠNāyĀäyłçŻyārżāžTčŽĐ  
 Boolean       éĀL' æNl' āZlāzRāLŪā; IJāyžē; ŠāĒēāRCæTřāĀĆ             çDūāŘŌē; ŞāĞž  
 iterable       āržeśaqy■ārżāžTēĀL' æNl' āZlāyž                          True       çŽĐāĒČct' āāĀĆ  
 ā; Šā; āēIJĀēēAçTlāRēād' ŪāyĀäyłçŻyāĒŞēATčŽĐāzRāLŪāIcēŁGæzd' æşŘāyłāzRāLŪćŽĐæŪūāĂZiijNēfZā  
 ærtĀeCiijNāAGāeCcŌrāIJlā; āæIJL' äyNeİcāyd' āLŪāTřæ■ōriijŽ

```
addresses = [
    '5412 N CLARK',
    '5148 N CLARK',
    '5800 E 58TH',
    '2122 N CLARK',
    '5645 N RAVENSWOOD',
```

```
'1060 W ADDISON',
'4801 N BROADWAY',
'1039 W GRANVILLE',
]
counts = [ 0, 3, 10, 4, 1, 7, 6, 1]
```

čŔŕăĬĴăĵăăČšăŕĚéČčăžŽăŕžăžŤ count âĀĭjăd' găžŔŔ5čŽĐăĬŕăĬĂăĚĬéČĬèĴŔăĜžĭĭjŇéČčăžĹăĵăăŔŕăžëèĤ

```
>>> from itertools import compress
>>> more5 = [n > 5 for n in counts]
>>> more5
[False, False, True, False, False, True, True, False]
>>> list(compress(addresses, more5))
['5800 E 58TH', '1060 W ADDISON', '4801 N BROADWAY']
>>>
```

èĤŽéĜŇčŽĐăĚşéŤŕčČžăĬĴăžŔŕăĬăĹŽăžžăŷĂăŷĬ Boolean  
 âžŔăĹŬĭĭjŇăŇĜčđ'žăŔăžžăĚČčŤ'ăçņęăŔĹăĬăžžŭăĂČ čĎŭăŔŔŔ compress()  
 âĜĵăŤŕăăžăăŕăĤăŷăŷăŕăĹŬăŔŕăĹăŇĬèĴŔăĜžăŕžăžŤăĵăĵăž True čŽĐăĚČčŤ'ăăĂČ

ăŖŇ filter() âĜĵăŤŕčşžăĭĭĭĭĭjŇ compress() âžŔăŤŕăĤăžđčŽĐăŷĂăŷĬèĤăžčăŽĬăĂČăŽăăđ'ĭĭj  
 éČčăžĹăĵăăĬĂăĚăĴăĵčŤĬ list() æĬăŕĚčžŔăđĬĲèĵăăčăŷăĹŬăĤăłčşžăđŇăĂČ

### 3.17 1.17 äžŔăŭăĚŷăŷăăŕŔăŔŬăŕŔéŽĚ

éŬŕéčŸ

ăĵăăČşăđĐéĂăŷŷĂăŷĬăŭăĚŷĭĭjŇăŕŔăŤŕăŕăđ' ŬăŷĂăŷĬăŭăĚŷčŽĐăŕŔéŽĚăĂČ

èĝčăĚşăŮžăăĴ

æĬĂăŕŔăŕăŕăŤčŽĐăŮžăĭĭŕăŔăŤŕăĴčŤĬăŭăĚŷăŕŔăŕĭĵăĂČăŕŤăĤčĭĭjŽ

```
prices = {
    'ACME': 45.23,
    'AAPL': 612.78,
    'IBM': 205.55,
    'HPQ': 37.20,
    'FB': 10.75
}
# Make a dictionary of all prices over 200
p1 = {key: value for key, value in prices.items() if value > 200}
# Make a dictionary of tech stocks
tech_names = {'AAPL', 'IBM', 'HPQ', 'MSFT'}
p2 = {key: value for key, value in prices.items() if key in tech_
     ↪ names}
```

## èóíéőž

ad' gad' ŽæTŕæČĚăĖtăyNă■ŮăĚyăŎlărijèČ;ăAŽăĹŕčŽĎrijNěĂŽēĚGăĹZăzzăyĂăylăĚČčzĎăžRăĹŮčĎŕ  
dict() ăĜ;æTŕăžšēČ;ăôđčŎŕăĂĆæŕTăēČrijŽ

```
p1 = dict((key, value) for key, value in prices.items() if value > 200)
```

ă;ĖæŸrijNă■ŮăĚyăŎlărijæŮzăijRêăĹæĎRæŽt'æyĖæŽrijNăzŭăyTăôđéŽĚăyĹăžšăijŽēĚRêăNčŽĎæŽt'  
rijĹăIJĹēĚăylă;Nă■Răy■rijNăôđéŽĚăŕNêŕTăGăăžŎæŕT dcit()  
ăĜ;æTŕæŮzăijRăĹnăTŕ'æTŕ'ăyĂă■ijĹ'ăĂĆ

æIJĹæŮŭăĂŽăôNăĹRăRŕNăyĂăzŭăžNăijŽæIJĹ'ad'ŽčĝæŮzăijRăĂĆæŕTăēČrijNčŕnăžNăylă;Nă■RčĹN

```
# Make a dictionary of tech stocks  
tech_names = { 'AAPL', 'IBM', 'HPQ', 'MSFT' }  
p2 = { key:prices[key] for key in prices.keys() & tech_names }
```

ă;ĖæŸrijNěĚRêăNăŮŭéŮt'ætNêŕTčžšæđIJæŸčđ'žēĚŽčĝæŮzăăĹăđ'gæĖĆæŕTčŕŕăyĂčĝæŮzăăĹă  
1.6ăĂăĂĆăĖĆăđIJăŕžčĹNăžRêĚRêăNăĂĝēČ;ēēAăśĆæŕTē;ČénŸčŽĎēŕIrijNěIJĂēēAēĹśčĆzæŮŭéŮt'ăŎžă  
ăĚšăžŎæŽt'ăđ'ŽēôăæŮŭăŠNăĂĝēČ;ætNêŕTrijNăRăžăăRČēĂĆ 14.13ăŕRêĹĆăĂĆ

## 3.18 1.18 æŸăăŕĎăR■čĝŕăĹŕăžRăĹŮăĚČčt'ă

### éŮóécŸ

ă;ăæIJĹ'ăyĂăôŕēĂŽēĚGăyNăăĖēôĹéŮôăĹŮēăĹăĹŮēĂĖăĚČčzĎăy■ăĚČčt'ăçŽĎăžčçăArijNă;ĖæŸŕēĚ  
ăžŎæŸŕă;ăæČšēĂŽēĚGăR■čĝŕăĹēēôĹéŮôăĚČčt'ăăĂĆ

### èĝčăĖşæŮzăăĹ

collections.namedtuple() ăĜ;æTŕēĂŽēĚGă;ĤčŦĹăyĂăylăŽôéĂŽčŽĎăĚČčzĎăŕžēşăēĹăyôă;ă  
ēĚăylăĜ;æTŕăôđéŽĚăyĹăŸŕăyĂăylēĚTăŽđPythonăy■ăăĜăĖĖăĚČčzĎčşzăđNă■RčşzčŽĎăyĂăylăŭēăŎČă  
ă;ăēIJĂēēAăijăēĂşyĂăylčşzăđNăR■ăŠNă;ăēIJĂēēAçŽĎă■ŮăôŕčžŽăôČrijNčĎŭăŔŎăôČăŕşăijŽēĚTăŽăyĂă  
ăžčçăAçđ'žă;NijŽ

```
>>> from collections import namedtuple  
>>> Subscriber = namedtuple('Subscriber', ['addr', 'joined'])  
>>> sub = Subscriber('jonesy@example.com', '2012-10-19')  
>>> sub  
Subscriber(addr='jonesy@example.com', joined='2012-10-19')  
>>> sub.addr  
'jonesy@example.com'  
>>> sub.joined  
'2012-10-19'  
>>>
```

```
>>> len(sub)
2
>>> addr, joined = sub
>>> addr
'jonesy@example.com'
>>> joined
'2012-10-19'
>>>
```

```
def compute_cost(records):
    total = 0.0
    for rec in records:
        total += rec[1] * rec[2]
    return total
```

```
from collections import namedtuple

Stock = namedtuple('Stock', ['name', 'shares', 'price'])

def compute_cost(records):
    total = 0.0
    for rec in records:
        s = Stock(*rec)
        total += s.shares * s.price
    return total
```

āš;āř■āĚĈčzĎāŘēäyÄäyĭçTĭléĀTārśæYřā;Ijāyžā■ŪāĚÿçŽĎæŽfäzçriiŇNāZāäyžā■ŪāĚyā■YāĆléIĀēēA  
 āēĆæđIĀj;āēIĀēēAæđĎāžžäyÄäyĭléidäyŷad'gçŽĎāNĚāŘnā■ŪāĚÿçŽĎæTřæ■ōçzŠæđDriiŇNēĆčāzLā;ŁçTĭlāš  
 ä;EāYřéIĀēēAæšlāĎRçŽĎæYřiiŇNäy■āČRā■ŪāĚÿēĆčæüriiŇNäyÄäyĭlāš;āř■āĚĈčzĎæYřäy■āRfæŽt'æTzç

```
>>> s = Stock('ACME', 100, 123.45)
>>> s
Stock(name='ACME', shares=100, price=123.45)
>>> s.shares = 75
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
```

```
AttributeError: can't set attribute
>>>
```

æĈædIJä;ăçIJşçŽĐéIJĀèēAæŦzâRŸâsđæĂğçŽĐăĀijīijNēĈcăzĹăRřazēä;£çŦlăŞ;ăR■ăĔĈçzĐăóđă;NçŽ  
\_replace() æŰzæşŦīijNăóĈăijŽăĹŽăzžăyĂăyĹăĒlăŰřçŽĐăŞ;ăR■ăĔĈçzĐăzŭărĒărzăžŦçŽĐă■ŰăóŧçŦlă

```
>>> s = s._replace(shares=75)
>>> s
Stock(name='ACME', shares=75, price=123.45)
>>>
```

\_replace() æŰzæşŦēſŸæIJLăyĂăyĹăĹăIJLçŦlçŽĐçĹzæĂğărsæŸră;Şă;ăçŽĐăŞ;ăR■ăĔĈçzĐăNē  
ăóĈæŸrăyĂăyĹēĹđăyŷæŰză;£çŽĐăăăăĒĒæŦræ■óçŽĐæŰzæşŦăĂĈ  
ă;ăăRřazēăĒĹăĹŽăzžăyĂăyĹăNĒăRŋijžçIJAăĀijçŽĐăŎşăđNăĔĈçzĐīijNçĐŭăRŎă;£çŦl  
\_replace() æŰzæşŦăĹŽăzžæŰřçŽĐăĀijjēcŋæŽŦæŰřēſĠççŽĐăóđă;NăĂĈæŦŦăçĈīijŽ

```
from collections import namedtuple

Stock = namedtuple('Stock', ['name', 'shares', 'price', 'date',
    ↪ 'time'])

# Create a prototype instance
stock_prototype = Stock('', 0, 0.0, None, None)

# Function to convert a dictionary to a Stock
def dict_to_stock(s):
    return stock_prototype._replace(**s)
```

ăyNēĹăæŸrăóĈçŽĐă;£çŦlăŰzæşŦīijŽ

```
>>> a = {'name': 'ACME', 'shares': 100, 'price': 123.45}
>>> dict_to_stock(a)
Stock(name='ACME', shares=100, price=123.45, date=None, time=None)
>>> b = {'name': 'ACME', 'shares': 100, 'price': 123.45, 'date':
    ↪ '12/17/2012'}
>>> dict_to_stock(b)
Stock(name='ACME', shares=100, price=123.45, date='12/17/2012',
    ↪ time=None)
>>>
```

æIJĀăRŎēēAęŦŦçŽĐæŸŦīijNăēĈædIJä;ăçŽĐçŽóæăĠæŸŦăóŽăzĹăyĂăyĹēIJĀèēAæŽŦæŰŦăĹăđŦŽăóđă;  
ēſŽăŰŭăĂŽă;ăăžŦērēēĂĈēŽŦăóŽăzĹăyĂăyĹăNĒăRŋ  
æŰzæşŦçŽĐçşīijĹăRĈēĂĈ8.4ărRēĹĈīijĹăĂĈ

\_\_slots\_\_

## 3.19 1.19 èĭñæ■cāzúāRŃæŮúèőaçóŮæTřæ■ó

### éŮóécŸ

äĭäéIJĀèĕAāIJĲæTřæ■óāžRāĹŮäyŁæL'gëaŃèAŽéŽEāĜĭæTřĭijŁæřTāĕĆ sum(), min(), max() ĭijŁ'ĭijŃ äĭEæŸřéĕŮāĚĹäĭäéIJĀèĕAāĚĹèĭñæ■cāŁŮèĀĚĕfĜæzd' æTřæ■ó

### èĝčāEşæŮzæaĹ

äyĀäyĹēĭdāyŷäĭjŸéŽĚĕŽĎæŮzāĭjRāŌzĕzŞāRĹæTřæ■óèőaçóŮäyŎèĭñæ■cāřsæŸřäĭĕĕTĲäyĀäyĲĕTřæĹŔæřTāĕĆĭijŃāĕCāđIJäĭäĕŞèőaçóŮāzşæŮzāŃĭijŃāRřæžēāĈRäyŃēĭĕĕfŽæāŭāAŽĭijŽ

```
nums = [1, 2, 3, 4, 5]
s = sum(x * x for x in nums)
```

äyŃēĭĕæŸřæŽt'ād'ŽĕŽĎäĭŃā■RĭijŽ

```
# Determine if any .py files exist in a directory
import os
files = os.listdir('dirname')
if any(name.endswith('.py') for name in files):
    print('There be python!')
else:
    print('Sorry, no python.')
# Output a tuple as CSV
s = ('ACME', 50, 123.45)
print(','.join(str(x) for x in s))
# Data reduction across fields of a data structure
portfolio = [
    {'name': 'GOOG', 'shares': 50},
    {'name': 'YHOO', 'shares': 75},
    {'name': 'AOL', 'shares': 20},
    {'name': 'SCOX', 'shares': 65}
]
min_shares = min(s['shares'] for s in portfolio)
```

### èőĹèőž

äyĹēĭĕĕŽĎĕd'žäĭŃāRŤSäĭäēĭjTĕd'žäžEāĭŞĕTřæĹŔāŽĹēāĹēĭĭāĭjRāĭIJäyžäyĀäyĹā■TĕŃŃāŔCæTřäĭjāéĀŞĕæřTāĕĆĭijŃäyŃēĭĕĕfŽäžŽēr■āRēæŸřĕ■ŁæTĲĕŽĎĭijŽ

```
s = sum((x * x for x in nums)) #_
→æŸĭĕd'žĕŽĎäĭjāéĀŞäŷĀäyĲĕTřæĹŔāŽĹēāĹēĭĭāĭjRāřžēśā
s = sum(x * x for x in nums) #_
→æŽt'āĹäāĭjŸéŽĚĕŽĎāőđĕŎřæŮzāĭjRĭĭjŃĕIJAĕTĕäžEæŃŃāŔŮ
```

äĭĕĕTĲäyĀäyĲĕTřæĹŔāŽĹēāĹēĭĭāĭjRāĭIJäyžāŔCæTřäĭjŽæřTāĚĹāĹZāžžäyĀäyĹäyŧ æŮŭāĹŮēāĹæŽt'āĹäēĭæřTāĕĆĭijŃāĕCāđIJäĭäy■äĭĕĕTĲĕTřæĹŔāŽĹēāĹēĭĭāĭjRĕŽĎērĭijŃäĭäāRřēĈĭäĭjŽēĀĈēŽSäĭĕĕTĲäyŃēĭĕĕŽĎä



```
nums = [1, 2, 3, 4, 5]
s = sum([x * x for x in nums])
```

æŁŹçġ■æŮzaijRāŔŇæūāŔŕázèè;ĭāĽŕæČšèèAçŽĎæŤĽæđIJiijŇā;EæŸŕāŏČaijŽād'ŽāyĀäyĽæ■ēēĽd'iiŇā  
 áržāžŎārRādŇāĽŮēāĽāŔŕèČ;æšāžāzĀāzĽāĔšçşziiŇā;EæŸŕāēČæđIJāĔČçt'āæŤŕēĠŔēĽđāyŷād'ğçŽĎæŮūāĀŹ  
 āŏČaijŽāĽŹāzžāyĀäyĽāūĽād'ğçŽĎāzĔāzĔēcñā;ĕçŤĽāyĀæñāŕšècñāyçaijČçŽĎāyŧ'æŮūæŤŕæ■ŏçzşæđĎāĀČè  
 āIJĽā;ĕçŤĽāyĀāzŽèAžÉZEāĠ;æŤŕæŕŤæČ min() āŠŇ max()  
 çŽĎæŮūāĀŹā;āāŔŕèČ;æŽŧ'āĽāāĀ;āŔŖSāzŎā;ĕçŤĽçŤşæĽŔāŹĽçĽĽæIJñiiŇ  
 āŏČāzñæŎēāŔŮçŽĎāyĀäyĽ key āĔşēŤŏā■ŮāŔČæŤŕæĽŮēŏyāržā;āāĽæIJĽāyŏāĽŧ'āĀČ  
 æŕŤæČiiŇāIJĽāyĽēĽççŽĎēŧAāĽyā;Ňā■Ŕāy■iiŇā;āāŔŕèČ;aijŽèĀČèŽSāyŇēĽççŽĎāŏđçŎŕçĽĽæIJñiiŇŽ

```
# Original: Returns 20
min_shares = min(s['shares'] for s in portfolio)
# Alternative: Returns {'name': 'AOL', 'shares': 20}
min_shares = min(portfolio, key=lambda s: s['shares'])
```

## 3.20 1.20 āŔĽāzūād'ŽāyĽā■ŮāĔŸæĽŮæŸāārĎ

### éŮŏéçŸ

çŎŕāIJĽæIJĽād'ŽāyĽā■ŮāĔŸæĽŮēĀĔæŸāārĎiiŇā;āæČşārEāŏČāzñāzŎēĀžè;SāyĽāŔĽāzūāyžāyĀäyĽā■  
 æŕŤæČæşēæĽ;āĀijæĽŮēĀĔæçĀæşēæşŔāzŽēŤŏæŸŕāŔēā■ŸāIJĽāĀČ

### èğçāEşæŮzæāĽ

āĀĠāēČā;āæIJĽāēČāyŇāyŧ'āyĽā■ŮāĔŸ:

```
a = {'x': 1, 'z': 3 }
b = {'y': 2, 'z': 4 }
```

çŎŕāIJĽāĀĠèŏ;ā;āāĔĔēāzāIJĽāyŧ'āyĽā■ŮāĔŸāy■æĽ'gèāŇæşēæĽ;æş■ā;IJiijĽæŕŤæČāĔĽāzŎ  
 a āy■æĽ;iiŇŇāēČæđIJæĽ;āy■āĽŕāE■āIJ b āy■æĽ;iiŇĽāĀČ  
 āyĀäyĽēĽđāyŷçŏĀā■ŤçŽĎèğçāEşæŮzæāĽāŕşæŸŕā;ĕçŤĽ collections æĽāāĽŮāy■çŽĎ  
 ChainMap çşzāĀČæŕŤæČĈiiŇŽ

```
from collections import ChainMap
c = ChainMap(a,b)
print(c['x']) # Outputs 1 (from a)
print(c['y']) # Outputs 2 (from b)
print(c['z']) # Outputs 3 (from a)
```

### èŏĽèŏž

āyĀäyĽ ChainMap æŎēāŔŮād'ŽāyĽā■ŮāĔŸāzūārEāŏČāzñāIJĽēĀžè;SāyĽāŔŸāyžāyĀäyĽā■ŮāĔŸāĀČ  
 çĎūāŔŎiiŇŇæĔŹāžŹā■ŮāĔŸāzūāy■æŸŕçIJşçŽĎāŔĽāzūāIJĽāyĀēŧūāžEŧiiŇ ChainMap

çşzâRlæYřâIJlâEĚĚČlálZázžâžEäyÄäyłăőžçžşēŁŻăžZăŰăĚyçŽĐăLŰəál  
ázúéG■æŰřăőŽăZL'ázEäyÄăžZăyÿèğAçŽĐăŰăĚyæŞ■ăIJæĬééA■ăŎĚēŁŽăyłăLŰəłăăĂĆăđ'ğéČlálĚĚăŰăĚ

```
>>> len(c)
3
>>> list(c.keys())
['x', 'y', 'z']
>>> list(c.values())
[1, 2, 3]
>>>
```

ăĚĆăđIJăGžçŎřéG■ăđ'■éŤŏijNĚĆčăžŁčňňăyĂæňăăGžçŎřçŽĐæYăârĐăĂijăijŽěćněŁŤăŽđăĂĆ  
ăŽăæ■đ'ijNăĴNă■ŘčĬNăžRăy■çŽĐ c['z'] æĂžæYřăijŽēŁŤăŽđăŰăĚy a  
ăy■ăržăžŤçŽĐăĂijijNĚĂňăy■æYř b äy■ăržăžŤçŽĐăĂijăĂĆ

ăržăžŎăŰăĚyçŽĐæŽt æŰřăĽŰăĽăéŽđ'æŞ■ăIJæĂžæYřă;śăŞ■çŽĐæYřăĽŰəłăăy■čňňăyĂäyłăŰăĚyă

```
>>> c['z'] = 10
>>> c['w'] = 40
>>> del c['x']
>>> a
{'w': 40, 'z': 10}
>>> del c['y']
Traceback (most recent call last):
...
KeyError: "Key not found in the first mapping: 'y'"
>>>
```

ChainMap âřžăžŎçijŰçĬNě■ēĬĂäy■çŽĐăIJçŤĬěŇČăŽt âRŸéĞRijŁærŤăĚĆ  
globals , locals ç■ĽijŁæYřěĬăyÿæIJŁçŤĬçŽĐăĂĆ  
ăžNăŏđăyĽijNăIJŁăyĂăžZăŰžæŞŤăRřăžă;łăŏČăRŸăĴŰçŏĂă■ŤijŽ

```
>>> values = ChainMap()
>>> values['x'] = 1
>>> # Add a new mapping
>>> values = values.new_child()
>>> values['x'] = 2
>>> # Add a new mapping
>>> values = values.new_child()
>>> values['x'] = 3
>>> values
ChainMap({'x': 3}, {'x': 2}, {'x': 1})
>>> values['x']
3
>>> # Discard last mapping
>>> values = values.parents
>>> values['x']
2
>>> # Discard last mapping
>>> values = values.parents
>>> values['x']
```

ä;IJäyž ChainMap çŽĐæZŁäzčijŃä;ăăRrèĈ;ăijŽèĂĈèZŚă;ŁçŦl update()  
æŰzæşŦarĖäy'd'äylă■ŰăĖyăŦŦăzŰăĂĈæŦăçĈijŽ

ɛfZæuäzʂɛÇjəaŋa, UéAŽijŋä; EæYraoČeIJAəAä; aaLZazäyÄäylaŋaEläy■aRŇçŽDä■UäEýárzèsäp  
 aRŇæUüijŋäČædIJAŌš■UäEýaAŽäZæZt æÜrijŇɛfZçg■aTžäRÝäy■äijZäR■azTäLræÜřčŽDäRŁazúa■

ChainMap ä;fcŤlăŎşælēcŽDă■ŬăĚyijŇăoČěĠlăuśăy■ăLZăzžæŮřcŽDă■ŬăĚyăĂĆăL'ĂăzēăoČăzŭăy

4 ċñňăžŇčnáĭĭĭŽă■ŮčņęăÿšăŠŇæŮĜæĬĴň

## Contents:

## 4.1 2.1 ä;£çŦíad'ŽäyłçŦŇăóŽçñęǎŁęǎŁ'sǎ■Ůçñęäyš

### éŮóéćŸ

ä;ǎéIJǎèçAǎřEǎyǎǎyłǎ■ŮçñęäyšǎŁęǎŁ'sǎyžǎd'Žäyłǎ■ŮǎóŧiijŇǎ;EǎŸřǎŁęéŽŦçñę(è£ŸǎIJL'ǎŚíǎŽŦ'çŽŦ

### èğčǎEşǎŮzǎǎŁ

string ǎřzèşǎçŽĐ split() ǎŮzǎşŦǎŦŦéǎĆǎžŦǎžŮéİǎyŷçŮǎǎ■ŦçŽĐǎ■ŮçñęäyšǎŁęǎŁ'sǎŦŦǎ;çŦij  
ǎŮŦǎžŮǎyǎ■ǎĚǎèçyǎIJL'ǎd'ŽäyłǎŁęéŽŦçñęǎŁŮèǎĚǎŸřǎŁęéŽŦçñęǎŚíǎŽŦ'ǎyǎ■çǎŮǎŮŽçŽĐçŦ'žǎǎijǎǎĆ  
ǎ;Şǎ;ǎéIJǎèçAǎŦŦ'ǎŁǎçAŦǎŦ'žçŽĐǎŁǎŁǎŁ'sǎ■ŮçñęäyšçŽĐǎŮǎǎŽŦijŇǎIJǎǎé;ǎ;£çŦí re.  
split() ǎŮzǎşŦŦijŽ

```
>>> line = 'asdf fjdk; afed, fjek,asdf, foo'
>>> import re
>>> re.split(r'[:,\s]\s*', line)
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
```

### èóíèóž

ǎĜ;ǎŦŦŦ re.split() ǎŸřéİǎyŷǎŮŮçŦíçŽĐŦijŇǎŽǎäyžǎŮŮǎĚǎèçyǎ;ǎäyžǎŁęéŽŦçñęǎŇǎŮŽǎd'Žäył  
ǎŦŦǎŦŦŦŦijŇǎIJǎyŁéİćçŽĐǎ;Ňǎ■ǎŸyǎŦijŇǎŁęéŽŦçñęǎŦŦǎžǎŸǎŸřǎŮǎŦŦŦijŇǎŁęǎŦŮǎŁŮèǎĚǎŸřçŦ'žǎǎijŦij  
ǎŦŦéçAǎŦŽäyłǎǎijŦéçŦǎŁ;ǎŦŦijŇéççǎžŦǎŦzéĚ■çŽĐǎŁęéŽŦçñęǎyŮ'è;ççŽĐǎŮŮǎ;ŞéÇ;ǎijŽéçŦǎ;ŞǎŁŦǎŸ  
è£ŦǎŽĐççŞǎđIJǎyžǎyǎǎyłǎ■ŮǎŮŮǎŁŮèǎŦŦijŇè£ŽäyłèŮş str.split()  
è£ŦǎŽĐǎǎijçşǎdŇǎŸřǎyǎǎǎüçŽĐǎǎĆ

ǎ;Şǎ;ǎä;£çŦí re.split() ǎĜ;ǎŦŦŦŮǎǎŽŦijŇéIJǎèçAçŦ'žǎŁŦǎşŦǎĐŦçŽĐǎŸřǎ■çǎŁŽèǎłè;ǎijŦǎy  
ǎŦŦĈǎđIJǎ;£çŦíǎžEǎ■ŦèŮǎŁęççŽĐijŇéççǎžŦéçŦǎŇzéĚ■çŽĐǎŮĜǎIJǎžşǎřEǎĜççŮǎIJççŞǎđIJǎŁŮèǎłǎy

```
>>> fields = re.split(r'(;|,|\s)\s*', line)
>>> fields
['asdf', ' ', 'fjdk', ';', 'afed', ',', 'fjek', ',', 'asdf', ',', 
  ↪ 'foo']
>>>
```

èŮŮǎŦŮǎŁęǎŁ'sǎ■ŮçñęǎIJǎşŦŦǎžŽǎŦŦǎĚǎĚǎyŇǎžşǎŸřǎIJL'çŦíçŽĐǎǎĆ  
ǎŦŦǎŦŦŦŦijŇǎ;ǎǎŦŦéç;ǎŦşǎİçŦŦǎŁęǎŁ'sǎ■ŮçñęäyšŦijŇçŦíǎłǎIJǎŦŮéİćéĜ■ǎŮŦǎđĐéǎǎyǎǎyłǎŮŦçŽĐè,

```
>>> values = fields[::2]
>>> delimiters = fields[1::2] + ['']
>>> values
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']
>>> delimiters
[' ', ';', ',', ', ', ', ', ', ', '']
>>> # Reform the line using the same delimiters
>>> ''.join(v+d for v,d in zip(values, delimiters))
'asdf fjdk;afed,fjek,asdf,foo'
>>>
```

re.split(r'(?!,|;|\\s)\\s\*', line)  
['asdf', 'fjdk', 'afed', 'fjek', 'asdf', 'foo']  
>>>

## 4.2 2.2

### éUóécŸ

ä;äéIJÄëAéÄŽëfGæNĜăóŽčŽDæŮĜæIJñælaaijRăŌzæčĂæšëăŮčņäÿšçŽDaijĂăd't æLŮčzŠăřčŽDăyĂăylčóĂăŮčzæšçTæŸřă;čçTl  
SchemeçLçLăĂĆ

### èğčăEşæŮzæąŁ

æčĂæšëăŮčņäÿšçŽDăyĂăylčóĂăŮčzæšçTæŸřă;čçTl str.  
startswith() æLŮčzĂĂæŸř str.endswith() æŮzæšTăĂĆæřTăçCiižŽ

```
>>> filename = 'spam.txt'
>>> filename.endswith('.txt')
True
>>> filename.startswith('file:')
False
>>> url = 'http://www.python.org'
>>> url.startswith('http:')
True
>>>
```

æčČăđIJă;ăäŷæČşăčĂæšëăđ'ŽçğăăŹéĚăăRfèČiijNăRlėIJÄëAăřEæLĂæIJL'čŽDăŹéĚăéązæTłăĚëăŁ  
çDúăRŌăijăçžŽ startswith() æLŮčzĂĂæŸř endswith() æŮzæšTŷijžŽ

```
>>> import os
>>> filenames = os.listdir('.')
>>> filenames
[ 'Makefile', 'foo.c', 'bar.py', 'spam.c', 'spam.h' ]
>>> [name for name in filenames if name.endswith(('.c', '.h')) ]
['foo.c', 'spam.c', 'spam.h']
>>> any(name.endswith('.py') for name in filenames)
True
>>>
```

äÿNéÍcæŸřăŘëäÿĂäÿłăNăŮRiižŽ

```
from urllib.request import urlopen
```

```
def read_data(name):
    if name.startswith(('http:', 'https:', 'ftp:')):
        return urlopen(name).read()
    else:
        with open(name) as f:
            return f.read()
```

æĖGæĀłçŽDæŸřijNèŁŻäyŁæŰzæŸTäy■āŁĖĖāzēēAēŁŠāĖĖäyĀäyŁāĖĈçzĎDāIJäyžāŖĆæŤrāĀĆ  
 āēĆāđIJā;āæAřāŭgæIJL'äyĀäyŁ list æŁŰēĀĖ set çšžādNçŽĎéĀL'æNŁ'ēāžřijN  
 ēēAçāđāŁĀijāēĀŠāŖĆæŤrāL'■āĖĖŁēřĈçŤĬtuple() āřĖāĖŰē;ñæ■cāyžāĖĈçzĎDçšžādNāĀĆæřŤāēĆřijŽ

```
>>> choices = ['http:', 'ftp:']
>>> url = 'http://www.python.org'
>>> url.startswith(choices)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: startswith first arg must be str or a tuple of str, not list
>>> url.startswith(tuple(choices))
True
>>>
```

## ēōĹēōž

startswith() āŠNendswith() æŰzæŸTæŖŖāŁZāžĖäyĀäyŁēĬđäyŷæŰzāŁçŽĎDæŰzāijŖāŌzāAžā  
 çšžāijijçŽĎDæŠ■āIJāžšāŖřāžēā;ŁçŤĬāŁĖçŁ'ĖāĖēāđđçŌřřijNā;ĖæŸřāžççāAçIJNēŁūāĖĖæšqæIJL'ēĆčāžĹāijŸē

```
>>> filename = 'spam.txt'
>>> filename[-4:] == '.txt'
True
>>> url = 'http://www.python.org'
>>> url[:5] == 'http:' or url[:6] == 'https:' or url[:4] == 'ftp:'
True
>>>
```

ājāāŖřāžēēĈ;ēŁŸæĈšā;ŁçŤĬā■čāŁŽēāĖēŁāijŖāŌzāđđçŌřřijNæřŤāēĆřijŽ

```
>>> import re
>>> url = 'http://www.python.org'
>>> re.match('http:|https:|ftp:', url)
<_sre.SRE_Match object at 0x101253098>
>>>
```

ēŁŽçg■æŰzāijŖāžšēāNā;ŰēĀŽřijNā;ĖæŸřāžžāžŌçōĀā■ŤçŽĎāNžēĖ■āđđāIJLæŸřæIJL'çĆžāŖŖāēĬŖād'g  
 æIJĀāŖŌæŖŖāyĀäyNřijNā;ŠāŠNāĖŰāžŰæŠ■ā;IJæřŤāēĆæŽōēĀŽæŤŖæ■ōēĀžāŖĹŁçŽŷçžšāŖĹŁçŽĎDæŰ  
 startswith() āŠN endswith() æŰzæŸTæŸřāŁäy■ēŤŽçŽĎāĀĆ  
 æřŤāēĆřijNäyNéĬçēŁZäyŁē■āŖēæçĀæšēæšŖāyŁæŰĖāžūāđ'žāy■æŸřāŖēā■ŸāIJLæNĖāđŽçŽĎDæŰĖāžūçšžād

```
if any(name.endswith(('.c', '.h')) for name in listdir(dirname)):
    ...
```

### 4.3 2.3 ıŤÍShelléĂŽéĚ■çęąŃzéĚ■ą■Ůçęäÿš

éŮőécŸ

ä;äăČsä;ŁçTÍ **Unix Shell** äÿ■äÿÿçTÍŁçŽĐéŽŽéĚ■çñē(æŕTāēĆ \*.py, Dat[0-9]\*.csv  
ç■L)ăŐzāŇzéĚ■æŪĜæIŇn■Ūçñēäÿš

èġčǎẸșæŮźæąŁ

```
fnmatch ælaǻlŮæRŘä;ZäžEäyd'äylåĜjæTřäĀTāĀT fnmatch() åŠŇ
fnmatchcase() iijNåRřäzčçTlæIæåđččŌřčZæåũçŽĐåNzéĚ■ăĀĆçTlæşTæçCäyNijŽ
```

```
>>> from fnmatch import fnmatch, fnmatchcase
>>> fnmatch('foo.txt', '*.txt')
True
>>> fnmatch('foo.txt', '?oo.txt')
True
>>> fnmatch('Dat45.csv', 'Dat[0-9]*')
True
>>> names = ['Dat1.csv', 'Dat2.csv', 'config.ini', 'foo.py']
>>> [name for name in names if fnmatch(name, 'Dat*.csv')]
['Dat1.csv', 'Dat2.csv']
>>>
```

f<sub>n</sub>match() ăĜ;æTřä;ľčTlāžTāsĆaŠ■ä;IļçşzçzşçŽĎad' ġārRāEžæTRăĎšëğĎāŁŽ(äy■āRŇčŽĎçşzçşç

```
>>> # On OS X (Mac)
>>> fnmatch('foo.txt', '*.TXT')
False
>>> # On Windows
>>> fnmatch('foo.txt', '*.TXT')
True
>>>
```

æCædIJä;äärzèfZäyIaŃzäLŃä;LäIJæĐRijŃäRäzëä;fçTl fnmatchcase()  
æIëäzçæZfäÄCäöČäöŃäÉIä;fçTlā;äçZĐäIäiIŘäd'gärRäEŽäŃzëĚ■äÄCærTäçCijZ

```
>>> fnmatchcase('foo.txt', '*.TXT')
False
>>>
```

ɛʃZäyð'äyläG;æTřéĀžāyŷaijŽěćnáŋ;çTęçŽDäyĀäyłçL'žæĀğæŸřaIJlād'DčŘEēlđæŮĞäzúaŘ■čŽDā■Ůç  
 æřTāęCiiŷNāAGēō;ä;äæIJL'äyĀäyłəaŮéAŠaIJřaIĀçŽDāLŮēałæTřæ■ōiijŽ



```
addresses = [
    '5412 N CLARK ST',
    '1060 W ADDISON ST',
    '1039 W GRANVILLE AVE',
    '2122 N CLARK ST',
    '4802 N BROADWAY',
]
```

āĳāāŔřăžěăĈŔēŁŻæăŭăĖŻăĹŮèąłæŌłŕĳĳĳŻ

```
>>> from fnmatch import fnmatchcase
>>> [addr for addr in addresses if fnmatchcase(addr, '* ST')]
['5412 N CLARK ST', '1060 W ADDISON ST', '2122 N CLARK ST']
>>> [addr for addr in addresses if fnmatchcase(addr, '54[0-9][0-9]_
↳*CLARK*')]
['5412 N CLARK ST']
>>>
```

èőłèőž

`fnmatch()` āĢĳæŦřăŅzéĚēĈĳăŁŻăžŅăžŌĉôĀăŦĉŽĎăŮĉņęäÿşæŮżæşŦăŠŅăĳžăđ'ğĉŽĎæċăĹŻèă  
 âĖĈăđĪĲăĪĲăŦřăēăđ'ĎĉŔĖæŞăăĲĲăŕĲĲăĖĖĀĉôĀăŦĉŽĎéĀŽéĚēĉņęăŕşèĈĳăôŅăĹŔĉŽĎæŮŭăĀŽĳĳĲ  
 âĖĈăđĪĲăĳăĉŽĎăžĉĉăĀĖĲăĖĖĀăĀŽæŮĢăžŭăŔăĉŽĎăŅzéĚēĳĳĲŅăĲăăĳăĲĉŦĲ glob  
 æĲăĲăŮăĀĈăŔĈèĀĈ5.13ăŕŔèĹĈăĀĈ

## 4.4 2.4 āŮĉņęäÿşăŅzéĚăăŠŅæŔĪĲĉŦĉ

éŮóéćŸ

ăĳăæĈşăŅzéĚēăĹŮèĂĖæŔĪĲĉŦĉĉĈĲăăŕăăŻăĲăĲăĲŔĉŽĎæŮĢăĲĲ

èğĉăĖşæŮżæąĹ

âĖĈăđĪĲăăæĈşăŅzéĚēĉŽĎæŸŕăŮéĲăăŮĉņęäÿşĳĳŅéĈĉăžĻăĳăĖĀŽăÿŷăŔĲĲăĖĖĀĖŕĈĉŦĲăşşæĲĲăăŮ  
 æŦŦăĖĈ str.find() , str.endswith() , str.startswith()  
 æĹŮèĂĖşăžăĳĳĲŽĎæŮżæşŦĳĳŻ

```
>>> text = 'yeah, but no, but yeah, but no, but yeah'
>>> # Exact match
>>> text == 'yeah'
False
>>> # Match at start or end
>>> text.startswith('yeah')
True
>>> text.endswith('no')
```

```
False
>>> # Search for the location of the first occurrence
>>> text.find('no')
10
>>>
```

árzäžŎäd'■æİĆçŽĐăŇzéĚ■éIJĀēęÄä;ęçŦlæ■čálŽèǎlè;ǎijŔăŠŇ re ælǎǎlŮăĂĆ  
 äyžäžĚęğćéĠŁæ■čálŽèǎlè;ǎijŔçŽĐăšžæIJňăŎșçŔĚřijŇăĂĜèő;ă;ăæČșăŇzéĚ■æŦřă■ŮăǎijǎijŔçŽĐæŮěæ  
 11/27/2012 ĩijŇăĭăăŔřäzèè£ŽæăũăĂŽĭijŽ

```
>>> text1 = '11/27/2012'
>>> text2 = 'Nov 27, 2012'
>>>
>>> import re
>>> # Simple matching: \d+ means match one or more digits
>>> if re.match(r'\d+/\d+/\d+', text1):
...     print('yes')
...     else:
...     print('no')
...
yes
>>> if re.match(r'\d+/\d+/\d+', text2):
...     print('yes')
...     else:
...     print('no')
...
no
>>>
```

ăęĆăedIJăĭăæČșă;ęçŦlăŔŇăyĂăyŦălǎǎijŔăŎžăĂžăd'ŽæňăăŇzéĚ■ĭijŇăĭăăžŦėřăăĚĠăřĚălǎǎijŔă■Ůçņėă

```
>>> datepat = re.compile(r'\d+/\d+/\d+')
>>> if datepat.match(text1):
...     print('yes')
...     else:
...     print('no')
...
yes
>>> if datepat.match(text2):
...     print('yes')
...     else:
...     print('no')
...
no
>>>
```

match() æĂžæŸřäžŎă■ŮçņăyšǎijĂăğŇăŎžăŇzéĚ■ĭijŇăęĆăedIJăĭăæČșășěæL'ă■ŮçņăyšăžzæĎŔė  
 ä;ęçŦl findall() æŮžæșŦăŎžăžčæŽĚăĂĆăřŦăęĆĭijŽ

```
>>> text = 'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> datepat.findall(text)
['11/27/2012', '3/13/2013']
>>>
```

findall() returns a list of all non-overlapping matches in the text. The matches are returned as a list of strings.

```
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)')
>>>
```

datepat is a compiled regular expression object. You can use it to find all matches in the text.

```
>>> m = datepat.match('11/27/2012')
>>> m
<sre.SRE_Match object at 0x1005d2750>
>>> # Extract the contents of each group
>>> m.group(0)
'11/27/2012'
>>> m.group(1)
'11'
>>> m.group(2)
'27'
>>> m.group(3)
'2012'
>>> m.groups()
('11', '27', '2012')
>>> month, day, year = m.groups()
>>>
>>> # Find all matches (notice splitting into tuples)
>>> text
'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> datepat.findall(text)
[('11', '27', '2012'), ('3', '13', '2013')]
>>> for month, day, year in datepat.findall(text):
...     print('{}-{}-{}'.format(year, month, day))
...
2012-11-27
2013-3-13
>>>
```

finditer() returns an iterator that yields match objects. This is useful when you want to process each match individually.

```
>>> for m in datepat.finditer(text):
...     print(m.groups())
...
('11', '27', '2012')
('3', '13', '2013')
>>>
```

## èóíëõž

āšžāžŌæ■čāLŽēālēĭāijRçŔEèōžçŽDæTžčlŃāũšçžRèūĒāGžāžEæIJñāžēçŽDēŃčāžt'āĀĆ  
āy■ēfGīijNēfZāyĀēLČēYŔēfřāžEā;fçTlreālāāUēfZēāNāNžēĒ■āšNæŔIJçt'cæŪGæIJñçŽDæIJāāšžæIJñæ  
æāyāfČæ■ēēld'rāšæYŔāĒLā;fçTl re.compile() çijŪērSæ■čāLŽēālēĭāijRā■ŪçņēāyšīijN  
çĎūāŔŌā;fçTl match() , findall() æLŪēĀĒ finditer() ç■LæŪžæšTāĀĆ

ā;ŠāEŽæ■čāLŽāijRā■ŪçņēāyšçŽDæŪūāĀŽīijNçŽyāržæŽōēA■çŽDāAŽæšTæYŔā;fçTlāŌšāgNā■Ūçņēā  
r'(\d+)/(\d+)/(\d+)' āĀĆ èfZçg■ā■ŪçņēāyšārEāy■āŌžēgčædŔāŔ■æŪIJæIāīijNēfZāIJāæ■čāLŽēālē  
āēČædIJāy■ēfZæāūāAŽçŽDērīijNā;āāfĒēāzā;fçTlāy'd'āyIāŔ■æŪIJæIāīijNçšžāijij  
'(\d+)/(\d+)/(\d+)' āĀĆ

ēIJĀēēAæšlæĎŔçŽDæYŔ match() æŪžæšTāžĒāžĒæčĀæšēā■ŪçņēāyšçŽDāijĀāgNēČlāLĒāĀĆāōČçŽ

```
>>> m = datepat.match('11/27/2012abcdef')
>>> m
<_sre.SRE_Match object at 0x1005d27e8>
>>> m.group()
'11/27/2012'
>>>
```

āēČædIJā;āæČšçšĭçāōāNžēĒ■īijNçāōāfIā;āçŽDæ■čāLŽēālēĭāijRāžēšçžšārĭijNāršāČŔēfZāžLēfZæāū

```
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)$')
>>> datepat.match('11/27/2012abcdef')
>>> datepat.match('11/27/2012')
<_sre.SRE_Match object at 0x1005d2750>
>>>
```

æIJĀāŔŌīijNāēČædIJā;āāžĒāžĒæYŔāAŽāyĀæñāçōĀā■TçŽDæŪGæIJñāNžēĒ■/æŔIJçt'cæš■ā;IJçŽDērī  
re ælāāIŪçžgāLŃçŽDāG;æTŕāijŽārEæIJĀēfŠçijŪērSēfGçŽDæIāāijRçijSā■YētuāIēīijNāZāæ■d'āžūāy■āijZæŪl  
ā;EæYŔāēČædIJā;fçTlēcĎçijŪērSæIāāijRçŽDērīijNā;āārEāijZāGRārSæšēāLĭāšNāyĀāžZēcIād'ŪçŽDād'Ď

```
>>> re.findall(r'(\d+)/(\d+)/(\d+)', text)
[('11', '27', '2012'), ('3', '13', '2013')]
>>>
```

ā;EæYŔēIJĀēēAæšlæĎŔçŽDæYŔīijNāēČædIJā;āæL'ŠçōŪāAŽād'gēGRçŽDāNžēĒ■āšNæŔIJçt'cæš■ā;IJ  
æIāāIŪçžgāLŃçŽDāG;æTŕāijŽārEæIJĀēfŠçijŪērSēfGçŽDæIāāijRçijSā■YētuāIēīijNāZāæ■d'āžūāy■āijZæŪl  
ā;EæYŔāēČædIJā;fçTlēcĎçijŪērSæIāāijRçŽDērīijNā;āārEāijZāGRārSæšēāLĭāšNāyĀāžZēcIād'ŪçŽDād'Ď

## 4.5 2.5 ā■ŪçņēāyšæŔIJçt'cāšNæŽĒæ■č

### ēŪōēcŸ

ā;āæČšāIJlā■Ūçņēāyšāy■æŔIJçt'cāšNāNžēĒ■æNĠāōŽçŽDæŪGæIJñæIāāijR

## èġċăĖşăŮzăăĹ

ărzăžŎçŏĂă■ŤçŽĐă■ŮéĬăĹăĭjRĭijŇçŽŤ æŎëăĭçŤĬ str.replace()  
æŮzăşŤă■şăŖĭijŇăŕŤăĕĆĭijŽ

```
>>> text = 'yeah, but no, but yeah, but no, but yeah'
>>> text.replace('yeah', 'yep')
'yep, but no, but yep, but no, but yep'
>>>
```

ărzăžŎăđ'■ăĬçŽĐăĹăĭjRĭijŇăŕŮăĭçŤĬ re æĹăăĬŮăŷ■çŽĐ sub()  
ăĢĭæŤŕăĂĆ äŷžăžĖĕŕŤ æŸŎëĤŽăŷĭijŇăĂĢĕŏĭăĭăăÇşăŕĖăĭăĭjRăŷž 11/27/2012  
çŽĐăŮăĖĬşăă■ŮçĭăŷşăŤzăĹŖ 2012-11-27 âĂĆçđ'žăĭŇăĕCăŷŇĭijŽ

```
>>> text = 'Today is 11/27/2012. PyCon starts 3/13/2013.'
>>> import re
>>> re.sub(r'(\d+)/(\d+)/(\d+)', r'\3-\1-\2', text)
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>>
```

sub() ăĢĭæŤŕăŷ■çŽĐçĭăŷĂăŷĹăŖĆăŤŕăŸŕăĕăŇăŇzéĖ■çŽĐăĹăĭjRĭijŇçĭăŷŇăŷĹăŖĆăŤŕăŸŕăZăĖ  
\3 æŇĢăŖŖşăĹ■ăĬăĹăĭjRçŽĐă■ŤăŎŮçžĐăŖŮăĂĆ

ăĕĆăđĬăĭăăĹŖşçŏŮçŤĬçŽŷăŖŇçŽĐăĹăĭjRăĂžăđ'ŽăŇăăZăă■ćĭijŇăĂĆăZşăĖĹćĭjŮăŕŖşăŏCăĬăăŖŖăŖ

```
>>> import re
>>> datepat = re.compile(r'(\d+)/(\d+)/(\d+)')
>>> datepat.sub(r'\3-\1-\2', text)
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>>
```

ărzăžŎăŽŤăĹăăđ'■ăĬçŽĐăZăă■ćĭijŇăŖŕăžăăĭăăĖĂşăŷĂăŷĹăZăă■ăZăđăŕCăĢĭæŤŕăĬăăžăăZăĭijŇăŕŖ

```
>>> from calendar import month_abbr
>>> def change_date(m):
...     mon_name = month_abbr[int(m.group(1))]
...     return '{} {} {}'.format(m.group(2), mon_name, m.group(3))
...
>>> datepat.sub(change_date, text)
'Today is 27 Nov 2012. PyCon starts 13 Mar 2013.'
>>>
```

ăŷĂăŷĹăZăă■ăZăđăŕCăĢĭæŤŕçŽĐăŖĆăŤŕăŸŕăŷĂăŷĹ match ărzăşăĭjŇăžşăŕşăŸŕ  
match() æĹŮăĂĖ find() ĕĤăZăçŽĐăŕzăşăăĂĆ äĭçŤĬ group()  
æŮzăşŤăĬăŖŖăŖŮŮçĹ'žăŏŽçŽĐăŇzéĖ■ăĬăĹăĂĆăZăđăŕCăĢĭæŤŕăĬăăŖŎăĖŤăZăđăZăă■ăŮçĭăŷşăĂ

ăĕĆăđĬăĖŽđ'ăžĖăăZăă■ăŖŎçŽĐçžşăđĬăđ'ŮĭijŇăĭăĖŸăăÇşşăăĂşăĬĹăđ'ŽăŕŖşăZăă■ăŖŖşçŤşăžĖŖ  
re.subn() æĬăăžăăZăăĂĆăŕŤăĕĆĭijŽ

```
>>> newtext, n = datepat.subn(r'\3-\1-\2', text)
>>> newtext
```

```
'Today is 2012-11-27. PyCon starts 2013-3-13.'
>>> n
2
>>>
```

## èõìèõž

ãĖšăžŎæ■čăĹŽèăĹèĹĹăĭjRăŘIJčť cáŠNăẂẂæ■ćĭjNăyĹéĹćăĭjTčď'žčŽĎ sub()  
æŮžăşŤăşžăIJnăũşçzRăűţçŽŮăžĖăĹ'ĂăIJĹ'ăĂĆ âĖŭăőđăIJĂéŽĹčŽĎĎČĹăĹĖăřsăĖŤřĭjŮăĖŽăæ■čăĹŽèăĹèĹĹă

## 4.6 2.6 â■ŮçņęäÿšăĖĭçŤěăď'ğăŖăĖŽčŽĎæŘIJčť'ćăẂẂæ■ć

### éŮőéćŸ

ăĭăĖIJĂĕĖAăžăăĖĭçŤěăď'ğăŖăĖŽčŽĎæŮžăĭjRăŘIJčť'ćăŷŎăẂẂæ■ćăŮĖăIJnă■Ůçņęäÿš

## èğčăĖşşăŮžăăĹ

ăÿžăžĖăIJĹăŮĖăIJnăş■ăĭjJăŮŭăăĭçŤěăď'ğăŖăĖŽĭjNăĭăĖIJĂĕĖAăIJĹăĭçŤĭ  
re áĹăăĹŮčŽĎæŮŭăĂŽčžŽĖĖŽăžŽăş■ăĭjJăĖŘăĹŽ re.IGNORECASE  
ăăĖăĖŮăŮăŖĆăŤŕăĂĆăŖŤăĖĆĭjŽ

```
>>> text = 'UPPER PYTHON, lower python, Mixed Python'
>>> re.findall('python', text, flags=re.IGNORECASE)
['PYTHON', 'python', 'Python']
>>> re.sub('python', 'snake', text, flags=re.IGNORECASE)
'UPPER snake, lower snake, Mixed snake'
>>>
```

ăIJĂăŖŎčŽĎĎČăÿĹăĹNă■RăĖ■ćď'žăžĖăÿĂăÿĹăŖĖĭjžĖŽŮĭjNăẂẂæ■ćă■Ůçņęäÿšăžăŭăÿ■ăĭjŽĖĖĹăĹĹăŭş  
ăÿžăžĖăĖăŮăď'■ĖĖŽăÿĭjNăĭăăŖĖĎČĭĖIJĂĕĖAăÿĂăÿĹăĹĖăĹ'ăĖĖăŤĭjNăŖşăĎŖăÿNéĹćçŽĎĎĖŽăăŭĭjŽ

```
def matchcase(word):
    def replace(m):
        text = m.group()
        if text.isupper():
            return word.upper()
        elif text.islower():
            return word.lower()
        elif text[0].isupper():
            return word.capitalize()
        else:
            return word
    return replace
```

ăÿNéĹćăŤŕăĭçŤĹăÿĹĖĖŖăĖĖăŤŖçŽĎæŮžăşŤĭjŽ

```
>>> re.sub('python', matchcase('snake'), text, flags=re.IGNORECASE)
'UPPER SNAKE, lower snake, Mixed Snake'
>>>
```

erSèĀĔæšlrijŽ matchcase('snake') èĤTāZđāžEäyĀäyĭāZđērČāĠ;æTŕ(āRCæTŕāĤĒéazæYŕ  
match ārzēsā)ijNāL■ēlčāyĀĔŁCæRŘāĬrēĠġijN sub()  
āĠ;æTŕēZd'āžEæŌēāRŪæZĤæ■čā■Ūçņēāyšād'ŪijNēĤYēČ;æŌēāRŪāyĀäyĭāZđērČāĠ;æTŕāĀĆ

## ēōlēōž

āržāžŌāyĀĔĤŇčŽĐāĤ;çTēād'gārRāĤZçŽĐāNzéĒ■æŠ■ā;IġijNčōĀā■TçŽĐāijāēĀŠāyĀäyĭ  
re.IGNORECASE æāĠāĤŪāRCæTŕārsāušçzRēūšād'šāžEāĀĆ  
ā;EæYŕēIJāēçAæšĭæĎRçŽĐæYŕijNēĤZāyĭāržāžŌæšRāžŽēIJāēçAād'gārRāĤZē;ĭnæ■čçŽĐUnicodeāNzéĒ■ā  
āRCēĀĆ2.10ārRēĤCāžEēğçæŽt'ād'ŽçzEēĤCāĀĆ

## 4.7 2.7 æĬĀçš■āNzéĒ■æĭāijR

### éŪōécY

ā;āæ■čāĬĤērTçĬĀçTĭæ■čāĤZēāĤē;ĭāijRāNzéĒ■æšRāyĭæŪĠæIJāĭāijRijNā;EæYŕāōCæĤ;ĭāĤçŽĐæY  
ēĀNā;āæČšāĤōæTžāōCāRŶæĬRæšēæĤ;ĭæĬĀçš■çŽĐāRēČ;āNzéĒ■āĀĆ

### ēğçāEşæŪzæāĤ

èĤZāyĭēŪōécYāyĀĔĤŇčŽĐāĤ;çTēād'gārRāĤZçŽĐāNzéĒ■āyĀāržāĤEēŽTçņēāžNēŪt'çŽĐæŪĠæIJāçŽĐæŪāĀ  
āyžāžEēŕt'æYŌæyĒæēŽijNēĀĆēŽSāçCāyNçŽĐā;Nā■RijŽ

```
>>> str_pat = re.compile(r'\"(.*)\"')
>>> text1 = 'Computer says "no."'
>>> str_pat.findall(text1)
['no.']
>>> text2 = 'Computer says "no." Phone says "yes."'
>>> str_pat.findall(text2)
['no." Phone says "yes.']
>>>
```

āĬĤēĤZāyĭā;Nā■Rāy■ijNāĭāijR r'\"(.\*)\"' çŽĐæĎRāZ;æYŕāNzéĒ■ēčāRŇāijTāRŪāNĒāRŇçŽ  
ā;EæYŕāĬĤæ■čāĤZēāĤē;ĭāijRāy■\*æš■ā;IçņēæYŕēt'ĭāĤçŽĐijNāZāæ■d'āNzéĒ■æš■ā;IāijZæšēæĤ;ĭæĬĀēT  
āžŌæYŕāĬĤçŇāžNāyĭā;Nā■Rāy■æRIJçt'ç text2 çŽĐæŪāĀZēĤTāZđçzSæđIJāzūāy■æYŕæĬSāžnæČšēçAç

āyžāžEāĤōæ■çēĤZāyĭēŪōécYijNāRŕāzēāĬĤāĭāijRāy■çŽĐ\*æš■ā;IçņēāRŌēĬcāĤāyĤ?āĤōēēŕçņēijNā

```
>>> str_pat = re.compile(r'\"(.*)?\"')
>>> str_pat.findall(text2)
['no.', 'yes.']
>>>
```



èŁŻæăăăřsä;Łă;ŮăŇzéĚ■ăŔŸæĹŔéĬđèťlăl'łăłăăijŔiijŇăžŎèĂŇă;ŮăĹŕæĬĂç§■çŽĐăŇzéĚ■iijŇăž§ăřsă

## èőĹèőž

èŁŻăŸĂèĹĆăŝŤçđ'žăžĚăĬĬăĚŽăŇĚăŔŇćĆž(.)ă■ŮçŇçŽĐæ■čăĹŽèăĹè;ăijŔçŽĐæŮăăĂŽéĂĜăĹŕçŽĐă  
ăĬĬăŸĂăŸłăłăăijŔă■ŮçŇăŸŸăŸ■iijŇćĆž(.)ăŇzéĚ■éŽđ'ăžĚæ■céăŇăđ'ŮçŽĐăžžă;Ťă■ŮçŇăĂĆ  
çĐđéĂŇiijŇăéĆăđĬă;ăăŕĚçĆž(.)ăŔăŮăŤ;ăĬĬăijĂăĝŇăŸŎçžŖăĬŝçŇç(æŕŤăçĆăijŤăŔăŮ)ăžŇéŮť'çŽĐæŮăăĂŽ  
èŁŻæăăéĂŽăŸŸăijŽăŕijèĜť'ă;Ĺăđ'ŽăŸ■éŮť'çŽĐèćăijĂăĝŇăŸŎçžŖăĬŝçŇçăŇĚăŔŇćŽĐæŮĜăĬĬŇèćăł;çŤă  
éĂŽèŁĜăĬĬ \* æĹŮèĂĚ + èŁŻæăăçŽĐæŖă;ĬçŇçăŔŎéĬćăăžăĹăăŸĂăŸł ?  
ăŔŕăžèăijžăĹăăŇzéĚ■čŮăçŤăŤžæĹŔăŕžæĹ;æĬĂç§■çŽĐăŔŕèČ;ăŇzéĚ■ăĂĆ

## 4.8 2.8 äđ'ŽèăŇăŇzéĚ■ăłăăijŔ

### éŮőéćŸ

ă;ăæ■čăĬĬŕŕŤçĬĂă;ŁçŤłæ■čăĹŽèăĹè;ăijŔăŎžăŇzéĚ■ăŸĂăđ'ĝăĬŮçŽĐæŮĜăĬĬiijŇèĂŇă;ăéĬĂèçĂèł

## èĝčăĚŖæŮžæăĹ

èŁŻăŸłéŮőéćŸă;ĹăĚŸăđŇçŽĐăĜžçŎŕăĬĬă;Ŗă;ăçŤĬćĆž(.)ăŎžăŇzéĚ■ăžžăđŔăăŮçŇçŽĐæŮăăĂŽiijŇă  
æŕŤăçĆiijŇăĂĜèđ;ă;ăæČŖŕŤçĬĂăŎžăŇzéĚ■ĆŕŕēĬĂăĹĚăĹŝçŽĐăŖłéĜĬiijŽ

```
>>> comment = re.compile(r'\/*(.??)\/')
>>> text1 = '/* this is a comment */'
>>> text2 = '''/* this is a
... multiline comment */
... '''
>>>
>>> comment.findall(text1)
[' this is a comment ']
>>> comment.findall(text2)
[]
>>>
```

ăŸžăžĚăĹŮă■çèŁŻăŸłéŮőéćŸiijŇă;ăăŔŕăžèăĹŮăŤžăłăăijŔă■ŮçŇçăŸŸiijŇăćđăĹăăŕžæ■céăŇçŽĐæŤŕæŇ

```
>>> comment = re.compile(r'\/*((?:.|\\n)*?)\/')
>>> comment.findall(text2)
[' this is a\n multiline comment ']
>>>
```

ăĬĬèŁŻăŸłăłăăijŔăŸ■iijŇ (?:.|\\n) æŇĜăđžăžĚăŸĂăŸłéĬă■ŤèŎűçžĐ  
(ăžŖăřsăŸŕăđĆăđŽăžĹăžĚăŸĂăŸłăžĚăžĚçŤłæĬăĂŽăŇzéĚ■iijŇèĂŇăŸ■č;éĂŽèŁĜăŤçŇŇăæ■ŤèŎűæĹŮèĂ

```
re.compile()      åĜ;æŦræÕěåRÛäYÄäylæāĞǻUåRĆæŦřăŔń      re.DOTALL
iiĵÑãIjleēZēĠÑēIdāyȳæIJL'çŦlāĂĈ āōČăŦřăzēēōſ æ■cālZēalēι;aiĵRäy■čŽĐćCz(.)(.āNžĚ■āNĚæNňæ■cēaN
```

[illegible]

éŮőécŸ

èġčǎẸșæŮźæąŁ

ẽŁŻĖŃçŽĐæŮĜæIJñâĀİSpicy JalapeÃsoãĀİä;ŁçŦlăžEäyđ'çĝ■ă;čajRæiẽəłçđ'žăĂĈ  
 çññăŷĂçĝ■ă;ŁçŦlăžŦt'ă;Şă■ŮçñăĀİĂşăĀİ(U+00F1)ııjŃçññăžŃçĝ■ă;ŁçŦlăNL'ăŷAă■Ůăŕă■ăĀİnăĀİăRŌéİcă

```
>>> import unicodedata
>>> t1 = unicodedata.normalize('NFC', s1)
>>> t2 = unicodedata.normalize('NFC', s2)
>>> t1 == t2
```

normalize() ċñňăŷĂăŷłăŔĈĊēȚŕăĤŇĞăőŽă■ŬćņęăŷşăăĞăĢĖăŃŨćŻĐăŮźăįŔăĂĈ  
NFCëàłçđ'žă■ŬćņęăżȚërěăÝŕăȚŕ'ä;ŞçzDăĹŔ(æŦăēĈăŔŕěĈ;çŻĐērłăŕśă;£ĈŦłă■ȚăŷĂçįŮćăA)ııjÑěĂŅŅNFİ  
PythonăŔŇăăüăȚŕăŇĂăĹŦ'ăsȚćŻĐăăĞăĢĖăŃŮă;ćăįŔŅFKCăŠŅŅFKDııjŅăőĈăżŋăIJlăđ'ĐĉŘĖăşĔ

èóìèőž

```
>>> t1 = unicodedata.normalize('NFD', s1)
>>> ''.join(c for c in t1 if not unicodedata.combining(c))
'Spicy Jalapeno'
>>>
```

æIJĀāRŌäyÄäyĭā;Ŋā■RāſTçd'zāzE unicodedata æĭaĭU̇çŽDāRēäyÄäyĭēG■ēAæŪzéĭcĭijŊāzšārsæ  
combining() āG;æTřāRřāzēæŋNērTäyÄäyĭā■U̇çņæYřāRēäyžāſNēššā■U̇çņāĀĆ  
āIJĭēZāyĭæĭaĭUāy■ēfYæIJĭāĒūāzŪāG;æTřçTĭāžŌæšēāL;ā■U̇çņçšzāĽñĭijŊæŋNērTæYřāRēäyžæTřā■Ūā  
UnicodeæY;çDūæYřäyÄäyĭā;Ľāđ'gçŽDäyžécYāĀĆāēĆāđIJæČšæŽt'æūsāĒēçŽDāžEēgčāĒšāžŌæāGāČ  
ērūçIJNēĀĆ UnicodeāōYç;Sāy■āĒšāžŌēfZēČĭāĽEçŽDēřt'æYŌ  
Ned BatchelderāIJĭ āzŪçŽDç;ſçñŽ äyĽāřZPythonçŽDUni-  
codeāđ'DçRĒēŪōēçYāzšæIJĭäyÄäyĭā;Ľāē;çŽDāžNçz■āĀĆ

æuũaŔĹä;ƒçŦÍUnicodeaŠŇæ■čāĹZèaĭè;āijRéĀŽāyŷäijŽèōŕ'ä;äæŁŞçŇĆāĀĆ  
 æĈĈæđIJā;äçIJşçŽĎæŁ'ŞçõŬèĚZæuũaAŹçŽĎŕİirijŇæIJĀæ;èĀĈèŽŚayŇāōŁ'èĉĖçñāyŁ'æŬzæ■čāĹZāijŔāzŞ  
 aōĈCāznāijŽāyžUnicodeçŽĎād'gārŔaEŽè;Ňæ■čāŠŇāEūāzŬād'gēGRæIJŁ'èuĉcŁ'žæĀgæŔŔä;ŽāĖĭēĭcçŽĎæŦŕ

## 4.11 2.11 aLăéZd'âUçņęäyşăy■ăy■élJĂèeAçŻDă■Uçņę

### éUóécŸ

äjäæČšăŎžæŎL'æŮĜæIJňă■UçņęäyşăijĂăd't'ijŇçzŞărĭæLŮèĂĚäy■éŮt'äy■æČşęeAçŻDă■UçņęijŇăer

### èğcâEşşæŮzæqĹ

strip() æŮzæşŦèČĭçŦlăžŎăLăéZd'ăijĂăğŇæLŮçzŞărĭçŻDă■UçņęăĂĆ  
rstrip() âŠŇ rstrip() âĹEăĹnăžŎăuęăŠŇăžŎăRşæL'ğęăŇăLăéZd'æŞ■ăĭIJăĂĆ  
ézŸeôd'æČĚăEęăyŇijŇeĚZăžZæŮzæşŦăijŽăŎžéZd'çl'žçŽĭă■UçņęijŇăĭEæŸřăĭăăžşăRřăžæŇĜăŏŽăĚüăžŮ

```
>>> # Whitespace stripping
>>> s = ' hello world \n'
>>> s.strip()
'hello world'
>>> s.lstrip()
'hello world \n'
>>> s.rstrip()
' hello world'
>>>
>>> # Character stripping
>>> t = '----hello===='
>>> t.lstrip('-')
'hello===='
>>> t.strip('--')
'hello'
>>>
```

### èóíèöž

ēĚZăžZ strip() æŮzæşŦăIJlérzărŮăŠŇăyĚşŖEæŦřæ■ôăžăd'ĜăRŎçz■ăd'ĐçŖEçŻDæŮüăĂZæŸřę  
ærŦăeĆĭijŇăĭăăRřăžęçŦlăŏČăžňæĭăŎžæŎL'çl'žæăijŇijŇăijŦăRŮăŠŇăŏŇæĹŖăĚüăžŮăăžăăĹăăĂĆ

ăĭEæŸřéIJăèeAęşăđŖçŻDæŸřăŎžéZd'æŞ■ăĭIJăy■ăijŽăřză■UçņęäyşçŻDăy■éŮt'çŻDæŮĜæIJňăžğçŦ

```
>>> s = ' hello      world \n'
>>> s = s.strip()
>>> s
'hello      world'
>>>
```

ăęĆădIJăĭăæČşăd'ĐçŖEęăy■éŮt'çŻDçl'žæăijŇijŇéĆăžĹăĭăéIJăèeAęşĆăĹ'ăĚüăžŮæĹĂăIJřăĂĆærŦăę  
replace() æŮzæşŦæLŮèĂĚæŸřęŦlă■căĹZęăĹęĭăĭăijRæZĚæ■căĂĆçd'žăĹŇăęCăyŇijŽ

```
>>> s.replace(' ', '')
'helloworld'
>>> import re
```

```
>>> re.sub('\s+', ' ', s)
'hello world'
>>>
```

éĀŽāyŷæĈĒĀĒġāyŊā;ăæĈşārĒā■Ūĉņēāyŝ strip æŞ■ă;IJăŞŊăĒŪāzŪēĤ■āzĉæŞ■ă;IJçŽŷçzŞăŔĹijŊæŕ  
 æĈĀđIJæŶŕēĤZæăŭçŽĎĕŕĹijŊēĈcāzĹĈŤşæĹŔăŽĹēāĹēĹăijŔăŕşăŔŕăzēăđ'ğæŶŷēznæĹŊăžĒăĀĈæŕŤæĈĹijŽ

```
with open(filename) as f:
    lines = (line.strip() for line in f)
    for line in lines:
        print(line)
```

ăIJĹēĤZēĠŊijŊēāĹēĹăijŔ lines = (line.strip() for line in f)  
 æĹġēāŊæŤŕæ■ōē;ŋæ■ĉæŞ■ă;IJăĀĈ ēĤŽçġ■æŪzăijŔēĪđāyŷēŊŶæŤĹijŊăZăāyžăōĈāy■ēIJăēēĀēĈĎăĒĹŕzăĀ  
 āōĈāzĒāzĒăŔŕæŶŕăĹZăāzăyĀăyĹĈŤşæĹŔăŽĹijŊăžŭāyŤæŕŔæŋæēĤăZđēāŊăžŊăĹ■ăijŽăĒĹæĹġēāŊ  
 strip æŞ■ă;IJăĀĈ

ărzāžŌæŽŕ'ēŊŶēŶŪçŽĎstripĹijŊā;ăăŔŕēĈŷēIJăēēĀă;ĤĈŤĹ translate()  
 æŪzæşŤăĀĈēŕăŔĈēŶĒăyŊăyĀēĹĈāzĒēġĉæŽŕ'ăđ'ŽăĒşăžŌă■ŪĉņēāyŝæyĒçŔĒçŽĎăĒĒăōzăĀĈ

## 4.12 2.12 āōāæşşæyĒçŔĒæŪĠæIJăăŪĉņēāyŝ

### éŪōēĈŶ

ăyĀăžZæŪăēĀĹĈŽĎăzijĹĹZēzŞăōĉăĹIJă;ăçŽĎç;ŞĉŊZēāŧēĹĉēāĹă■Ťāy■ēĹŞăĒēæŪĠæIJăăĀipĀ;tĀēĀŭĀŝ

### ēġĉăĒşæŪzæāĹ

æŪĠæIJăăyĒçŔĒēŪōēĈŶăijŽæŭĹăŔĹăĹŕăŊĒæŊŋæŪĠæIJăăēġĉăđŔăyŌæŤŕæ■ōăđ'ĎçŔĒç■ĹăyĀçşză  
 āIJĹēĪđāyŷçōĀă■ŤçŽĎæĈĒă;ĉăyŊŊijŊā;ăăŔŕēĈŷēIJăēēĀēĈĎăĒĹŕzăĀ  
 str.upper() āŞŊ str.lower() )ărĒæŪĠæIJăă;ŋăyžăăĠăĠĒæăijăijŔăĀĈ ä;ĤĈŤĹ  
 str.replace() æĹŪēĀĒ re.sub() çŽĎçōĀă■ŤæZĒæ■ĉæŞ■ă;IJēĈŷăĹăēZđ'æĹŪēĀĒæŤzăŔŶæŊĠăō  
 ä;ăăŔŊæăŭēĤŶăŔŕăzēă;ĤĈŤĹ2.9ărŔēĹĈçŽĎ unicodedata.normalize()  
 āĠæŤŕăŕĒunicodæŪĠæIJăăăĠăĠĒæŊŭăĀĈ

çĎŭăŔŌĹijŊæIJĹæŪŭăĀZă;ăăŔŕēĈŷēŶæĈşăIJăyĒçŔĒæŞ■ă;IJăyĹæŽŕ'ēĤZăyĀæ■ēăĀĈæŕŤæĈĹijŊă  
 äyžăžĒēĤZæăŭăĀŽĹijŊă;ăăŔŕăzēă;ĤĈŤĹçzŔăyŷăijŽēĉŋăŷ;ēġĒçŽĎ str.translate()  
 æŪzæşŤăĀĈ äyžăžĒæijŤĈđ'žĹijŊăĀĠēōĹă;ăçŌŕăIJăIJĹăyŊēĹĉēĤZăyĹăĠŊăžşçŽĎă■ŪĉņēāyŝĹijŽ

```
>>> s = 'pÃ;tÄëÃŭÃŝ\u0000fis\tawesome\r\n'
>>> s
'pÃ;tÄëÃŭÃŝ\u0000cis\tawesome\r\n'
>>>
```

çŋŋăyĀæ■ēæŶŕæyĒçŔĒçĹ'žçŽă■ŪĉņēăĀĈăyžăžĒēĤZæăŭăĀŽĹijŊăĒĹăĹZăāzăyĀăyĹăŔŔçŽĎē;ŋæ■ĉēāĹ  
 translate() æŪzæşŤĹijŽ

```
>>> remap = {
...     ord('\t') : ' ',
...     ord('\f') : ' ',
...     ord('\r') : None # Deleted
... }
>>> a = s.translate(remap)
>>> a
'pÃ;tÄëÃüÃś is awesome\n'
>>>
```

æ■čæĆä;äçIJŇçŽĐéĆčæüüijŇçl'žçŽ;ā■Ůçñē \t āŠŇ \f  
 āũščzŔècñéĜ■æŮræŸāārĐāĽrāyÄäyĽçl'žæäijāĂĆāŽđē;çā■ŮçñerçŽt' æŎëècñāĽăéŽd' āĂĆ  
 ä;āāŔřäzëäzëèēŹäyĽēāĽæäijäyžāšžçāĂēŹäyĂæ■ēädĐāžžæŽt' ād' ġçŽĐēāĽæäijāĂĆærŤæĆriijŇèol' æĽŚā

```
>>> import unicodedata
>>> import sys
>>> cmb_chrs = dict.fromkeys(c for c in range(sys.maxunicode)
...                          if unicodedata.combining(chr(c)))
...
>>> b = unicodedata.normalize('NFD', a)
>>> b
'pÃ;tÄëÃüÃś is awesome\n'
>>> b.translate(cmb_chrs)
'python is awesome\n'
>>>
```

äyĽéĽcä;Ňā■Ŕäy■riijŇéĂŽēĽĜä;ĽçŤĭ dict.fromkeys()  
 æŮzæšŤädĐéĂäyÄäyĽā■ŮäËyüijŇærŔäyŮUnicodeāŠŇéšçñēä;IJäyžēŤōriijŇāržāžŤçŽĐāĀijāĒĽéĆĽäyž  
 None āĂĆ

çĐúāŔŎä;ĽçŤĭ unicodedata.normalize() āŕĒāŎšāġŇē;ŠāĒēæāĜāĜĒāŇŮäyžāĽĒēġçā;çäijŔā■  
 çĐúāŔŎāĒē■ērČçŤĭ translate āĜ;æŤŕāĽăéŽd' æĽ'ĂæIJĽ'éĜ■éšçñēāĂĆ  
 āŔŇæäüçŽĐæĽĂæIJřäžšāŔřäzëècñçŤĭæĽēāĽăéŽd' āĒüāzŮçšžāđŇçŽĐā■Ůçñē(ærŤæĆæŎġāĽūā■Ůçñēç■Ľ)ā  
 ä;IJäyžāŔēäyÄäyĽä;Ňā■ŔriijŇēŹéĜŇädĐéĂäyÄäyĽārĒæĽ'ĂæIJĽUnicodeæŤŕā■Ůā■ŮçñēæŸāārĐāĽ

```
>>> digitmap = { c: ord('0') + unicodedata.digit(chr(c))
...             for c in range(sys.maxunicode)
...             if unicodedata.category(chr(c)) == 'Nd' }
...
>>> len(digitmap)
460
>>> # Arabic digits
>>> x = '\u0661\u0662\u0663'
>>> x.translate(digitmap)
'123'
>>>
```

āŔēäyĂçġ■æyĒçŔĒæŮĜæIJŇçŽĐæĽĂæIJřæŮĽ'āŔĽāĽŕĭ/OēġççāĀäyŎçijŮçāĀāĜ;æŤŕāĂĆēŹéĜŇçŽĐ  
 çĐúāŔŎāĒē■çzŠāŔĽ encode() æĽŮëĂĒ decode() æŠ■ä;IJæĽēæyĒéŽd' æĽŮäſōæŤžāōČāĂĆærŤæĆriijŽ

```
>>> a
'pÃ;tÄëÃüÃs is awesome\n'
>>> b = unicodedata.normalize('NFD', a)
>>> b.encode('ascii', 'ignore').decode('ascii')
'python is awesome\n'
>>>
```

èŁÉĜŃŻĐæĀĜĖĀŅŮæŠ■ā;IJāŖĒāŌſæİēçŽĐæŮĜæIJñĀŁĒēğçäyžā■TçNñçŽĐāŠŅéſſçņēāĀĆæŌē.  
ā;ſçĐŮīījŅēŁŻçğ■æŮžæſTāžĒāžĒāŖĀIJĀIJĀāŖŌçŽĐçŽōæāĜāŖſæŸŖēŌūāŖŮāŁŖæŮĜæIJñāŖžāžŤACSIIēā

## èőİèőž

æŮĜæIJñā■ŮçņæyŸĖŘĒäyÄäyĀæIJÄäyžèēAçŽĐēŮōēçŸāžŤēŖēæŸŖēŁŖēāŃçŽĐæĀĝēČ;āĀČäyÄēŁñæ  
āržāžŌçōĀā■TçŽĐæŽŁæ■ćæſ■ā;IJīījŅ str.replace() æŮžæſTēĀŽāyŸæŸŖæIJĀāŁŃçŽĐīījŃçŤŽēĜſāIJ  
æŖŤæČīījŃäyžāžĒæyŸĖŘĒēçŁ'žçŽ;ā■ŮçņēīījŃä;āāŖŖāžèēŁŽæūāAŽīījŽ

```
def clean_spaces(s):
    s = s.replace('\r', '')
    s = s.replace('\t', ' ')
    s = s.replace('\f', ' ')
    return s
```

æĖČæđIJā;āāŌžæŤŃērTçŽĐērīījŃä;āāŖſāījŽāŖſçŌŖēŁŻçğ■æŮžāījŖāījŽæŖŤä;ŁçŤĪ  
translate() æŁŮēĀĒæ■čĀŁŽēāİē;āījŖēēAāŁŃā;Łād'ŽāĀĆ

āŖēäyÄæŮžēİēīījŃæĖČæđIJā;æēIJĀēēAæŁ'ĝēāŃāžžā;Ťād'■æİČā■Ůçņæāržā■ŮçņççŽĐēĜ■æŮŖæŸāārĐæ  
tanslate() æŮžæſTāījŽēİdāyŸçŽĐāŁŃāĀĆ

āžŌād'ğçŽĐæŮžēİēāİēēōſīījŃāŖžāžŌā;āçŽĐāžŤçŤĪçĪŃāžŖæİēēŖ'æĀĝēČ;æŸŖä;āäy■ā;Ůäy■āŌžēĜĀū.  
äy■āžyçŽĐæŸīījŃæŁſāžŃäy■āŖŖēČ;çžŽā;āāžžēōōäyÄäyŁçŁ'žāōŽçŽĐæŁĀæIJŖīījŃä;ŁāōČēČ;ād'ſēĀČāžŤæ  
āŽāæ■d'āōđēŽĒæČĒāĒŤäy■ēIJĀēēAā;æēĜĀūſāŌžārĪērŤäy■āŖŃçŽĐæŮžæſTāžūērĐāījŖāōČāĀĆ

ār;çōāēŁŽäyĀēŁČēZEäy■èőİèőžçŽĐæŸŖæŮĜæIJñīījŃä;ĒæŸŖçſžāīījçŽĐæŁĀæIJŖāžſāŖŖāžēēĀĆçŤĪāž

## 4.13 2.13 ā■Ůçņæyſāržē;Ŗ

### éŮōēçŸ

ā;āæČſēĀŽēŁĜæſŖçğ■āržē;ŖæŮžāījŖæİēæāījāījŖāŅŮā■Ůçņæyſ

### ēğçĀĒſæŮžæāŁ

āržāžŌāſžæIJñçŽĐā■Ůçņæyſāržē;Ŗæſ■ā;IJīījŃāŖŖāžēä;ŁçŤĪā■ŮçņæyſççŽĐ ljust()  
,rjust() āŠŃ center() æŮžæſTāĀČæŖŤæČīījŽ

```
>>> text = 'Hello World'
>>> text.ljust(20)
```



```
'Hello World'
>>> text.rjust(20)
'          Hello World'
>>> text.center(20)
'    Hello World    '
>>>
```

æL'ĂæIJL'èŁZăžZæŮzæŝTéČ;èČ;æŎěăRŮăyĂăyĹăRréĂL'čŽDăăăăĚĚă■ŮčņăĂĂCærŤăeĆrijŽ

```
>>> text.rjust(20, '=')
'=====Hello World'
>>> text.center(20, '*')
'****Hello World*****'
>>>
```

ăĜ;æŤř format() ăRŇæăăăăRřăžēcŤĹăĹăăĹăăŏzæŸŝčŽDărzé;Řă■ŮčņăyŝăĂĆ  
ă;ăđeAăĂZčŽDărŝæŸřă;ŁčŤĹ<, > æĹŮěĂĚ ^ ă■ŮčņăăRŎěĹćŧ'ğeŭŝăyĂăyĹăŇĜăăŏŽčŽDăăŏ;ăžeăĂĂCærŤăeĆ

```
>>> format(text, '>20')
'          Hello World'
>>> format(text, '<20')
'Hello World          '
>>> format(text, '^20')
'    Hello World    '
>>>
```

ăeĆăedIJă;ăeČŝæŇĜăăŏŽăyĂăyĹăĹđčŁ'žæăijčŽDăăăăĚĚă■ŮčņărijŇăřĚăăŏČăĚŽăĹăřăřzé;Řă■ŮčņăčŽDăăL■

```
>>> format(text, '=>20s')
'=====Hello World'
>>> format(text, '*^20s')
'****Hello World*****'
>>>
```

ă;ŝæăijăijŘăŇŮăđ'ŽăyĹăĂijčŽDăŮăăĂŽijŇeŁZăžZæăijăijRăžččăĂăžŝăRřăžěećŋčŤĹăIJĹ  
format() æŮzæŝŤăy■ăĂCærŤăeĆrijŽ

```
>>> '{:>10s} {:>10s}'.format('Hello', 'World')
'          Hello          World'
>>>
```

format() ăĜ;æŤřčŽDăyĂăyĹăe;ăđ'DăŸăăŏČăy■ăžĚěĂĆčŤĹăžŎă■ŮčņăyŝăĂĆăăŏČăRřăžēcŤĹăĹăăăij  
ærŤăeĆrijŇă;ăăRřăžēcŤĹăăŏČăĹăăăijăijŘăŇŮăŤřă■ŮrijŽ

```
>>> x = 1.2345
>>> format(x, '>10')
'          1.2345'
>>> format(x, '^10.2f')
'    1.23    '
>>>
```

## èõìèõž

åIJlèĀAçŽĐäzççăĀäy■īījNă;ăçzRăyŷăijŽçIJNăĹrēcñçŦlăİēăăijăijRăNŨăŨĜăIJñçŽĐ  
% æŞ■ă;IJçñēăĀĈærŦăēĈīījŽ

```
>>> '%-20s' % text
'Hello World          '
>>> '%20s' % text
'          Hello World'
>>>
```

ă;ĒæŸrīījNăIJlăŨrçĹĹæIJñäzççăĀäy■īījNă;ăăžŦèrēăijŸăĒĹéĀĹ'æNŦ'  
format() åĜ;æŦræĹŨèĀĒæŨzæşŦăĀĈ format() èēĀærŦ %  
æŞ■ă;IJçñēçŽĐăĹşèĈ;æŽŦăyžăijžăđ'găĀĈ åžŭăyŦ format() äžşærŦă;ŧçŦĪ  
ljust(), rjust() æĹŨ center() æŨzæşŦăæŽŦéĀŽçŦĪīījNă  
ăŽăäyžăŏĈăRŦăzēçŦĪlăİēăăijăijRăNŨăžzæĎŦăržēsăīījNēĀNăy■ăžĒăžĒæŸŦă■ŨçñēăyşăĀĈ  
 æĈăēđIJăĈşēēĀăŏNăĒĪăžĒēğç format() åĜ;æŦŦçŽĐæIJĹçŦĪçĹ'žæĀğīījNă  
èrŭăŦŦĈèĀĈ åIJçžŧPythonæŨĜæăç

## 4.14 2.14 åŦĹăžŭæNijæŐēă■Ũçñēăyş

### éŨŏéçŸ

ă;ăæĈşăŦĒăĜăäyĹăŦŦçŽĐă■ŨçñēăyşăŦĹăžŭăyžăyĀăyĹăđ'ğçŽĐă■Ũçñēăyş

### èğçăĒşæŨzæăĹ

æĈăēđIJă;ăæĈşēēĀăŦĹăžŭçŽĐă■ŨçñēăyşæŸŦăIJăyĀăyĹăžŦăĹŨăĹŨèĀĒ iterable  
ăy■īījNēĈçăžĹæIJăăŦñçŽĐæŨzăijRăŦşæŸŦă;ŧçŦĪ join() æŨzæşŦăĀĈærŦăēĈīījŽ

```
>>> parts = ['Is', 'Chicago', 'Not', 'Chicago?']
>>> ' '.join(parts)
'Is Chicago Not Chicago?'
>>> ', '.join(parts)
'Is, Chicago, Not, Chicago?'
>>> ''.join(parts)
'IsChicagoNotChicago?'
>>>
```

åĹçIJNēŦŭăİēīījNēŧŽçğ■ē■æşŦçIJNăyĹăŐžăijžærŦè;ĈæĀŦīījNă;ĒæŸŦ  
join() ècñăNĜăŏžăyžă■ŨçñēăyşçŽĐăyĀăyĹæŨzæşŦăĀĈ  
èŧŽăăŭăĀŽçŽĐēĈĪăĹĒăŐşăŽăæŸŦă;ăæĈşăŐžèŧđæŐēçŽĐăržēsăăŦŦèĈ;æİēēĜĪăŦĐçğ■ăy■ăŦNŦçŽĐæŦŦæ■  
 æĈăēđIJăIJlăĹ'ĀæIJĹ'èŧŽăžŽăržēsăăyĹēĈ;ăŏžăžĹ'ăyĀăyĹ join()  
æŨzæşŦăŸŐăŸ;æŸŦăĒŨă;ŽçŽĐăĀĈ åŽăæ■đ'ă;ăăŦĪēIJăēēĒăNĜăŏžă;ăæĈşēēĀçŽĐăĹĒăĹ'şă■Ũçñēăyşă  
join() æŨzæşŦăŐžăŦĒæŨĜăIJñçĹĹĜăŏŦçžĐăŦĹēŦŭăİēăĀĈ

æĈăēđIJă;ăăžĒăžĒăŦŦæŸŦăŦĹăžŭăŦşæŦŦăĜăäyĹă■ŨçñēăyşīījNă;ŧçŦĪăĹăăŦŦŦ(+ )éĀŽăyŷăăşçžŦèŭşăđ'ş

```
>>> a = 'Is Chicago'
>>> b = 'Not Chicago?'
>>> a + ' ' + b
'Is Chicago Not Chicago?'
```

åŁääRû(+)  
æŞ■ä;IJçñåIJlä;IJäyžäYÄzžZäd'■æÍCā■ŪçñäyşæäijäijRāŃŪçžDæžŁäzčæŪzæąŁçžDæŪúā

```
>>> print('{} {}'.format(a,b))
Is Chicago Not Chicago?
>>> print(a + ' ' + b)
Is Chicago Not Chicago?
>>>
```

æĈCæđIJä;äæĈşåIJläzŔçăAäy■ärĖäyd'äylā■ŪelĈā■ŪçñäyşāŔĹāzűeĭūæİēijŃä;ääŔĤēIJÄēæAçóĀā■Ŧçž

```
>>> a = 'Hello' 'World'
>>> a
'HelloWorld'
>>>
```

## èĹèőž

ā■ŪçñäyşāŔĹāzűāŔŕèĈ;çIJŃäyŁāŌžāzűäy■ēIJÄēæAçŦĹäyĀæŦŦ'èŁCæİēèĹèőžāĀĈ  
ä;EæŸŕäy■āžŦēŕēārŔçIJŃēŁZäyĤēŪōēĈŸŕijŃçĹŃāžŔāŚŸēĀŽäyŷāIJā■ŪçñäyşæäijäijRāŃŪçžDæŪúāĀŽāž

æIJÄéĜ■ēæAçžDēIJÄēæAäijŦēĭūæşĹæĐŔçžDæŸŕijŃā;ŞæĹŚāžŋä;ŁçŦĹāŁääRû(+)  
āžäyžāŁääRûēŁđæŌēäijžäijŦēĭūāĖĖā■Ÿäd'■āŁüāžēāŔĹādĈāIJġāžđæŦūæŞ■ä;IJāĀĈ  
çŁ'žāŁŋçžDŕijŃä;äæŕyēŁJēĈ;äy■āžŦāĈŔäyŃēĹĈēŁZæūāāĖŽā■ŪçñäyşēŁđæŌēäzčçăAŕijž

```
s = ''
for p in parts:
    s += p
```

èŁžĈĝ■āĖžæşŦäijžæŕŦä;ŁçŦĹ join() æŪzæşŦēŔēāŃçžDēæAæĖcäyĀäžžŕijŃāžäyžæŕŔäyĀæŋæŁ  
ä;äæIJÄæē;æŸŕāĖĹæŦūēŁZæŁĀæIJĹçžDā■ŪçñäyşçŁĜæġŦçDūāŔŌāĖ■ārĖāōĈāžŋēŁđæŌēēĭūæİēāĀĈ

äyĀäyĤçŽyāržæŕŦē;ĈēAĤæŸŌçžDæŁĀäüĝæŸŕāĹĹçŦĹçŦşæĹŔāžĹēāĹē;ĹäijŔ(āŔĈēĀĈ1.19ārŔēŁĈ)ē;ŋā

```
>>> data = ['ACME', 50, 91.1]
>>> ','.join(str(d) for d in data)
'ACME,50,91.1'
>>>
```

ārŃæäüēŁŸā;ŪæşĹæĐŔäy■āŁĖēæAçžDā■ŪçñäyşēŁđæŌēæŞ■ä;IJāĀĈæIJĹæŪúāĀžçĹŃāžŔāŚŸāIJlä

```
print(a + ':' + b + ':' + c) # Ugly
print(':'.join([a, b, c])) # Still ugly
print(a, b, c, sep=':') # Better
```

ā;ŠæūāāŔĹā;ŁçŦĪĪ/OæŠ■ā;IJāŠŦā■ŪçņęäyšēŁđæŌēæŠ■ā;IJçŽĐæŪūāĀŽīijŦæIJĻ'æŪūāĀŽēIJĀēēAāžŦ  
æŦŦāēČīijŦēĀČēŽŚāyŦēĪčçŽĐäyđ'çŋŦāžčçāAçĻ'ĠæōŦīijŽ

```
# Version 1 (string concatenation)
f.write(chunk1 + chunk2)

# Version 2 (separate I/O operations)
f.write(chunk1)
f.write(chunk2)
```

āēČæđIJäyđ'äyĹā■Ūçņęäyšā;ĹārŦīijŦēČčāžĹçŋŋäyĀäyĹçĻ'ĹæIJŋæĀgēČ;āijŽæŽŦ'āē;āžŽīijŦāŽāāyžĪ/Oç  
āŦēād'ŪäyĀæŪzéĪčīijŦæČæđIJäyđ'äyĹā■Ūçņęäyšā;Ĺād'gīijŦēČčāžĹçŋŋāžŦäyĹçĻ'ĹæIJŋāŦŦēČ;āijŽæŽŦ'āēĹ  
āŽāāyžāōČēAāāĒ■āžĒāĹZāžžāyĀäyĹā;Ĺād'gçŽĐäyŦ'æŪūçžŚæđIJāžūāyŦēēAāđ'■āĹūād'gēĠŦçŽĐāĒēĀ■Ÿā  
ēŁŸæŸŦēČčāŦēēŦīijŦæIJĻ'æŪūāĀŽæŸŦēIJĀēēAæāžæ■ōā;āçŽĐāžŦçŦĪçĪŦāžŦçĻ'žçČžæĪēāĒāōŽāžŦēŦēā;Ł

æIJĀāŦŌēŦĹäyĀäyŦīijŦæČæđIJā;āāĠēĀđ'ĠçijŪāĒŽæđĐāžžād'gēĠŦārŦā■ŪçņęäyšçŽĐē;ŚāĠžāžççā  
ā;āæIJĀāē;ēĀČēŽŚāyŦā;ŁçŦĪçŦšæĹŦāŽĪāĠ;æŦīijŦāĹ'çŦĪyieldēŦāŦēāžgçŦšē;ŚāĠžçĻ'ĠæōŦāĀČæŦŦāēČ

```
def sample():
    yield 'Is'
    yield 'Chicago'
    yield 'Not'
    yield 'Chicago?'
```

ēŁŽçg■æŪžæşŦäyĀäyĹæIJĻ'ēūčçŽĐæŪzéĪçæŸŦāōČāžūæşqæIJĻ'āržē;ŚāĠžçĻ'ĠæōŦāĹŦāžŦēēAæĀŌæāū  
ā;ŦāēČīijŦā;āāŦŦāžēçōĀā■ŦçŽĐä;ŁçŦĪ join() æŪžæşŦārĒēŁžāžçĻ'ĠæōŦāŦĹāžūēŦūæĪēīijŽ

```
text = ''.join(sample())
```

æĹŪēĀĒä;āāžşāŦŦāžēāŦēā■ŪçņęäyşçĻ'ĠæōŦēĠ■āōŽāŦŦāĹŦ/OīijŽ

```
for part in sample():
    f.write(part)
```

āĒ■æĹŪēĀĒä;āēŁŸāŦŦāžēāĒēŽāĠžāyĀāžŽçžşāŦĹĪ/OæŠ■ā;IJçŽĐæūūāāŦĹæŪžæāĹīijŽ

```
def combine(source, maxsize):
    parts = []
    size = 0
    for part in source:
        parts.append(part)
        size += len(part)
        if size > maxsize:
            yield ''.join(parts)
            parts = []
            size = 0
    yield ''.join(parts)

# çžşāŦĹæŪĠžāžūæş■ā;IJ
with open('filename', 'w') as f:
    for part in combine(sample(), 32768):
        f.write(part)
```

```
>>> s.format(name='Guido')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'n'
>>>
```

```
__missing__(): æÚzæsŤçŽĐa■ŮaĚýaržesajijŇarsaČŘayŇelčefŽæuuijŽ
```

```
class safesub(dict):  
    """éŸšæ■ckeyæL'¿äÿ■åĹř"""  
    def __missing__(self, key):  
        return '{' + key + '}'
```

çŮraIJlä;ääRřazěaĹ'çŤlèfŽäytçszaŇĚčĚē;ŠaĚēaŔŌäijæéĂšçžŽ format\_map() iijŽ

```
>>> del n # Make sure n is undefined  
>>> s.format_map(safesub(vars()))  
'Guido has {n} messages.'  
>>>
```

æęĆadIJä;ääRŚçŮřèĠaũsâIJläžčçăAäÿ■écŚçžAçŽĐæL'gëaŇèfŽäžŽæ■éld'rijŇä;ääRřazěaŕEaŔŸéGRæ

```
import sys  
  
def sub(text):  
    return text.format_map(safesub(sys._getframe(1).f_locals))
```

çŮraIJlä;ääRřazěaČŘayŇelčefŽæuuaĚžæEijŽ

```
>>> name = 'Guido'  
>>> n = 37  
>>> print(sub('Hello {name}'))  
Hello Guido  
>>> print(sub('You have {n} messages.'))  
You have 37 messages.  
>>> print(sub('Your favorite color is {color}'))  
Your favorite color is {color}  
>>>
```

## èóléőž

äd'Žäzt'äzælēçŤšäžŮPythonçijžazŔaržâŔŸéGRæŽŁæ■ççŽĐaĚĚç;őæŤŕæŇAæĀŇarijèĠt'äžEaŔĐçg■äÿ  
ä;IJäÿžæIJnèĹCäÿ■ašŤçđ'žçŽĐäÿÄäÿĹaŔŕèÇ;çŽĐègčâEşæŮzæaĹiiijŇä;ääRřazěæIJL'æŮuāĀŽaijŽçIJŇaĹŕäČ

```
>>> name = 'Guido'  
>>> n = 37  
>>> '%(name) has %(n) messages.' % vars()  
'Guido has 37 messages.'  
>>>
```

ä;ääRřèÇ;èfŸaijŽçIJŇaĹŕa■ŮçñęäÿşæĹæĹççŽĐä;ççŤliijŽ

```
>>> import string
>>> s = string.Template('$name has $n messages.')
>>> s.substitute(vars())
'Guido has 37 messages.'
>>>
```

çDûëÄÑñijÑ format() åŠÑ format\_map() çŽÿæŕTèçÇäyLéÍcèfZäzZæŰzæqLèÄÑñüšæZt'åLääĖL  
 ä;fçTÍ format() æŰzæſTèfYæIJL'äyÄäyIäç;äd'ĐârſæYřä;ääRřäzèèŎüä;Űâržā■ŰçñëäyšæäijäijRāÑŰçŽĐ  
 èÄÑèfZäzZçL'zæÄgæYřä;fçTÍlāČRæIqæİfā■ŰçñëäyšäzNçšzçŽĐæŰzæqLäy■āRřèČ;èŎüä;ŰçŽĐāĀĆ

æIJñæIJzèfYèČlāLEäzNçz■äzEäyÄäzŽénYçžgçL'zæÄgāĀĆæYāârDæLŰèÄĖā■ŰāĖyçšzäy■čšIJäyžäzž  
 \_\_missing\_\_() æŰzæſTāRřäzèèŎl'ä;ääŰZäzL'äçCä;Tād'ĐçŘEçijžād'šçŽĐāĀijāĀĆ åIJ  
 SafeSub çšzäy■ñijÑèfZäyIæŰzæſTècñāŰZäzL'äyžâržçijžād'šçŽĐāĀijèfTāZđäyÄäyIā■ää;■çñëāĀĆ  
 ä;ääRřäzèāRŠçŎřçijžād'šçŽĐāĀijäijZāGžçŎřāIJlçzšæđIJā■Űçñëäyšäy■(āIJlèŕCèŕTçŽĐæŰüāÄZāRřèČ;ä;Lā  
 KeyError äijČäyŷāĀĆ

sub() åĜ;æTřä;fçTÍ sys.\_getframe(1) èfTāZđèŕČçTÍèÄĖçŽĐæāLäyğāĀĆāRřäzèäzŎäy■èŎfèŰ  
 f\_locals æIèèŎüä;ŰāsÄéČlāRŸéGRāĀĆ æŕñæŰäçŰSèŰŎçzIād'ğçČlāLEæČĖāĖtäyNāIJlāzççāAäy■āŰzç  
 ä;EæYřijNāržäzŎāČRā■ŰçñëäyšæZfæ■cāüèāĖüāĜ;æTřèÄÑèIĀāŰČæYřèIđäyŷæIJL'çTÍçŽĐāĀĆ  
 āRēād'ŰñijNāĀijä;ŰäſIæĐRçŽĐæYř f\_locals æYřäyÄäyIād'■āLŰèŕČçTÍlāĜ;æTřçŽĐæIJñāIJřāRŸéGRçŽ  
 āŕ;çŏqä;ääRřäzèæTzāRŸ f\_locals çŽĐāĖĖāŰzñijNā;EæYřèfZäyIāfŏæTzāržäzŎāRŎéIççŽĐāRŸéGRèŎfè  
 æL'ÄäzèñijNèŽ;èŕt'èŎfèŰŎäyÄäyIæāLäyğçIJNäyLāŰzä;LéČIæAññijNā;EæYřāržāŰČçŽĐäzžä;Tæš■ā;IJäy■ā

## 4.16 2.16 äžèæÑĜāŰŽāLŰāŰ;æäijäijRāÑŰā■Űçñëäyš

### éŰŎéćŰ

ä;äæIJL'äyÄäzŽèTfā■ŰçñëäyšñijNæČšäzèæÑĜāŰŽçŽĐāLŰāŰ;ārĖāŰČäzñèĜ■æŰřæäijäijRāÑŰāĀĆ

### èğçāEşæŰzæqL

ä;fçTÍ textwrap æIqāIŰæIèæäijäijRāÑŰā■ŰçñëäyšçŽĐèç;šāĜzāĀĆæŕTäçČññNāAĜäçCä;ææIJL'äyNā

```
s = "Look into my eyes, look into my eyes, the eyes, the eyes, \
the eyes, not around the eyes, don't look around the eyes, \
look into my eyes, you're under."
```

äyNéIçæijTçd'žä;fçTÍ textwrap æäijäijRāÑŰā■ŰçñëäyšçŽĐād'Žçğ■æŰzäijRñijŽ

```
>>> import textwrap
>>> print(textwrap.fill(s, 70))
Look into my eyes, look into my eyes, the eyes, the eyes, the eyes,
not around the eyes, don't look around the eyes, look into my eyes,
you're under.

>>> print(textwrap.fill(s, 40))
Look into my eyes, look into my eyes,
the eyes, the eyes, the eyes, not around
```

```
>>> print(textwrap.fill(s, 40, initial_indent='    '))
    Look into my eyes, look into my
    eyes, the eyes, the eyes, the eyes, not
    around the eyes, don't look around the
    eyes, look into my eyes, you're under.

>>> print(textwrap.fill(s, 40, subsequent_indent='    '))
    Look into my eyes, look into my eyes,
    the eyes, the eyes, the eyes, not
    around the eyes, don't look around
    the eyes, look into my eyes, you're
    under.
```

```
textwrap ælǣlIŭarfzāžŌā■ŬçņēäÿsæL'Sā■ræYřéldäÿyæIJL'çTİçŽDrijÑçL'zâLnæYřā;ŞăjääÿNæIJZeç;
äjäåRfrazëä;ŁçTİos.get_terminal_size() æÚzæsTælëëÖŭāRŪčZĹcnrcŽDād'gārRåržāryāĂĆærTāeČ
```

`fill()` æÚzæşŦæÖëáRÛäýĂăžŽăĚúázŰáRréĂLăRCæŦræIěæŎğĂŁutabijNër■ăRēcȳŞărç■LăĂĆ  
ăRĆéYĚ `textwrap.TextWrapper`æŬGăaç èŬăăRŮăZt'ăd'ŽăĚĖăőžăĂĆ

ä;äČšârEHTMLæŁŨëÄËXMLăõđä;ŠăëĆ      &entity;      æŁŨ      &#code;  
æZfæ■cäyžăržăžTčZDæŨĞæIJňăĂĆ      âE■ëÄËijŃă;ăeIJăèëAè;ňæ■cæŨĞæIJňăy■çL'žăõŽčZDă■Ůçņę(æřTă  
>, æŁŨ &)ăĂĆ

æĈæđIjä;äăĈşæŻfæ■ćæŮĜæIJnă■Ůçņęäÿsäÿ■çŽĐ âĀŸ<âĂŹ æĹŮèĂĚ âĂŸ>âĂŹ  
iijNă;ĴćŢĭ ħtml.escape() âĠ;æŦrăRăřăzěă;ĹăőzăŸŞçŽĐăőŦăĹŦăĂĈærŦăęĈiijŽ

```
>>> s = 'Elements are written as "<tag>text</tag>".'  
>>> import html  
>>> print(s)  
Elements are written as "<tag>text</tag>".
```



```
>>> print(html.escape(s))
Elements are written as '<tag>text</tag>'.

>>> # Disable escaping of quotes
>>> print(html.escape(s, quote=False))
Elements are written as "<tag>text</tag>".
>>>
```

æĈæđĬä;äæ■ĉăĬlâđ'ĐĉŘĚçŽĐæŸřASCIIæŮĜæĬññĭjŇázúäyŤæČšřĚéĭđASCIIæŮĜæĬññřžâžŤçŽĐç  
 âŖřäžëçžZæŠŖäžŽĬ/OăĜ;æŤřäĭjäéĂŠăŖĈæŤř errors='xmlcharrefreplace'  
 æĭëèĭăĹŖëŤZäyĭçŽôăĂĈæŤřæĈñĭjŽ

```
>>> s = 'Spicy Jalapeño'
>>> s.encode('ascii', errors='xmlcharrefreplace')
b'Spicy Jalape&#241;o'
>>>
```

äyžäžĚæŽæ■ĉæŮĜæĬññäy■çŽĐçĭjŮčăĂăđă;ŠĭĭjŇă;ăéĬĂĊëĂă;ŤçŤĭăŖëăđ'ŮäyĂçĝ■æŮžæšŤăĂĈ  
 æĈæđĬä;äæ■ĉăĬlâđ'ĐĉŘĚHTMLæĹŮĊĂĚXMLæŮĜæĬññĭjŇëŤçĬĂăĒĹă;ŤçŤĭăyĂäyĭăŖĹĒĂĈçŽĐHTML  
 éĂžăyŷæĈĚăĚĭyŇĭĭjŇëŤžăžŽăüăăĚüăĭjŽëĜĭăĹăŽŤæ■ĊëŤžăžŽçĭjŮčăĂăĭĭjŇă;ăæŮăéĬĂæŇĒăŤĈăĂĈ  
 æĬĹæŮŮăĂžĭjŇăæĈæđĬä;äæŮĊæŤŮăĹŖăžĚäyĂăžŽăŖŇæĬĹçĭjŮčăĂăĭĭjŽĐăŮšăĝŇæŮĜæĬññĭjŇëŤ  
 éĂžăyŷă;ăăŖĭĹĂĊëĂă;ŤçŤĭHTMLæĹŮĊĂĚXMLëĝĊæđŖăŽĭçŽĐäyĂăžŽçŽyăĚšăüăăĚüăĜ;æŤř/æŮžæšŤă■

```
>>> s = 'Spicy &quot;Jalape&#241;o&quot;'
>>> from html.parser import HTMLParser
>>> p = HTMLParser()
>>> p.unescape(s)
'Spicy "Jalapeño".'
>>>
>>> t = 'The prompt is &gt;&gt;&gt;'
>>> from xml.sax.saxutils import unescape
>>> unescape(t)
'The prompt is >>>'
>>>
```

## èőĭëőž

ăĬĹçŤšæĹŖHTMLæĹŮĊĂĚXMLæŮĜæĬñçŽĐæŮŮăĂžĭjŇăæĈæđĬäæ■čăăçŽĐë;Ňæ■ĊĹžăăĹăăĜëĊ  
 çĹžăĹŇæŸřă;Šă;ăă;ŤçŤĭprint()ăĜ;æŤřæĹŮĊĂĚăĚŮăžŮă■ŮĉŇäyšæăĭĭjŖăŇŮæĭăžĝçŤšëĭŠăĜžçŽĐă  
 ä;ŤçŤĭăĈŖhtml.escape()çŽĐăüăăĚüăĜ;æŤřăŖŖăžëăĭăĹăőžæŸšçŽĐëĝĊăĚšëŤžçšžéŮőĊŸăĂĈ

æĈæđĬä;äæČšžăžăĚŮăžŮăŮŮžăĭjŖăđ'ĐĉŘĚæŮĜæĬññĭjŇëŤŸæĬĹăyĂăžŽăĚŮăžŮčŽĐăüăăĚüăĜ;æŤřă  
 xml.sax.saxutils.unescape()ăŖŖăžëăyăăĹăĭăăĂĈ  
 çĐŮĊĂŇñĭjŇă;ăăžŤëŖăăĒĹĕŖĈçăŤăyĚăĊŽăĂŮăăüă;ŤçŤĭăyĂäyĭăŖĹĒĂĈçŽĐëĝĊæđŖăŽĭăĂĈ  
 æŤřăĊĭĭjŇăæĈæđĬä;ăăĬĹăđ'ĐĉŘĚHTMLæĹŮXMLæŮĜæĬññĭjŇ  
 ä;ŤçŤĭăšŖăyĭëĝĊæđŖăĹăăĭŮăŖŤăĊĈhtml.parseæĹŮxml.etree.ElementTree  
 äŮšçžŖăyăă;ăëĜĭăĹăĹăđ'ĐĉŘĚăžĚçŽyăĚšçŽĐæŽŤæ■ĊçžĚĹĈăĂĈ

## 4.18 2.18 á■Ůčņęäýšäzd'çL'NèğčædŘ

### éŮóécŸ

ä;äæIJL'äýÄäýł■ŮčņęäýšiiĴNæČšäzŌäüçèĜšāRšārEāĔüèğčædŘäýžäýÄäýłäzd'çL'NætAāĂĈ

### èğčāEşæŮzæąŁ

āAĜāçCā;äæIJL'äýNéÍçèŁZæäüäýÄäýłæŮĜæIJñā■ŮčņęäýšiiĴ

```
text = 'foo = 23 + 42 * 10'
```

äýžāZĖäzd'çL'NāNŮā■ŮčņęäýšiiĴNā;ääý■äzĖĖIJĀèçAāNžéĔ■æłāāijRiiĴNèŁŸāŁŮæNĜāōZæłāāijRçŽDç  
ærŤāçĈiiĴNā;āāRrèĈ;æČšārEā■ŮčņęäýšāĈRäýNéÍçèŁZæäüè;ñæ■čäýžāžRāŁŮāržiiĴ

```
tokens = [('NAME', 'foo'), ('EQ', '='), ('NUM', '23'), ('PLUS', '+'),  
          ('NUM', '42'), ('TIMES', '*'), ('NUM', '10')]
```

äýžāZĖæL'gèqNèŁZæäüçŽDāŁĜāŁEĕiiĴNçññäýÄæ■čārśæŸrāĈRäýNéÍçèŁZæäüāŁ'çŤłāŚ;āŘ■æ■ŤèŌüçž

```
import re  
NAME = r'(?P<NAME>[a-zA-Z_][a-zA-Z_0-9]*)'  
NUM = r'(?P<NUM>\d+)'  
PLUS = r'(?P<PLUS>\+)'  
TIMES = r'(?P<TIMES>\*)'  
EQ = r'(?P<EQ>=)'  
WS = r'(?P<WS>\s+)'  
  
master_pat = re.compile('|'.join([NAME, NUM, PLUS, TIMES, EQ, WS]))
```

āIJłäýŁéÍççŽDæłāāijRäý■iiĴN ?P<TOKENNAME> çŤłāžŌçžZäýÄäýłæłāāijRāŚ;āŘ■iiĴNāŁZāŘŌéÍçä;ŁçŤ

äýNäýÄæ■ēiiĴNäýžāZĖäzd'çL'NāNŮiiĴNā;ŁçŤłæłāāijRāržèśāāŁārŚèçñāžžçšēéAşçŽD  
scanner() æŮzæşŤāĂĈ èŁZäýłæŮzæşŤäijŽāŁZāžžäýÄäýł  
scanner āržèśāiiĴN āIJłèŁZäýłāržèśāāýŁäý■æŮ■çŽDèrĈçŤí match()  
æŮzæşŤäijŽäýÄæ■ēæ■ēçŽDæL'náæRRçŽōæāĜæŮĜæIJñiiĴNærRæ■äýÄäýłāNžéĔ■āĂĈ  
äýNéÍçāŸræijŤçd'žäýÄäýł scanner āržèśāāçCā;Ťāüèä;IJçŽDäzd'äžŠāijRäŁNā■ŘiiĴ

```
>>> scanner = master_pat.scanner('foo = 42')  
>>> scanner.match()  
<_sre.SRE_Match object at 0x100677738>  
>>> _.lastgroup, _.group()  
('NAME', 'foo')  
>>> scanner.match()  
<_sre.SRE_Match object at 0x100677738>  
>>> _.lastgroup, _.group()  
('WS', ' ')  
>>> scanner.match()  
<_sre.SRE_Match object at 0x100677738>
```

```
>>> _.lastgroup, _.group()
('EQ', '=')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('WS', ' ')
>>> scanner.match()
<_sre.SRE_Match object at 0x100677738>
>>> _.lastgroup, _.group()
('NUM', '42')
>>> scanner.match()
>>>
```

åödéZĚä;ŁçTlêŁŻçğ■æŁĂæIJŁŻDæŮúăĂZġijŃăŔŕăzēăĹăőzæŸŞçŻDăČŔăyŃéİcēŁZæăüăŕĚăyŁèŁŕăz

```
def generate_tokens(pat, text):
    Token = namedtuple('Token', ['type', 'value'])
    scanner = pat.scanner(text)
    for m in iter(scanner.match, None):
        yield Token(m.lastgroup, m.group())

# Example use
for tok in generate_tokens(master_pat, 'foo = 42'):
    print(tok)

# Produces output
# Token(type='NAME', value='foo')
# Token(type='WS', value=' ')
# Token(type='EQ', value='=')
# Token(type='WS', value=' ')
# Token(type='NUM', value='42')
```

åĚČædIJă;ăæČşēŁĚæzd'ăzd'çŁŃæŁAġijŃă;ăăŔŕăzēăőZăZŁæŽŁ'ăd'ŽçŻDçŤşæŁŔăZlăĜ;æŤŕæLŮèĂĚă;æŕŤăĚČġijŃăyŃéİcēġĤçd'zæĂŬæăüēŁĚæzd'æŁĂæIJŁçŻDçŁ'žçŽ;ăzd'çŁŃġijŽ

```
tokens = (tok for tok in generate_tokens(master_pat, text)
           if tok.type != 'WS')
for tok in tokens:
    print(tok)
```

## ëőİëőž

éĂŽăyŷæİëèőşăzd'çŁŃăŃŮæŸŕăĹăd'ŽénŸçžğæŮĜæIJñēğçædŔăyŎăd'DçŔĚçŻDçñăyĂæ■čăĂČăyžăžĚă;ŁçTlăyŁéİcçŻDæŁŋæŔŔæŮzæşŤġijŃă;ăéIJăĚæAĚőŕă;ŔēŁŽéĜŃăyĂăžŻéĜ■ēēAçŻDăĜăçČzăĂČçñăyĂçČzăŕşæŸŕă;ăăŁĚēăzçăőēőd'ă;ăă;ŁçTlæ■čăŁŽēăİēĹăġijŔæŃĜăőZăžĚæŁĂæIJŁēĹŞăĚēăy■ăŔŕēČ;ăĜăĚČædIJæIJŁăzză;Ťăy■ăŔŕăŃzéĚ■çŻDæŮĜæIJŋăĜçŎŕăžĚġijŃæŁŋæŔŔăŕşăġijŽçŽŁæŎēăAIJæ■čăĂČēŁZă

ăzd'çŁŃçŻDēăžăžŔăžşæŸŕæIJŁă;şăş■çŻDăĂČ re æĹăăİŮăġijŽæŃŁçĚĝæŃĜăőZăē;çŻDēăžăžŔăŎzăAăZăæ■d'ġijŃăĚČædIJăyĂăyŁăĹăġijŔæAŕăē;æŸŕăŔēăyĂăyŁæŽŁ'ēŤŁăĹăġijŔçŻDă■Ŕă■ŮçŋēăyşġijŃéČčăžŁă;ăē

```

LT = r'(?P<LT><)'
LE = r'(?P<LE><=)'
EQ = r'(?P<EQ>=)'

master_pat = re.compile(''.join([LE, LT, EQ])) # Correct
# master_pat = re.compile(''.join([LT, LE, EQ])) # Incorrect

```

çññāžŃäyġæġāijRæYřéŤŽčŽDřijŇāZāyžāōČāijŽārĚæŮĜæIJñ<=āŇzéĚäyžāzd'çL'ŇLTçť'ğèùşçİĀEQ

æIJĀāŔŌřijŇāġæIJĀēçAçŤZæĎRäyŇāŔāŮçñçäyşāġāijRçŽDæġāijRāĀĆæŕŤāçĆřijŇāAĜèöġāġæIJ

```

PRINT = r'(?P<PRINT>print)'
NAME = r'(?P<NAME>[a-zA-Z_][a-zA-Z_0-9]*)'

master_pat = re.compile(''.join([PRINT, NAME]))

for tok in generate_tokens(master_pat, 'printer'):
    print(tok)

# Outputs :
# Token(type='PRINT', value='print')
# Token(type='NAME', value='er')

```

ăĚşāžŌæZŕ'énYéYŭçŽDāzd'çL'ŇāŇŮæĹĀæIJřijŇāġāŔřèČġēIJĀēçAæşēçIJŇ PyPars-

ing æĹŮèĀĚ PLY āŇĚāĀĆ äyĀäyġerČçŤĪPLYçŽDāġŇāŔāIJläyŇäyĀèĹČāijŽæIJL'æijŤçď'žāĀĆ

## 4.19 2.19 áódçŌřäyĀäyġçŏĀāŤçŽDēĀŞāġŞäyŇéŽāĹĚæďŔāŽĪ

### éŮóécŸ

äġæČşæāžæŏäyĀçzDëræşŤğDāĹŽèğçæďŔæŮĜæIJñāžūæĹ'ğèāŇāŚġāzd'řijŇæĹŮèĀĚæďDēĀäyĀā

æÇæďIJēræşŤēĪdāyŷçŏĀāŤřijŇāġāŔřāžèèĜġāūsāĚŽèçŽäyġèğçæďŔāŽĪřijŇèĀŇäyæYřāġçŤĪäyĀäžZæāĚ

### èğčĀĚşæŮžæāĹ

āIJġēçŽäyġéŮóécŸäyřijŇæĹSāžñéŽĚäyŏēŏēŏžæāžæŏçĹ'žæŏĹēræşŤāŌžèğçæďŔæŮĜæIJñçŽDēŮóé

äyžāžĚèçZæāūāĀřijŇāġæçŮāĚĹèçAāžèBNFæĹŮèĀĚBNFāġāijRæŇĜāŏŽäyĀäyġæāĜāĜĚēræşŤāĀĆ

æŕŤāçĆřijŇäyĀäyġçŏĀāŤæŤŕāŕæēāĹèġāijRēræşŤāŔřèČġāČŔäyŇēĲçèçZæāūřijŽ

```

expr ::= expr + term
      | expr - term
      | term

term ::= term * factor
      | term / factor
      | factor

```

```
factor ::= ( expr )
        |   NUM
```

æŁŨèĀĖiijNäzēEBNFā;ćaijRiijŽ

```
expr ::= term { (+|-) term } *
term ::= factor { (*|/) factor } *
factor ::= ( expr )
         |   NUM
```

āIJĖBNFäy■iijNēcñāNĖāRñāIJĭ { . . . } \* äy■çŽDēğDāŁŽæYřāRřéĀL'çŽDāĀĆ\*āzçèāĭ0æñæŁŨād'ŽæŁŨāIJĭiijNāēĆædIJā;āārZBNFçŽDāuēä;IJæIJžāŁŨēŁYäy■æYřāŁæYŖçŽçŽDēřIiijNārsæŁŁāōĆā;ŠāAäyĀēŁŨæIēēōsriijNēğçædRçŽDāŖçRĖārsæYřā;āāŁ'çTĭBNFāōNæLRād'ŽäyŁæŽŁæ■ćāŠNæL'āśTāzēāNzéĖäyžāzĖæijTçd'žriijNāĀĖēōĭä;āæ■ćāIJĭēğçædRā;ćāēĆ 3 + 4 \* 5 çŽDēāĭē;ĭāijRāĀĆēŁŽäyŁēāĭē;ĭāijRāĖĖēĀēĀŽēŁĖā;ŁçTĭ2.18ēŁĆäy■āzNçz■çŽDæŁĀæIJřāŁēēğçäyžäyĀçzDāzd'çL'NætĀāĀçzŠædIJāRřēĆ;æYřāČRäyNāŁŨēŁŽæāuçŽDāzd'çL'NāzRāŁŨiijŽ

```
NUM + NUM * NUM
```

āIJĭæ■d'āšžçāĀäyŁriijN ēğçædRāŁĭā;IJāijŽēřTçĭĀāŖzéĀŽēŁĖæŽŁæ■ćæŠ■ā;IJāNzéĖ■ēř■æšTāŁrē;ŠāĖ

```
expr
expr ::= term { (+|-) term } *
expr ::= factor { (*|/) factor } * { (+|-) term } *
expr ::= NUM { (*|/) factor } * { (+|-) term } *
expr ::= NUM { (+|-) term } *
expr ::= NUM + term { (+|-) term } *
expr ::= NUM + factor { (*|/) factor } * { (+|-) term } *
expr ::= NUM + NUM { (*|/) factor } * { (+|-) term } *
expr ::= NUM + NUM * factor { (*|/) factor } * { (+|-) term } *
expr ::= NUM + NUM * NUM { (*|/) factor } * { (+|-) term } *
expr ::= NUM + NUM * NUM { (+|-) term } *
expr ::= NUM + NUM * NUM
```

äyNēĭcæŁ'ĀæIJŁ'çŽDēğçædRæ■ēĭd'āRřēĆ;ēIJĀēēĀēŁšĆzæŨūēŨř'āijDæYŖçŽçŽiijNā;ĖæYřāōČāzñāōçññäyĀäyŁē;ŠāĖēāzd'çL'NæYřNUMriijNāZāæ■d'æZŁæ■céēŨāĖĖĭāijŽāNzéĖ■ēĆčäyŁēĆĭāŁēāĀĆäyĀæŨēāNzéĖ■æŁRāŁšriijNārsāijŽēŁZāĖēäyNäyĀäyŁāzd'çL'N+riijNāzēæ■d'çšzæŖĭāĀĆā;ŠāūsçzRçāōāōŽäy■ēĆ;āNzéĖ■äyNäyĀäyŁāzd'çL'NçŽDæŨūāĀŽriijNāRšē;žçŽDēČĭāŁē(æřTāēĆ{ (\*|/) factor } \*)ārsāijŽēcñäyĖĖçRĖæŖĖāĀĆ āIJĭäyĀäyŁæŁRāŁšçŽDēğçædRäy■iijNæTř'äyŁāRšē;žæIJŁ'āžĖāŁ'■ēĭçŽDçšēēřĖēČNæZřriijNäyNēĭcæŁŠāzñäy;äyĀäyŁçōĀā■Tçd'žā;ŁæĭēāšTçd'žāēĆā;Tādĭ

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
"""
Topic: äyNéŽ■ēğçædRāŽĭ
Desc :
"""
```

```

import re
import collections

# Token specification
NUM = r'(?P<NUM>\d+)'
PLUS = r'(?P<PLUS>\+)'
MINUS = r'(?P<MINUS>-)'
TIMES = r'(?P<TIMES>\*)'
DIVIDE = r'(?P<DIVIDE>/)'
LPAREN = r'(?P<LPAREN>\(''
RPAREN = r'(?P<RPAREN>\))'
WS = r'(?P<WS>\s+)'

master_pat = re.compile(''.join([NUM, PLUS, MINUS, TIMES,
                                  DIVIDE, LPAREN, RPAREN, WS]))

# Tokenizer
Token = collections.namedtuple('Token', ['type', 'value'])

def generate_tokens(text):
    scanner = master_pat.scanner(text)
    for m in iter(scanner.match, None):
        tok = Token(m.lastgroup, m.group())
        if tok.type != 'WS':
            yield tok

# Parser
class ExpressionEvaluator:
    '''
    Implementation of a recursive descent parser. Each method
    implements a single grammar rule. Use the ._accept() method
    to test and accept the current lookahead token. Use the ._
    expect()
    method to exactly match and discard the next token on on the
    input
    (or raise a SyntaxError if it doesn't match).
    '''

    def parse(self, text):
        self.tokens = generate_tokens(text)
        self.tok = None # Last symbol consumed
        self.nexttok = None # Next symbol tokenized
        self._advance() # Load first lookahead token
        return self.expr()

    def _advance(self):
        'Advance one token ahead'
        self.tok, self.nexttok = self.nexttok, next(self.tokens,
        None)

```

```

def _accept(self, toktype):
    'Test and consume the next token if it matches toktype'
    if self.nexttok and self.nexttok.type == toktype:
        self._advance()
        return True
    else:
        return False

def _expect(self, toktype):
    'Consume next token if it matches toktype or raise_
↪SyntaxError'
    if not self._accept(toktype):
        raise SyntaxError('Expected ' + toktype)

# Grammar rules follow
def expr(self):
    "expression ::= term { ('+'|'-') term }*"
    exprval = self.term()
    while self._accept('PLUS') or self._accept('MINUS'):
        op = self.tok.type
        right = self.term()
        if op == 'PLUS':
            exprval += right
        elif op == 'MINUS':
            exprval -= right
    return exprval

def term(self):
    "term ::= factor { ('*'|'/') factor }*"
    termval = self.factor()
    while self._accept('TIMES') or self._accept('DIVIDE'):
        op = self.tok.type
        right = self.factor()
        if op == 'TIMES':
            termval *= right
        elif op == 'DIVIDE':
            termval /= right
    return termval

def factor(self):
    "factor ::= NUM | ( expr )"
    if self._accept('NUM'):
        return int(self.tok.value)
    elif self._accept('LPAREN'):
        exprval = self.expr()
        self._expect('RPAREN')
        return exprval
    else:
        raise SyntaxError('Expected NUMBER or LPAREN')

```

```

def descent_parser():
    e = ExpressionEvaluator()
    print(e.parse('2'))
    print(e.parse('2 + 3'))
    print(e.parse('2 + 3 * 4'))
    print(e.parse('2 + (3 + 4) * 5'))
    # print(e.parse('2 + (3 + * 4)'))
    # Traceback (most recent call last):
    #   File "<stdin>", line 1, in <module>
    #   File "exprparse.py", line 40, in parse
    #   return self.expr()
    #   File "exprparse.py", line 67, in expr
    #   right = self.term()
    #   File "exprparse.py", line 77, in term
    #   termval = self.factor()
    #   File "exprparse.py", line 93, in factor
    #   exprval = self.expr()
    #   File "exprparse.py", line 67, in expr
    #   right = self.term()
    #   File "exprparse.py", line 77, in term
    #   termval = self.factor()
    #   File "exprparse.py", line 97, in factor
    #   raise SyntaxError("Expected NUMBER or LPAREN")
    # SyntaxError: Expected NUMBER or LPAREN

if __name__ == '__main__':
    descent_parser()

```

## èóìèőž

æŮĜæIJñèġcædŘæŸräyÄäyġŁŁad'ġçŽDäyžécŸiijŇäyÄeĽnäijŽā■āçŦġā■ēçŦšā■ēçāžāçijŮērŚēr;çĹŇæŮ  
 āēČædIJā;āāIJāeĽ;āržāĒšāžŮēr■æšŦiijŇèġcædŘčŮŮæšŦç■ĽčŽyāĒšçŽDēČŇæŽřçšēērĒçŽDērĲiijŇä;āāžŦēr  
 āġĽæŸçDŮiijŇāĒšāžŮērZæŮžéĲçŽDāĒĒāōžād'ġad'ŽiijŇäy■ārRèČ;āIJèŁŽéĜŇāĒĲéČġāsŦāijĀāĀC

ār;çŮāāēČæ■d'ĲiijŇçijŮāĒŽäyÄäyġĀšā;ŠäyŇéŽ■èġcædŘāŽĲçŽDæŦr'ä;ŠæĀĲeŮræŸrærŦe;ČçŮĀā■ŦçŽ  
 āijĀāġŇçŽDæŮŮāŽiijŇä;āāĒĽeŮŮāġŮāĽĀæIJĽçŽDēr■æšŦçġDāĽŽiijŇçDŮāŮŮāŮĒāĒĒēĲnæ■cäyžäyÄäy  
 āžāæ■d'āēČædIJā;āçŽDēr■æšŦçšžäijijèŁZæāŮiijŽ

```

expr ::= term { ('+' | '-') term } *

term ::= factor { ('*' | '/') factor } *

factor ::= '(' expr ')'
         | NUM

```

ä;āāžŦērēēŮāĒĲārĒāōČäzñè;næ■cāĽŘäyĀçžDāČŘäyŇēĲçēŁZæāŮçŽDæŮžæšŦiijŽ



```
class ExpressionEvaluator:
```

```
    ...  
    def expr(self):
```

```
    ...  
    def term(self):
```

```
    ...  
    def factor(self):
```

```
    ...
```

æfRäylæÚzæşTëeAãoNæLŔçŽDäzzaLqâŁŁçôĀā■T - āōČāĚÉéazāzŌāũeèGşāRşéA■āŌEèr■æşTëgDāLŽ  
āzŌæşRçg■æDRāzLāyŁeōšīijNæÚzæşTçŽDçZōçŽDārsæYřeAāzLād'DçRĚāōNēr■æşTëgDāLŽīijNēeAāzL  
āyžāžEèfZæāũāAŽīijNéIJĀéGĜçTlāyNéIççŽDēfZāzŽāōđçŌræÚzæşTīijŽ

- āēČædIJëgDāLŽāy■çŽDāyNāyŁçņeāRūæYřāRēād'ŪāyĀāyĻēr■æşTëgDāLŽçŽDāR■āŪ(æfTāeĆtermæ  
èfZārsæYřerēçŌŪæşTāy■āĀlāyNéZ■āĀIççŽDçT'sæIē  
æŌgāLūāyNéZ■āLrāRēāyĀāyĻēr■æşTëgDāLŽāy■āŌzāĀĆ  
æIJLæŪūāĀZëgDāLŽāijŽerČçTlāũşçzRæL'gēāNçŽDæŪzæşT(æfTāeĆīijNāIJl  
factor ::= '('expr ')'  
ēfZārsæYřçŌŪæşTāy■āĀIēĀŞā;ŞāĀIççŽDçT'sæIēāĀĆ  
āy■ārfzexprçŽDērČçTlā)āĀĆ
- āēČædIJëgDāLŽāy■āyNāyĀāyŁçņeāRūæYřāyŁçL'zæŌŁçņeāRū(æfTāeĆ())īijNā;āā;ŪæşēæL'çāyNāyĀāy  
āēČædIJāy■āNzéĒ■īijNārsāzğçTşāyĀāyĻēr■æşTēTŽērřāĀĆeēfZāyĀeLÇāy■çŽD  
\_expect() æŪzæşTārsæYřçTlāIēāAŽēfZāyĀæ■ēçŽDāĀĆ
- āēČædIJëgDāLŽāy■āyNāyĀāyŁçņeāRūāyžāyĀāzZāRrēČçŽDēĀL'æNl'ēāz(æfTāeĆ +  
æLŪ-)īijNā;āāĚÉéazāzæfRāyĀçg■āRrēČ;æČĒāEĵæČĀæşēāyNāyĀāyŁāzd'çL'NīijNāRlæIJLā;ŞāŌČāN  
ēfZāzşæYřæIJNēLÇçd'žā;Nāy■ \_accept() æŪzæşTçŽDçZōçŽDāĀĆ  
āŌČçŽyā;ŞāžŌ \_expect()æŪzæşTçŽDāijsāNŪçL'ŁæIJNīijNāZāāyžāēČædIJāyĀāyŁāNzéĒ■æL'çāLrāzĒā  
ā;EæYřāēČædIJæşqæL'çāLrīijNāŌČāy■āijŽāzğçTşēTŽērřēĀNæYřāZđæzŽ(āĒAeōyāRŌçz■çŽDæçĀæ  
āijŽāzŌæIJL'ēĜ■ād'■ēČlāLēçŽDēgDāLŽ(æfTāeĆāIJlëgDāLŽēālēçāijR ::= term {  
( '+' | '-' ) term } \* āy■īijNēĜ■ād'■āLlā;IJēĀZēfGāyĀāyŁwhileāŁçŌrāIēāŌđçŌrāĀĆ  
āŁçŌrāyžā;ŞāijŽæTūēZEæLŪād'DçRĚæL'ĀæIJLçŽDēĜ■ād'■āĒČçT'āçZt'āLræşqæIJL'āĒūāzŪāĒČçT'ā  
āyĀæŪæT'rāyĻēr■æşTëgDāLŽād'DçRĚāōNæLŔīijNæfRāylæÚzæşTāijŽēfTāZđæşRçg■çzşædIJçzŽē  
ēfZārsæYřāIJlëgçædRēfĜçlNāy■āĀijæYřæĀŌæāũçt'fāŁāççŽDāŌşçRĚāĀĆ  
æfTāeĆīijNāIJlēālēçāijRæşČāĀijçlNāzRāy■īijNēfTāZđāĀijžāçēālēālēçāijRëgçædRāRŌçŽDēČlāLēç  
æIJĀāRŌæL'ĀæIJL'āĀijāijŽāIJlæIJĀēāũāsČçŽDēr■æşTëgDāLŽæŪzæşTāy■āRlāzūētūāIēāĀĆ

ārçōqāRŞā;æāijTçd'žçŽDæYřāyĀāyŁçŌĀ■TçŽDā;Nā■RīijNēĀŞā;ŞāyNéZ■ëgçædRāZlāRrāzççTlāIēā  
æfTāeĆīijNPythonēr■ēlĀæIJNēžnārsæYřēĀZēfGāyĀāyŁēĀŞā;ŞāyNéZ■ëgçædRāZlāŌzëgççGLççŽDāĀĆ  
āēČædIJā;āārfæ■d'æDşāĒt'ēūçīijNā;āāRfāzēēĀZēfGæşēçIJNPythonæžRçāAæŪGāzūGrammar/GrammaræI  
çIJNāŌNā;āāijŽāRŞçŌīijNēĀZēfGæL'NāLlæŪzāijRāŌZāōđçŌrāyĀāyŁëgçædRāZlāĒūāōđāijŽæIJL'ā;Lād'Žç

āĒūāy■āyĀāyŁāsĀēZŔārsæYřāŌČāznāy■ēČ;ēçncTlāzŌāNēĀRnāzžā;TāũēēĀŞā;ŞçŽDēr■æşTëgDāLŽāy

```
items ::= items ',' item  
        | item
```

āyžāžEèfZæāũāAŽīijNā;āāRrēČ;āijŽāČRāyNéIççēfZæāũā;ŁçTlā items() æŪzæşTīijŽ

```
def items(self):
    itemsval = self.items()
    if itemsval and self._accept(','):
        itemsval.append(self.item())
    else:
        itemsval = [ self.item() ]
```

āTrāyĀçŽĐēŮōécŸæŸrèŁŻäytæŮžæşŦæžæIJñäy■ēČ;ăüēă;IJiijŃăžŃăōđăyŁiijŃăōČaijŽăžğçŦşăyĂăy  
 äĖşăžŌēr■æşŦèğĐăĹŹæIJñèžñă;ăăRrèČ;ăžşăijŽççrăĹrăyĂăžŹæçŸæĹŃçŽĐēŮōécŸăĂČ  
 æŦŦæČiijŃă;ăăRrèČ;æČşşēēAŞăyŃéĹçēŁŻäytçōĂă■ŦæĹijēr■æşŦæŸŦăŦŦēăĹēŦŦă;Ůă;ŞiijŽ

```
expr ::= factor { ('+'| '-'| '*'| '/') factor }*

factor ::= '(' expression ')'
        | NUM
```

èŁŻäytēr■æşŦçIJŃäyĹăŌžæşăŦŦēēŮōécŸiijŃă;EæŸŦăōČă■Ŧăy■ēČ;ărşèğĹăĹŦæăĠăĠŦăŽăĹŹæŁŦçōŮ  
 æŦŦæČiijŃăēăĹēĹăijŦ "3 + 4 \* 5" äijŽăĹŮăĹŦŦ35ēĂŃäy■æŸŦæIJşæIJŽçŽĐ23.  
 âĹĖăijĂă;ŁçŦĹăĂĹexprăĂĹăŞŃăĂĹtermăĂĹèğĐăĹŹăŦŦăžēēŮŦăōČă■ççăŦçŽĐăüēă;IJăĂČ

äržăžŌăđ'■ăĹČçŽĐēr■æşŦiijŃă;ăæIJĂăē;æŸŦŦăĹŦăŦŦăşŦăyĹēğçăđŦăüēăĹŮăæŦŦæČŦPyParsingăĹŮēĂ  
 äyŃéĹæŸŦă;ŁçŦĹPLYăĹēēĠăĹŦăēăĹēĹăijŦăşČăĂijçĹŃăžŦçŽĐăžççăĂiijŽ

```
from ply.lex import lex
from ply.yacc import yacc

# Token list
tokens = [ 'NUM', 'PLUS', 'MINUS', 'TIMES', 'DIVIDE', 'LPAREN',
    ↪ 'RPAREN' ]
# Ignored characters
t_ignore = ' \t\n'
# Token specifications (as regexs)
t_PLUS = r'\+'
t_MINUS = r'\-'
t_TIMES = r'\*'
t_DIVIDE = r'\/'
t_LPAREN = r'\('
t_RPAREN = r'\)'

# Token processing functions
def t_NUM(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Error handler
def t_error(t):
    print('Bad character: {!r}'.format(t.value[0]))
    t.skip(1)
```

```

# Build the lexer
lexer = lex()

# Grammar rules and handler functions
def p_expr(p):
    '''
    expr : expr PLUS term
          | expr MINUS term
    '''
    if p[2] == '+':
        p[0] = p[1] + p[3]
    elif p[2] == '-':
        p[0] = p[1] - p[3]

def p_expr_term(p):
    '''
    expr : term
    '''
    p[0] = p[1]

def p_term(p):
    '''
    term : term TIMES factor
          | term DIVIDE factor
    '''
    if p[2] == '*':
        p[0] = p[1] * p[3]
    elif p[2] == '/':
        p[0] = p[1] / p[3]

def p_term_factor(p):
    '''
    term : factor
    '''
    p[0] = p[1]

def p_factor(p):
    '''
    factor : NUM
    '''
    p[0] = p[1]

def p_factor_group(p):
    '''
    factor : LPAREN expr RPAREN
    '''
    p[0] = p[2]

```

```
def p_error(p):
    print('Syntax error')
```

```
parser = yacc()
```

èŁŻäÿłçłŃăžŘäÿ■ījŇæŁ'ĂæIJL'ăžčçăĂéČ;ă;■ăžŎăÿĂăÿłæŕŤè;ČénŸçŽĐăŝĆăňăăĂĆă;ăăŔłéIJĂèĕAăÿ;  
èĂŇăôđéŽĚçŽĐèŁŘèăŇĕğčăđŘăŽłījŇæŎĚăŔŮăžđ'çŁŇç■Ł'ç■Ł'ăžŤăŝĆăŁłă;IJăűŝçžŔèćňăžŖăĜ;æŤŕăôđçŎ  
ăÿŇéłćæŸŕăÿĂăÿłæĂŎăűă;ŁçŤłă;ŮăŁŕçŽĐĕğčăđŘăŕžĕŝăçŽĐă;Ňă■ŔījŽ

```
>>> parser.parse('2')
2
>>> parser.parse('2+3')
5
>>> parser.parse('2+(3+4)*5')
37
>>>
```

ăĕĆăđIJă;ăăČŝăIJłă;ăçŽĐçijŮçłŇĕŁĜçłŇăÿ■ăłĕçĆžæŇŖŝæŁŸăŖŇăŁžæŁĂījŇçijŮăĖŽĕğčăđŘăŽłăŖŇă  
ăĖ■ăňăījŇăÿĂæIJŇçijŮĕŕŖăŖŝăŽłçŽĐăžĕçŝ■ăijŽăŇĚăŔňă;Łăđ'ŽăžŤăŝĆçŽĐçŔĖĕőžçŖĕŕĖăĂĆăÿ■ĕŁĜă;Łăđ'  
PythonĕĜăűŝçŽĐăŝăłăłŮăžŖăĂījă;ŮăŎžçIJŇăÿĂăÿŇăĂĆ

## 4.20 2.20 á■ŮĕŁĆă■ŮçĕăÿŝăÿŁçŽĐă■ŮçĕăÿŝăŖ■ă;IJ

### éŮőéćŸ

ă;ăăČŝăIJłă■ŮĕŁĆă■ŮçĕăÿŝăÿŁæŁĜĕăŇæŽőéĂŽçŽĐăŮĜăIJňăŖ■ă;IJ(æŕŤăĕČçğžéŽđ'ījŇæŔIJčŕ'ćă

### ĕğčăĖŝăĖŮžăăŁ

ă■ŮĕŁĆă■ŮçĕăÿŝăŖŇăűăžŖăŤŕăŇăĂăđ'ĝĕĆłăŁĖăŖŇăŮĜăIJňă■ŮçĕăÿŝăÿĂăűăçŽĐăĖĚç;őăŖ■ă;IJ

```
>>> data = b'Hello World'
>>> data[0:5]
b'Hello'
>>> data.startswith(b'Hello')
True
>>> data.split()
[b'Hello', b'World']
>>> data.replace(b'Hello', b'Hello Cruel')
b'Hello Cruel World'
>>>
```

èŁŽăžŽăŖ■ă;IJăŖŇăűăžŖăĖĂĆçŤłăžŎă■ŮĕŁĆăŤŕçžĐăĂĆăŕŤăĕĆījŽ

```
>>> data = bytearray(b'Hello World')
>>> data[0:5]
bytearray(b'Hello')
```

```
>>> data.startswith(b'Hello')
True
>>> data.split()
[bytearray(b'Hello'), bytearray(b'World')]
>>> data.replace(b'Hello', b'Hello Cruel')
bytearray(b'Hello Cruel World')
>>>
```

ā;āāRfāzēā;£çTīā■čāLZēālē;āijRāNzéĚ■ā■ŮèŁĆā■ŮçņēäyšijNā;EæYřæ■čāLZēālē;āijRæIJñèžnáĚ

```
>>>
>>> data = b'FOO:BAR, SPAM'
>>> import re
>>> re.split('[:,]', data)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "/usr/local/lib/python3.3/re.py", line 191, in split
return _compile(pattern, flags).split(string, maxsplit)
TypeError: can't use a string pattern on a bytes-like object
>>> re.split(b'[:,]', data) # Notice: pattern as bytes
[b'FOO', b'BAR', b'SPAM']
>>>
```

## èóíèőž

ād'ġād'ŽæTřæČĚāĒtāyNīijNāIJlæŮĠæIJñā■ŮçņēäyšāyŁçŽDæŞ■ā;IJāĪĠāRřçTīāžŌā■ŮèŁĆā■Ůçņēäyšā  
çDūēĀNīijNēfŽēĠNāžšæIJLāyĀāžZēIJĀēēAæşlæDRçŽDāy■āRŇçCzāĀĆēēŮāĒLīijNā■ŮèŁĆā■Ůçņēäyšç

```
>>> a = 'Hello World' # Text string
>>> a[0]
'H'
>>> a[1]
'e'
>>> b = b'Hello World' # Byte string
>>> b[0]
72
>>> b[1]
101
>>>
```

èĚŽçġ■èř■āzL'āyŁçŽDāNžāĹnāijŽāržāžŌād'DçŘĚĪcāRŠā■ŮèŁĆçŽDā■ŮçņēæTřæ■ōæIJL'ā;şāŞ■āĀĆ  
çñnāžNçCzīijNā■ŮèŁĆā■Ůçņēäyšāy■āijŽæRŘä;ZāyĀāyłç;ŌèġĆçŽDā■Ůçņēäyšēāłçd'zīijNāžšāy■èČ;ā

```
>>> s = b'Hello World'
>>> print(s)
b'Hello World' # Observe b'...'
>>> print(s.decode('ascii'))
Hello World
>>>
```

çşzäijijçŽĎĭjŇázšäy■ā■ŸāIJlázä;ŤĕĂĈçŤlázŎā■ŮĕĹĈā■ŮçñĕäyšçŽĎäĭjāijŔāŇŮæš■ā;IJĭjŽ

```
>>> b'%10s %10d %10.2f' % (b'ACME', 100, 490.1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for %: 'bytes' and 'tuple'
>>> b'{} {} {}'.format(b'ACME', 100, 490.1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'bytes' object has no attribute 'format'
>>>
```

āĕĈæĎIJā;āæĈşæĭjāijŔāŇŮā■ŮĕĹĈā■ŮçñĕäyšĭijŇä;āā;ŮāĔĹä;ĕçŤĹæāĠāĠĖĕçŽĎæŮĠæIJñā■Ůçñĕäyš

```
>>> '{:10s} {:10d} {:10.2f}'.format('ACME', 100, 490.1).encode(
↳ 'ascii')
b'ACME 100 490.10'
>>>
```

æIJĀāŔŎĕIJĀĕĕAæşĹæĎŔçŽĎæŸŕĭjŇä;ĕçŤĹā■ŮĕĹĈā■ŮçñĕäyşāŔĕĈ;āijŽæŤzāŔŸäyĀāzŽæš■ā;IJçŽĹ  
æŔĹæĈĭijŇāĕĈæĎIJā;āā;ĕçŤĹäyĀäyĹçijŮçāAäyžā■ŮĕĹĈçŽĎæŮĠäzūāŔ■ĭijŇĕĀŇäy■æŸŕäyĀäyĹæŽŏĕĀŽçŽ

```
>>> # Write a UTF-8 filename
>>> with open('jalape\xflo.txt', 'w') as f:
...     f.write('spicy')
...
>>> # Get a directory listing
>>> import os
>>> os.listdir('.') # Text string (names are decoded)
['jalapeÃso.txt']
>>> os.listdir(b'.') # Byte string (names left as bytes)
[b'jalapen\xcc\x83o.txt']
>>>
```

æşĹæĎŔä;Ňā■Ŕäy■çŽĎæIJĀāŔŎĕĈĹĹĕçzŽçŽŏā;ŤāŔ■āijäĕĀşäyĀäyĹā■ŮĕĹĈā■ŮçñĕäyşæŸŕæĀŎæāŮ  
āIJĹçŽŏā;Ťäy■çŽĎæŮĠäzūāŔ■āŇĕāŔŇāŎşāġŇçŽĎUTF-8çijŮçāAāĀĈ  
āŔĈĕĀĈ5.15āŔŔĕĹĈĕĖŮāŔŮæŽŦāĎŽæŮĠäzūāŔ■çŽyāĔşçŽĎāĖĔāŏzāĀĈ

æIJĀāŔŎæŔŔäyĀçĈzĭijŇäyĀāzŽçĹŇāzŔāŤŸäyžāzĖæŔŔā■ĠçĹŇāzŔæĹġĕāŇçŽĎĕĀşāžĕāijŽāĹ;āŔŤā  
āŕ;çŏāæš■ā;IJā■ŮĕĹĈā■ŮçñĕäyşçāŏāŏĎāijŽæŔĹæŮĠæIJñæŽŦāĹāĕŇŸæŤĹ(āŽāäyžāĎĎĈŔĖæŮĠæIJñāŽzæĹ  
ĕĔŹæāŮāĀŽĕĀŽäyāijŽāŕijĕĠŦĕĹäyŸæĹĈāzşçŽĎäzççāAāĀĈā;āāijŽçzŔäyŸāŔŤçŎŔā■ŮĕĹĈā■Ůçñĕäyşāzūāy  
āzūāyŤä;āĕŦŸä;ŮāĹŮāĹĹāĎĎĈŔĖæĹĀæIJĹçŽĎçijŮçāA/ĕġççāAæš■ā;IJāĀĈ  
āĹĕçŽĭĕŏŭijŇāĕĈæĎIJā;āāIJĹāĎĎĈŔĖæŮĠæIJñçŽĎĕŦĭijŇāŕşçŽŦæŎĕāIJĹçĹŇāzŔäy■ā;ĕçŤĹæŽŏĕĀŽçŽĎæŮĠ

## 5 çññäyĹçñāĭijŽæŤŕā■ŮæŮĕæIJşāŤŇæŮŮĕŮŦ

āIJĹPythonäy■æĹġĕāŇæŤŦæŤŕāŤŇæŦŏçĈzæŤŕççŽĎæŤŕā■ĕĔŔçŏŮæŮŮā;ĹçŏĀā■ŤçŽĎāĀĈ  
āŕ;çŏāāĕĈæ■ĎĭijŇāĕĈæĎIJā;āĕIJĀĕĕAæĹġĕāŇāĹĖæŤŕāĀAæŤŕçzĎæĹŮĕĀĖæŸŕæŮĕæIJşāŤŇæŮŮĕŮŦççŽĎ

æIJñçnäéŽĚäy■èóìèőžčŽĎārśæŸřèŁŻăžZăyžécŸăĂĆ

Contents:

## 5.1 3.1 æŤřā■ŮčŽĎāŽŽēĹ■ăžŤăĚě

éŮóécŸ

äĳăæČşāržæŧőčČžæŤřæŁ'ğèąNăŃĠăőŽčşĭăžęçŽĎēĹ■ăĚěèŁŖčőŮăĂĆ

èğčăĚşæŮžæąĹ

āržăžŎčőĂă■ŤčŽĎēĹ■ăĚěèŁŖčőŮĭĳNăĭŁçŤĹăĚĚçĭőčŽĎ  
ndigits) äĜĳæŤřă■şăŖřăĂĆærŤăęĆĭĳŽ

round(value,

```
>>> round(1.23, 1)
1.2
>>> round(1.27, 1)
1.3
>>> round(-1.27, 1)
-1.3
>>> round(1.25361, 3)
1.254
>>>
```

ăĳŞăyĂăyĹăĂĭĳăĹŽăęĭăĬĹăyđ'ăyĹèĭžçŤŃçŽĎăy■éŮŧçŽĎæŮŭăĂŽĭĳN  
äĜĳæŤřèŁŤăŽđęçžăőČæĬĂèŁŞçŽĎăĀŭæŤřăĂĆ äžşārśæŸřèt'ĭĳNārž1.5æĹŮèĂĚ2.5çŽĎēĹ■ăĚěèŁŖčőŮéČ

round

ăĭĳăçžŽ round() äĜĳæŤřçŽĎ ndigits ārČæŤřăŖřăžžæŸřèt'şæŤřĭĳNèŁŽçğ■æČĚăĚăyNĭĳN  
ēĹ■ăĚěèŁŖčőŮăĭĳŽăĬçŤĹăĬă■Āăĭ■ăĂăçŽĭăĭ■ăĂă■Čăĭ■ç■ĹăyĹéĹčăĂĆærŤăęĆĭĳŽ

```
>>> a = 1627731
>>> round(a, -1)
1627730
>>> round(a, -2)
1627700
>>> round(a, -3)
1628000
>>>
```

èóìèőž

äy■èęĂārĚēĹ■ăĚěăŞNăăĭĳăĭŖăŃŮēĭŞăĜžæŖđæŭŭæŭĚăžĚăĂĆ  
ăęČăđĬĂăĭçŽĎçŽőçŽĎăŖĹæŸřčőĂă■ŤčŽĎēĭŞăĜžăyĂăőŽăőĭăžęçŽĎæŤřĭĳNăĭăăy■éĬĂèęĂăĭŁçŤĹ  
round() äĜĳæŤřăĂĆ èĂNăžĚăžĚăŖĹēĬĂăęĂăĬăăĭĳăĭŖăŃŮçŽĎæŮŭăĂŽæŃĠăőŽčşĭăžęă■şăŖřăĂĆærŤ

```
>>> x = 1.23456
>>> format(x, '0.2f')
'1.23'
>>> format(x, '0.3f')
'1.235'
>>> 'value is {:.3f}'.format(x)
'value is 1.235'
>>>
```

āŕŅæūīījŅäy■ēēAērTçĬĀāŌzèĹ■āĔĖæŧōçĆzāĀijæĭēāĀĬāġōæ■cāĀĭēāĭēĬcäyŁçIJŅètūæĭēæ■čçāōçŽĎéŮ

```
>>> a = 2.1
>>> b = 4.2
>>> c = a + b
>>> c
6.3000000000000001
>>> c = round(c, 2) # "Fix" result (???)
>>> c
6.3
>>>
```

āržāžŌād'ġād'ŽæTŕä;ŁçTĭāĹŕæŧōçĆzçŽĎĭĬŅāžRīījŅæšæIJL'āŁĔēēAāžšäy■æŌĭē■ŘēŁŽæūāAŽāĀĆ  
 āŕ;čōāĀĬĭēōāçŌŮçŽĎæŮūāĀŽāījŽæIJL'äyĀçĆzçĆzārRçŽĎèŕŕāūōīījŅā;EæŸŕēŁŽāžZārRçŽĎèŕŕāūōæŸŕēČ;ē  
 āēČæĎIJäy■ēČ;āĔAēōyēŁŽæūçŽĎārRèŕŕāūō(æŕTāēČæŮĹ'ārĹāĹŕēĠSēĎ■ēēEāšš)īījŅēČčāžĹĀŕšā;ŮēĀČēŽ  
 decimal æĭāĭŮāžEīījŅäyŅäyĀēŁČæĹSāžñāījŽèŕēçzEēōĭēōžāĀĆ

## 5.2 3.2 æĹ'ġēāŅçš;çāōçŽĎæŧōçĆzæTŕèĤŔçóŮ

### éŮōēčŸ

ā;āēIJĀēēAāržæŧōçĆzæTŕæĹ'ġēāŅçš;çāōçŽĎèōāçŌŮæŠ■ā;IJīījŅāžūāyTäy■āyŅæIJŽæIJL'āžžā;TārRèŕŕ

### ēġcāEşæŮzæāĹ

æŧōçĆzæTŕçŽĎäyĀäyĭæŽōēA■ēŮōēčŸæŸŕāōČāžñāžūāy■ēČ;çš;çāōçŽĎēāĭçĎ'žā■AēŁZāĹūæTŕāĀĆ  
 āžūāyTīījŅā■şā;ŁæŸŕæIJĀçŌĀā■TçŽĎæTŕā■ēēŁŔçŌŮāžšāījŽāžġçTşārRçŽĎèŕŕāūōīījŅæŕTāēČīījŽ

```
>>> a = 4.2
>>> b = 2.1
>>> a + b
6.3000000000000001
>>> (a + b) == 6.3
False
>>>
```

ēŁŽāžŽēTŽèŕŕæŸŕçTşāžTşāšČCPUāšŅĬEEE 754æāĠāĠEēĀŽēŁĠēĠāūšçŽĎæŧōçĆzā■Tā;■āŌzæĹ'ġēāŅ  
 çTşāžŌPythonçŽĎæŧōçĆzæTŕæ■ōçšzāĎŅā;ŁçTĭāžTşāšČēāĭçĎ'žā■ŸāČĭæTŕæ■ōīījŅāŽæ■Ď'ā;āæšāĹĎæşTşāŌ



æƿƿædIJä;äæČšæŽt'ăŁăçş;çăo(ăžűèČ;ăoźăf■ăyĂăoŽçŽDæĂgèČ;æ■şèĂŮ)iiĳNă;ăăŔŕăžèä;ŁçŦĬ  
decimal æłăăĬŮiiĳŽ

```
>>> from decimal import Decimal
>>> a = Decimal('4.2')
>>> b = Decimal('2.1')
>>> a + b
Decimal('6.3')
>>> print(a + b)
6.3
>>> (a + b) == Decimal('6.3')
True
```

ăĬĬçIJNĕtŭæĬĕriĳNăyŁĕĬçŽDăžçăĂăĕ;ăČŔæIJL'çČžăĕĞæĂĥiiĳNăŕŦăĕČæĬSăžŋçŦĬă■ŮçĥăyşæĬĕĕăĬç  
çDűĕĂNĕiiĳNDecimalăŕžĕşăăiĳŽăČŔæŽŕĕĂŽăŦŕçČžæŦŕăyĂăăŭçŽDăŭĕă;IJ(æŦŕæNĂæĬĂæIJL'çŽDăyŷç'  
ăĕČædIJä;äæĬŞă■ăŕăŔČăžŋæĬŮĕĂĕăIJĬă■ŮçĥăyşæăiĳăiĳŔăNŮăĜ;æŦŕăy■ă;ŁçŦĬăŕăŔČăžŋiiĳNçIJNĕtŭæĬĕĕŭşă

decimal æłăăĬŮçŽDăyĂăyĬăyžĕĕĂçĬŕă;ĂæŦŕăĕĂĕŕăyă;ăăŖĝăĬŮĕŕăçŕŮçŽDăŕŔăyĂæŮžĕĬçiiĳNăN  
ăyžăžĕĕŦŽăăŭăĂŽiiĳNă;ăăĕĬă;ŮăĬŽăžăyĂăyĬæIJăIJŕăyĬăyNăŮĜăžŭæŽt'æŦžăŕăŔČŽDĕŕç;ĭŕiiĳNăŕŦăĕ

```
>>> from decimal import localcontext
>>> a = Decimal('1.3')
>>> b = Decimal('1.7')
>>> print(a / b)
0.7647058823529411764705882353
>>> with localcontext() as ctx:
...     ctx.prec = 3
...     print(a / b)
...
0.765
>>> with localcontext() as ctx:
...     ctx.prec = 50
...     print(a / b)
...
0.76470588235294117647058823529411764705882352941176
>>>
```

## ĕŕĕŕă

decimal æłăăĬŮăŕăŔčŖăžĕĬBMçŽDăĂĬĕĂŽçŦĬăŕŔæŦŕĕŁŔçŕŮĕĝDĕNČăĂĬăĂČăy■çŦĬĕŕt'iiĳNăIJL'ă;

PythonæŮŕæĬNăiĳŽăĂ;ăŔŞăžŖă;ŁçŦĬdecimal æłăăĬŮæĬĕăd'DçŔĕăŦŕçČžæŦŕçŽDçş;çăŕŕĕŁŔçŕŮăĂ  
çDűĕĂNĕiiĳNăĕĬçŔĕĕĝă;ăçŽDăžŦçŦĬçĬNăžŔçŽŕçŽDæŦŕĕĬăyŷĕĞ■ĕĕĂçŽDăĂČ  
ăĕČædIJä;äæŦŕăIJăĂŽçĝŞă■ĕĕŕăçŕŮŮăĬŮăŭĕĬNĕçĕăşşçŽDĕŕăçŕŮăĂĂçŦĬĕDŞçžŦăŽ;iiĳNăĬŮĕĂĕăŦŕç  
ĕČăžĬă;ŁçŦĬăŽŕĕĂŽçŽDăŦŕçČžçşădNăŦŕăŕŦĕ;ČæŽŕĕĂ■çŽDăĂŽăşŦăĂČ  
ăĕŮăy■ăyĂăyĬăŖăŖăŦăŦæŦŕiiĳNăIJĬçIJşăŕăŕăŦŮçŦNăy■ă;ĬăŔŕăiĳŽĕĕĂăşČçş;çăŕăŔăŔăŦăŽŕĕĂŽăŦŕçČžæŦŕĕČ;æ  
ăŽăă■d'iiĳNĕŕăçŕŮŮĕĝĜĬNăy■çŽDĕČăžĬăyĂçČžçČžçŽDĕŕŕăŭŕăŦŕĕçŋăĕĂĕŕăçŦŕĕçŽDăĂČ  
çŋăžNĕČăŕăşæŦŕiiĳNăŖŮçŦŦçŽDăŦŕçČžæŦŕĕŕăçŕŮŮĕĕĂăŦŋçŽDăd'Ž-  
æIJL'æŮŮăĂŽă;ăăIJăĬĕĝăNăd'ĝĕĞŔĕĕŔçŕŮŮçŽDăŮŮăĂŽĕĂşăžĕăžşæŦŕĕĬăyŷĕĞ■ĕĕĂçŽDăĂČ

āṣä;ŁæĆæṑ'īijNä;āṑ't'äyṑēČ;āōNāĒlāŁ;çTēērrāũōāĀĆæTṙāṑēāōūēŁsāžEāđ'gēGRæŪūēŪt'āŌžčāTçl  
ä;āāžŠā;ŪæšlāĎRāyNāGRæšTāLāēZđ'āžēāRŁād'gæTṙāŠNārRæTṙçŽĐāŁāāLĒēŁRçōŪæL'ĀāyçælēčŽĐā;šā

```
>>> nums = [1.23e+18, 1, -1.23e+18]
>>> sum(nums) # Notice how 1 disappears
0.0
>>>
```

äyLēlčçŽĐēTŽērrāRāžēāL'çTl`math.fsum()` æL'ĀæRŘä;ŽçŽĐæŽt'çš;çāōēōāçōŪēČ;āŁZælēēğčāE

```
>>> import math
>>> math.fsum(nums)
1.0
>>>
```

çĐūēĀNīijNāržāžŌāĒūāžŪçŽĐçōŪæšTīijNä;āāžTēēāžTçzEçāTçl'ūāōČāžūçRĒēğčāōČçŽĐērrāũōāžçç  
æĀžçŽĐælēērt'īijN decimal ælāāIŪäyžēēAçTlāIJlæūL'āRŁāLrēGŠēđṑçŽĐēčEāššāĀĆ  
āIJlēŁŽçšzçlNāžRāyṑīijNāŠlæĀTæYřāyĀçČzārRārRçŽĐērrāũōāIJlēōāçōŪēŁGçlNāyṑēTŠāžūēČ;æYřāyṑāĒA  
āZāæṑ'īijN decimal ælāāIŪäyžēēğčāEšēŁŽçšzēŪōēčYæRŘä;ZāžEæŪzæšTāĀĆ  
ā;šPythonāŠNæTṙæṑōāžŠæL'Šāžđ'ēAšçŽĐæŪūāĀZāžšēĀŽāyāijŽēAĞāLř Decimal  
āržēšāijNāžūāyTīijNēĀŽāyāžšæYřāIJlād'ĐçRĒēGŠēđṑæTṙæṑōçŽĐæŪūāĀZāĀĆ

## 5.3 3.3 æTṙāṑçŽĐæāijāijRāNŪē;ŠāGž

### éŪōēčY

ä;āēIJĀēçAārEæTṙāṑæāijāijRāNŪāRŌē;ŠāGžīijNāžūæŌğāLūæTṙāṑçŽĐä;ṑæTṙāĀAāržē;RāĀAṑṑČ

### ēğčāEšæŪzæāŁ

æāijāijRāNŪē;ŠāGžāṑTäyŁæTṙāṑçŽĐæŪūāĀZīijNāRāžēä;ŁçTlāĒēç;ōçŽĐ  
`format()` āĞ;æTṙīijNærTāēČīijŽ

```
>>> x = 1234.56789

>>> # Two decimal places of accuracy
>>> format(x, '0.2f')
'1234.57'

>>> # Right justified in 10 chars, one-digit accuracy
>>> format(x, '>10.1f')
'      1234.6'

>>> # Left justified
>>> format(x, '<10.1f')
'1234.6      '
```

```
>>> # Centered
>>> format(x, '^10.1f')
' 1234.6 '
```

```
>>> # Inclusion of thousands separator
>>> format(x, ',')
'1,234.56789'
>>> format(x, '0,.1f')
'1,234.6'
>>>
```

æĈædIJä;äæĈšä;ġçTlæŃĜæTṛèõræşTṛijŃârEġæTzæĹŖæĹŨèĂĖĖ(ârŨăEşăzŌæŃĜæTṛèçŞăĠzçŽDă

```
>>> format(x, 'e')
'1.234568e+03'
>>> format(x, '0.2E')
'1.23E+03'
>>>
```

ârŃæŨăæŃĜăŏŽăŏ;ăžăăŃŃçş;ăžççŽDăyĂēĹŃă;ćăijŖæŸŕ  
 '[<>^]?width[,]?(.digits)?' iijŃ äĖŭäy width  
 äŃŃ digits äyžæTt æTṛijŃṛijşăzçēăĹăŖéĂĹéĈĹăĹEăĂĈ  
 âŃŃæŭçŽDæăijăijŖăzşēćŋçTlăIJlăŭçŋēäyşçŽD format() æŨzæşTäyăăĂĈærTăēĈijŽ

```
>>> 'The value is {:0,.2f}'.format(x)
'The value is 1,234.57'
>>>
```

## ēōlēōž

æTṛăŭæăijăijŖăŃŨēçŞăĠžēĂŽăyŷæŸŕæŕTēçĈçŏĂăŃçŽDăĂĈăyĹēĹćæijTçd'žçŽDæĹĂæIJŖăŃŃæŨă  
 decimal æĹăăĹŨăyçŽD Decimal æTṛăŭăŕžēsăăĂĈ

ă;ŞæŃĜăŏŽæTṛăŭçŽDă;æTṛăŖŌṛijŃçzŞædIJăĂijăijŽæăžæŏ round()  
 âĠ;æTṛăŖŃŃæŭçŽDēğDăĹŽēġZèqŃăŽZēĹăžTăĖēăŖŌēġTăZdăĂĈærTăēĈijŽ

```
>>> x
1234.56789
>>> format(x, '0.1f')
'1234.6'
>>> format(-x, '0.1f')
'-1234.6'
>>>
```

ăŃĖăŖŃăăĈă;çŋççŽDæăijăijŖăŃŨēŭşæIJŃăIJŖăŃŨăşşæIJĹăăĖşçşzăĂĈ  
 æĈædIJä;ăēIJĂēēĂæăžæŏăIJŖăŃŃæĹēæŸçd'žăăĈă;çŋçēijŃă;ăēIJĂēēĂēĠăŭsăŌzèŕĈæşēäyŃ  
 locale æĹăăĹŨăyçŽDăĠ;æTṛăžEăĂĈ ä;ăăŖŃŃæŭăžşăŖŕăžēă;ġçTlăŭçŋēäyşçŽD  
 translate() æŨzæşTăĹēăžd'æăăăĈă;çŋçēăĂĈærTăēĈijŽ

```
>>> swap_separators = { ord('.'): ',', ord(','): '.' }
>>> format(x, ',').translate(swap_separators)
'1.234,56789'
>>>
```

ǎIǎĬŁăďŽPythonăžččăĂăy■ăiŷŻçIJNăĹră;£çTĭ%æİěæăiŷăiŷRăNŨæTŗă■ŮçZĐiiŷNærTăeĆiiŷ

```
>>> '%0.2f' % x
'1234.57'
>>> '%10.1f' % x
'      1234.6'
>>> '%-10.1f' % x
'1234.6      '
>>>
```

èfZçg■æäijäijRāNŨæÚzæşTăzşæYřāRřeaŇčŽDñijÑäy■èŁGærTæŽt'āŁääĚLèfZçŽD  
format() èeAũöäÿĂçCžãĀĆ ærTăeĆñijNāIJlä;ŁçTl'æ%š■ä;IjçņæäijäijRāNŨæTřā■ŮčŽDæUũāĂZñijÑäy

### 5.4 3.4 äžŃăĖŋă■AǎĖ■èŁZǎLúæŦt'æŦř

éŮőécŸ

ä:äéIJǻèèAè:ñæ■cæLŨèĀĒè:ŞǻĠzǻ:£çTǻǻzNè£ZǻLŭiijNǻĒĒè£ZǻLŭæLŨǻ■AǻĒ■è£ZǻLŭèǻlçd'žçZǻDǻT

èġčǎẸșæŮźæąŁ

äyžäZĖårĖæTt æTřejñæ■cäyžäZÑefZǎLúǎĀAǎĖñefZǎLúǎLŮǎ■AǎĖ■efZǎLúǎŽDǎŮĜæIñäyřijñ  
ǎRǎžčǎLEǎLñǎ;řčTl bin() , oct() æLŮ hex() ǎĜ;æTřijŽ

```
>>> x = 1234
>>> bin(x)
'0b10011010010'
>>> oct(x)
'0o2322'
>>> hex(x)
'0x4d2'
>>>
```

âRëad' ŪijŃæĈæđIjä;äy■æČšè;ŠåĞž      0b      ,      0o      æŁÛëĂĚ      0x  
čŽďÄL'■cijĂčŽďërliijŃăRřäzëä;£çTÍ format ()    âĜ;ætŘrăĂĈærTăeĆiiJŽ

```
>>> format(x, 'b')
'10011010010'
>>> format(x, 'o')
'2322'
>>> format(x, 'x')
'4d2'
>>>
```

æTt'æTṛæYṛæIJL'çñęąRũçŽDīijNæL'ÄäzëåĊæđIJä;ääIJlâd'ĐçŘĚèť §æTṛçŽDèřIijNè;ŠăĠžçzŠæđIJäij

```
>>> x = -1234
>>> format(x, 'b')
'-10011010010'
>>> format(x, 'x')
'-4d2'
>>>
```

æĊæđIJä;æĊšăžğçTšăyÄäyĽæUăçñęąRũăĀijīijNă;ăéIJĀëĖAăćđăLăäyÄäyĽæŃĠçđ'žæIJĀăd'gă;ĖTĽăž

```
>>> x = -1234
>>> format(2**32 + x, 'b')
'1111111111111111111111111111111101100101110'
>>> format(2**32 + x, 'x')
'fffffb2e'
>>>
```

äyžăžEäzëäy■ăRŃçŽDèřŽăĽŭë;ñæ■ćæTt'æTṛă■ŮçñęäyšīijŃçôĀă■TçŽDă;ĤçTĽăyęæIJL'èřŽăĽŭçŽD  
int()ăĠjæTṛă■šăRřīijŽ

```
>>> int('4d2', 16)
1234
>>> int('10011010010', 2)
1234
>>>
```

## ěőĽěőž

ăđ'găđ'ŽæTṛæĈĚăĖtăyŃăđ'ĐçŘĚăžŃëřŽăĽŭăÄăĖñëřŽăĽŭăŠŃă■AăĖ■ëřŽăĽŭæTt'æTṛæYṛă;ĽçôĀă  
ăRĽëĖAęőřă;ŘëřŽăžŽë;ñæ■ćăśđăžŌæTt'æTṛăŠŃăĖŭăřžăžTçŽDæŮĠæIJñëăĽçđ'žăžŃëŮťçŽDë;ñæ■ćă■šăRřă

æIJĀăRŎīijNă;ĤçTĽăĖñëřŽăĽŭçŽDċĽăžRăŠŸæIJL'ăyĀçĈzéIJĀëĖAăşĽăĐRăyŃăĀĈ  
PythonăŃĠăőŽăĖñëřŽăĽŭæTṛçŽDèř■ăşTēũşăĖŭăžŮër■ëĽĀċĽæIJL'ăy■ăRŃăĀĈăřTăęĈīijŃăęĈæđIJä;ăăĈ

```
>>> import os
>>> os.chmod('script.py', 0755)
File "<stdin>", line 1
    os.chmod('script.py', 0755)
    ^
SyntaxError: invalid token
>>>
```

éIJĀçăőăřĽăĖñëřŽăĽŭæTṛçŽDăĽ■çijĀæYř 0o īijŃăřśăĈRăyŃëĽçëřŽăăŭīijŽ

```
>>> os.chmod('script.py', 0o755)
>>>
```

## 5.5 3.5 ā■ŪēŁĆāŁřāđ'ġæŦŦ'æŦŦçŽĐæŁ'ŠāŃĒäyŌèġcāŃĒ

### ēŪōēćŸ

äĵāæIJL'äyÄäyġā■ŪēŁĆā■ŪçņęäyśāzŭæČšārĒāōČèġcāŌŃæŁŔäyÄäyġæŦŦ'æŦŦřāĀĆæŁŪēĀĒiĵŃäĵāéIJĀ

### èġcāĒşæŪzæąŁ

āĀĠēōĵ;äĵčŽĐĠŃāžŔéIJĀēēĀāđ'ĐçŔĒäyÄäyġæŃēæIJL'128äĵ■ēŦŦçŽĐ16äyġāĒĒçŦŦ'äçŽĐā■ŪēŁĆā■Ūç

```
data = b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
```

äyžāžĒārĒbytesèġcāđŔäyžæŦŦ'æŦŦiĵŃäĵçŦŦĪ int.from\_bytes()  
æŪzæşŦŦiĵŃāžŭāČŔäyŃēĪcēŁZæāŭæŃĠāōŽā■ŪēŁĆéāžāžŔiĵŽ

```
>>> len(data)
16
>>> int.from_bytes(data, 'little')
69120565665751139577663547927094891008
>>> int.from_bytes(data, 'big')
94522842520747284487117727783387188
>>>
```

äyžāžĒārĒäyÄäyġāđ'ġæŦŦ'æŦŦŕēĵŃæ■cäyžäyÄäyġā■ŪēŁĆā■ŪçņęäyśiĵŃäĵçŦŦĪ int.to\_bytes()  
to\_bytes() æŪzæşŦŦiĵŃāžŭāČŔäyŃēĪcēŁZæāŭæŃĠāōŽā■ŪēŁĆæŦŦřāŃŃā■ŪēŁĆéāžāžŔiĵŽ

```
>>> x = 94522842520747284487117727783387188
>>> x.to_bytes(16, 'big')
b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
>>> x.to_bytes(16, 'little')
b'4\x00#\x00\x01\xef\xcd\x00\xab\x90x\x00V4\x12\x00'
>>>
```

### èŌĪēōž

āđ'ġæŦŦ'æŦŦřāŃŃā■ŪēŁĆā■ŪçņęäyśāzŃēŪŦ'çŽĐēĵŃæ■cæŞ■äĵIJāžŭäy■äyŷēġĀāĀĆ  
çĐŭēĀŃiĵŃäIJäyÄäžŽāžŦçŦŦēcĒāşşæIJL'æŪŭāĀžāžşäĵŽāĠžçŌŕiĵŃæŦŦāēČārĒçāĀā■ēæŁŪēĀĒçĴçzIJā  
äĵŃāēČiĵŃiĴv6çĴçzIJāIJřāĪĀäĵçŦŦĪäyÄäyġ128äĵ■çŽĐæŦŦ'æŦŦŕēāġçđ'žāĀĆ  
āēČæđIJāĵāēēĀāžŌäyÄäyġæŦŦŕæ■ōēŕāĵŦäy■æŔŔāŔŪēŁZæāŭçŽĐāĪiĵçŽĐæŪŭāĀžŕiĵŃäĵāārşäĵŽēĪcāržēŁZ

äĵIJäyžäyĀçġ■æŁēāžcæŪzæąŁiĵŃäĵāārŕēČĴæČşäĵçŦŦĪ6.11ārŔēŁĆäy■æŁ'ÄäžŃçz■çŽĐ  
struct æĪāāĪŪæĪēēġcāŌŃā■ŪēŁĆāĀĆ èŁZæāŭāžşēāŃäĵŪēĀžŕiĵŃäy■ēŁĠāĪŦçŦŦĪ  
struct æĪāāĪŪæĪēēġcāŌŃāržāžŌæŦŦ'æŦŦçŽĐāđ'ġārŔæŸŕæIJL'éžŔāĪŭçŽĐāĀĆ  
āžāæ■đ'ŕiĵŃäĵāārŕēČĴæČşēēġcāŌŃāđ'Žäyġā■ŪēŁĆäyśāzŭārĒçşşæđIJāŔĪāžŭäyžæIJāçžŁçŽĐçzşæđIJiĵŃārş

```
>>> data
b'\x00\x124V\x00x\x90\xab\x00\xcd\xef\x01\x00#\x004'
>>> import struct
```

```
>>> hi, lo = struct.unpack('>QQ', data)
>>> (hi << 64) + lo
94522842520747284487117727783387188
>>>
```

ā■ŪēŁĆężāžŘèĎĀĹŽ(littleēĹŪbig)äzĚäzĚæŇĠăŏŽāžĚæđĎāžžæŦŦ æŦŦæŪŭçŽĎā■ŪēŁĆçŽĎäĭŎäĭ■  
æĹŚāžñāžŎäyŇéĬçşĭ;ăĤĈæđĎéĀăçŽĎ16ēĤZăĹŪæŦŦçŽĎēăĭçđ'žäy■ăŦŕăžēăĭĹăŏžæŸŞçŽĎĎĭJŇăĠžæĭēĭjŽ

```
>>> x = 0x01020304
>>> x.to_bytes(4, 'big')
b'\x01\x02\x03\x04'
>>> x.to_bytes(4, 'little')
b'\x04\x03\x02\x01'
>>>
```

ăĉĈæđĬJăĭăĕŦŦçĬĀăŦĚäyĀăyĹæŦŦ æŦŦæĹ'ŞăŇĚäyžă■ŪēŁĈă■ŪçņęäyşĭĭjŇéĈçăžĹăŏĈăŕşäy■ăŦĹéĀĈăžĚ  
ăĉĈæđĬĬēĬĀăĉĀçŽĎĕŦĭĭjŇăĭăăŦŕăžēăĭ;ĤçŦĬ int.bit\_length()  
æŪžæşŦŦăĭăĚşăŏŽēĬJĀăĉĀăđ'ŽăŦŦă■ŪēŁĈăĭ■ăĭă■ŸăĈĭēĤŽăyĹăĀĭjăĀĈ

```
>>> x = 523 ** 23
>>> x
335381300113661875107536852714019056160355655333978849017944067
>>> x.to_bytes(16, 'little')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
OverflowError: int too big to convert
>>> x.bit_length()
208
>>> nbytes, rem = divmod(x.bit_length(), 8)
>>> if rem:
...     nbytes += 1
...
>>>
>>> x.to_bytes(nbytes, 'little')
b'\x03X\xf1\x82iT\x96\xac\xc7c\x16\xf3\xb9\xcf...\xd0'
>>>
```

## 5.6 3.6 āđ■æŦŦçŽĎæŦŦă■ēēĤŦçŏŪ

### éŪŏécŸ

ăĭăăĚŽçŽĎæĬJĀăŦŦçŽĎĎĭŞçžĬĚŏđ'ēŦĀăŪžæăĹăžççăĀăĚĀĠŦŕăžĚäyĀăyĹēŽĭēĉŸĭĭjŇăžŭäyŦăĭăăŦŦŕăy.  
ăĚ■æĹŪēĀĚæŸŦăĭăäzĚäzĚēĬJĀăĉĀăĭ;ĤçŦĬăđ'■æŦŦæĭăĚĹġēăŇăyĀăžŽēŏăçŏŪăŞ■ăĭJăĀĈ

## èġċàEşæŪzæąŁ

ād'■æTŗăRřăzēcŤlă;ŁçŤlăĠjæTŗ complex(real, imag)  
æŁŪěĂĖæŸřăŸæIJL'ăŘŎçijĂjçŽĐæŁçĆzæTŗæIěæŃĠăŎŽăĂĆæŤăċŤiijŽ

```
>>> a = complex(2, 4)
>>> b = 3 - 5j
>>> a
(2+4j)
>>> b
(3-5j)
>>>
```

ărzăžŤçŽĐăôđēĈlăĂAēŽŽēĈlăSŃăĖŖē;■ād'■æTŗăRřăzēăĠLăôžæŸŞçŽĐēŎăăRŪăĂĆăŖŝăĈRăyŃéIćēŁZ

```
>>> a.real
2.0
>>> a.imag
4.0
>>> a.conjugate()
(2-4j)
>>>
```

ăRēăđ'ŪiijŃæL'ĂæIJL'ăŸŸēġAçŽĐæTŗă■ēēŁŖçŎŪēĈ;ăRřăzēăŭēă;IJiijŽ

```
>>> a + b
(5-1j)
>>> a * b
(26+2j)
>>> a / b
(-0.4117647058823529+0.6470588235294118j)
>>> abs(a)
4.47213595499958
>>>
```

ăċĆăđIJēċAæL'ġēăŃăĖŭăžŪçŽĐăđ'■æTŗăĠjæTŗăŤăċĆæ■ċăijēăĂĂă;ŽăijēæŁŪăžşæŪzæăžiiijŃă;ŁçŤlă  
cmath æIăăIŪiijŽ

```
>>> import cmath
>>> cmath.sin(a)
(24.83130584894638-11.356612711218174j)
>>> cmath.cos(a)
(-11.36423470640106-24.814651485634187j)
>>> cmath.exp(a)
(-4.829809383269385-5.5920560936409816j)
>>>
```



## èõíèõž

Pythonäy■åđ' ġéČláLĚäyŎæTřā■ęçŽyăĚşçŽĎæłāłŮéČjèČjāđ' ĎçŘĚāđ' ■æTřāĀĆ  
ærTăęĆăęĆæđIJă;ăă;ŁçTl numpy iijŇăŔřăžěăŁăŏžæŸŞçŽĎæđĎéĀăăŸĀăŸłāđ' ■æTřæTřçžĎăžŭăIJlèŁŽăŸłæ

```
>>> import numpy as np
>>> a = np.array([2+3j, 4+5j, 6-7j, 8+9j])
>>> a
array([ 2.+3.j, 4.+5.j, 6.-7.j, 8.+9.j])
>>> a + 2
array([ 4.+3.j, 6.+5.j, 8.-7.j, 10.+9.j])
>>> np.sin(a)
array([ 9.15449915 -4.16890696j, -56.16227422 -48.50245524j,
       -153.20827755-526.47684926j, 4008.42651446-589.49948373j])
>>>
```

PythonçŽĎæăĠăĠĚæTřā■ęăĠjæTřçăŏăŏđæČĚăĚtăŸŇăžŭăŸ■èČjăžġçTşăđ' ■æTřăĀijīijŇăŽăæ■đ' äjăçŽl

```
>>> import math
>>> math.sqrt(-1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: math domain error
>>>
```

ăęĆæđIJă;ăæČşçTşæŁŔăŸĀăŸłāđ' ■æTřēŁTăŽđçzŞæđIJīijŇă;ăăŁĚéąžæŸŁçđ' žçŽĎă;ŁçTl  
cmath æłăłŮŮiijŇăŁŮēĀĚăIJlæşŔăŸłæTřæŇĀăđ' ■æTřçŽĎăžŞăŸ■ăčŕæŸŎăđ' ■æTřçşzăđŇçŽĎă;ŁçTlăĀĆă

```
>>> import cmath
>>> cmath.sqrt(-1)
1j
>>>
```

## 5.7 3.7 æŮăçł' ůăđ' ġăŸŎNaN

### éŮŏécŸ

ăjăăČşăŁŽăžžæŁŮætŇērTă■čæŮăçł' ůăĀĀèt' şæŮăçł' ůăŁŮNaN(éíđæTřă■Ů)çŽĎætŏçĆzæTřăĀĆ

### èġčăĚşæŮzæăŁ

PythonăžŭăşşæIJL' çŁ' žæŏŁçŽĎēr■æşTăēlēăłçđ' žèŁŽăžŽçŁ' žæŏŁçŽĎætŏçĆzăĀijīijŇă;ĚæŸŕăŔřăžěă;Ł  
float() ælēăŁŽăžžăŏČăžŇăĀĆærTăęĆīijŽ

```
>>> a = float('inf')
>>> b = float('-inf')
>>> c = float('nan')
>>> a
```

```
inf
>>> b
-inf
>>> c
nan
>>>
```

äyžāžĒætNērTēfZāžZāĀijçŽDā■YāIJīijNā;£çTÍ math.isinf() āšŃ math.isnan() āĠ;æTřāĀĆærTāęĆiijŽ

```
>>> math.isinf(a)
True
>>> math.isnan(c)
True
>>>
```

## ěőléőž

æČšāžĒęčæŽt'ād'Žè£ZāžZçL'zæōŁætōçCzāĀijçŽDāŁæAřīijNāRřāžčāRCèĀČIEEE  
754ěğDèNČāĀC çDūēĀNīijNāžšæIJL'äyĀāžZāIJřæŮzéIJĀēęAä;ăçL'zāŁnæşŁæĎŘīijNçL'zāŁnæYřèuşærTè;ç  
æŮăçl'ŭăd'ğæTřāIJæL'ğèąNæTřā■ęèőăçőŮçŽDæŮŭāĀŽăijŽăijăæŠ■īijNærTāęĆiijŽ

```
>>> a = float('inf')
>>> a + 45
inf
>>> a * 10
inf
>>> 10 / a
0.0
>>>
```

ä;ĒæYřæIJL'ăžZæŞ■ă;IJæŮŭæIJăőŽăžL'çŽDăžŭăijŽè£TăŽďäyĀăy†NaNçzŞæđIJăĀĆærTāęĆiijŽ

```
>>> a = float('inf')
>>> a/a
nan
>>> b = float('-inf')
>>> a + b
nan
>>>
```

NaNāĀijăijŽăIJĹæL'ĀæIJL'æŞ■ă;IJăy■ăijăæŠ■īijNèĀŃăy■ăijŽăžğçTşăijCăyyāĀĆærTāęĆiijŽ

```
>>> c = float('nan')
>>> c + 23
nan
>>> c / 2
nan
```

```
>>> c * 2
nan
>>> math.sqrt(c)
nan
>>>
```

NaNaĀijçŽDäyĀäyŁçL'žāŁŋçŽDāIJræŪzæŪūāōČāznāzNēŪt'çŽDærTè;ČæŠ■ā;IJæĀzæŸrèŁTāZdFalse

```
>>> c = float('nan')
>>> d = float('nan')
>>> c == d
False
>>> c is d
False
>>>
```

çTšāžŌēŁZāyŁāŌŠāZārijNætNērTāyĀäyNaNaĀijā;ŪāTřāyĀāōL'āĒłçŽDæŪzæšTārśæŸřā;ŁçTł  
math.isnan() ĩijNāzšārśæŸřāyŁēłçæijTčd'žçŽDēČčæūāĀČ

æIJL'æŪūāĀZčłNāzRāSŸæČšæTzāRŸPythonézŸēōd'èaŇāyžĩijNāIJłēŁTāZdæŪāçł'ūād'gæŁŪNaNçzŠæ  
fpectl æłāāIŪāRřāzèçTłæłæTzāRŸèŁZçg■èaŇāyžĩijNā;EæŸřāōČāIJłæāGāGEçŽDPythonædDāzžāy■āžū  
āžūāyTēŠŁāržçŽDæŸřāyŠāōūçžgçłNāzRāSŸāĀČāRřāzēāRCèĀČāIJłçžŁçŽDPythonæŪGæāçèŌūāRŪæZt'ād

## 5.8 3.8 āŁEæTřèŁRçōŪ

### éŪōéčŸ

ā;āèŁZāĒēæŪūēŪt'æIJžāZłĩijNçłAçDūāRŠçŌřā;āæ■čāIJłāAžārRā■ēāōūāž■ā;IJāyŽĩijNāžūæŪL'āRLāŁřā  
æŁŪēĀĒā;āāRřèČ;éIJāèçAāEZāzčçāAāŌžèōāçōŪāIJłā;āçŽDæIJłāūēāūēāŌČāy■çŽDætNēGRāĀijāĀČ

### èğčāEšæŪzæāŁ

fractions æłāāIŪāRřāzèèçTłæłæL'gèaŇāŇĒāRnāŁEæTřçŽDæTřā■èŁRçōŪāĀČæřTāçĆĩijŽ

```
>>> from fractions import Fraction
>>> a = Fraction(5, 4)
>>> b = Fraction(7, 16)
>>> print(a + b)
27/16
>>> print(a * b)
35/64

>>> # Getting numerator/denominator
>>> c = a * b
>>> c.numerator
35
>>> c.denominator
64
```

```
>>> # Converting to a float
>>> float(c)
0.546875

>>> # Limiting the denominator of a value
>>> print(c.limit_denominator(8))
4/7

>>> # Converting a float to a fraction
>>> x = 3.75
>>> y = Fraction(*x.as_integer_ratio())
>>> y
Fraction(15, 4)
>>>
```

## ěóíěőž

aIJa'd' g'a'd' ZæTɾçl'NázRáy■āy'ĀeL'nāy■aij'ZāGžç'Ō'ra'LEæTɾç'ZDèðaç'ð'Ūe'Ūðeç'Y'ijNā;EæY'  
 ærT'æç'çii'jNāIJa'y'Āāy'lā'EĀeðy'æŌeā'RŪa'LEæTɾā;çai'jRç'ZDæt'NērT'ā■Tā;■āzūāzēā'LEæTɾā;çai'j'  
 çZt'æŌeā;ççç'Tīā'LEæTɾāR'ražēā'GRār'SæL'Nā'lē;ñæ■cāy'zārRæTɾæL'Ūæt'ōç'ČzæTɾç'ZDāuēā;IJā

### 5.9 3.9 ăđ'ğăđNæŤřçzĎèŁŘćóŮ

## éÚóécŸ

ä;äéIĀĎèĕAāIĴlād'gæT̥ræ■óéZĒ(æfT̥āĕCæT̥rçzDæLŮĵ;Šæäij)äyŁéíćæL'gèāÑèðaçōŮãĀĆ

## èğçâEşşæÚzæąŁ

æul'ârĹāĹræȚrçzDçZĐēĠēGRçzğēfRçōŪæŞ■ä;IġijŊāRřäzēä;ŁçŦĬ NumPy āzŞāĀĆ  
NumPy çZĐäyĀäyĹäyžēēAçŁ'zā;AæŸrāōÇäijZçzZPythonæRŘä;ZäyĀäyĹæȚrçzDăržēsajijŊçZ  
äyŊéĬæŸrāyĀäyĹçōĀ■ȚçZĐărŘä;Ŋā■RġijŊāRŚajāāsȚçd'zæāGāGEāĹŪēalăržēsāāŊ  
NumPy æȚrçzDăržēsāāzŊéŪŦ'çZĐāũōāĹŋijZ

```
>>> # Python lists
>>> x = [1, 2, 3, 4]
>>> y = [5, 6, 7, 8]
>>> x * 2
[1, 2, 3, 4, 1, 2, 3, 4]
>>> x + 10
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate list (not "int") to list
>>> x + y
[1, 2, 3, 4, 5, 6, 7, 8]
```

```
>>> # Numpy arrays
>>> import numpy as np
>>> ax = np.array([1, 2, 3, 4])
>>> ay = np.array([5, 6, 7, 8])
>>> ax * 2
array([2, 4, 6, 8])
>>> ax + 10
array([11, 12, 13, 14])
>>> ax + ay
array([ 6, 8, 10, 12])
>>> ax * ay
array([ 5, 12, 21, 32])
>>>
```

æ■çæĆæL'ÀèġAīijNāyd'çġæŮzæaLāy■æTřčzDčŽDāšzæIJnæTřā■æēfRčōŮčzŠædIJāzūāy■çZyāRŇāA  
 çL'zāLnčŽDīijN NumPy äy■çŽDæāĠéĠRēfRčōŮ(æfTāēĆ ax  
 \* 2 æLŮ ax + 10 )āijŽā;IJçTlāIJlæfRāyĀāyġāĒČçt'āāyġāĀĆ  
 āRēād'ŮīijNā;Šāyd'āyġæS■ā;IJæTřēČ;æYřæTřčzDčŽDæŮūāĀZæL'ġèāNāĒČçt'āāřzç■L'ā;■ç;ōēōaçōŮīijNāz  
 āřzæTř'āyġæTřčzDāy■æL'ĀæIJL'āĒČçt'āāRŇæŮūæL'ġèāNāTřā■æēfRčōŮāRřāzēā;ġā;Ůā;IJçTlāIJlæTř'ā  
 æfTāēĆīijNāēĆædIJā;āæČšēōaçōŮād'ŽéāzāijRčŽDāĀijīijNāRřāzēēfZæāūāĀŽīijŽ

```
>>> def f(x):
...     return 3*x**2 - 2*x + 7
...
>>> f(ax)
array([ 8, 15, 28, 47])
>>>
```

NumPy ēfYāyžæTřčzDæS■ā;IJæRŖā;ŽāžEāđ'ġéĠRčŽDēĀŽçTlāĠ;æTřīijNēfZāžZāĠ;æTřāRřāzēā;IJā  
 math æġāāŮāy■çšzāijāĠ;æTřčŽDæŽfāzčāĀĆæfTāēĆīijŽ

```
>>> np.sqrt(ax)
array([ 1. , 1.41421356, 1.73205081, 2. ])
>>> np.cos(ax)
array([ 0.54030231, -0.41614684, -0.9899925 , -0.65364362])
>>>
```

ā;ġçTlēfZāžZēĀŽçTlāĠ;æTřēAæfTā;ġçŌræTřčzDāzūā;ġçTl  
 math æġāāŮāy■çŽDāĠ;æTřæL'ġèāNēōaçōŮēēAāfŋçŽDād'ŽāĀĆ  
 āZāæ■d'īijNāRlēēAæIJL'āRfēČ;çŽDēfġā;ēĠRēĀL'æŇŮ NumPy çŽDæTřčzDæŮzæaLāĀĆ

āžTāšĆāōđçŌrāy■īijN NumPy æTřčzDā;ġçTlāžEĆæLŮēĀĒFortranērēġġçŽDæIJzāLūāLēēĒ■āĒēā■Yā  
 āžšāřsæYřērt'īijNāōĆāznæYřāyĀāyġēġāyāđ'ġçŽDēfđçz■çŽDāžūçTlāRŇçšzādNæTřæ■ōçzDæLŖçŽDāĒēā  
 æL'ĀāžēīijNā;āāRřāzēædĒēĀāyĀāyġæfTæZōēĀŽPythonāLŮēāġād'ġçŽDād'ŽçŽDæTřčzDāĀĆ  
 æfTāēĆīijNāēĆædIJā;āæČšædĒēĀāyĀāyġ10,000\*10,000çŽDætōçČzæTřāžNçzt'ç;ŠæāijīijNā;Lē;žæġīijŽ

```
>>> grid = np.zeros(shape=(10000,10000), dtype=float)
>>> grid
array([[ 0.,  0.,  0., ...,  0.,  0.,  0.]
```

```

[ 0., 0., 0., ..., 0., 0., 0.],
[ 0., 0., 0., ..., 0., 0., 0.],
...,
[ 0., 0., 0., ..., 0., 0., 0.],
[ 0., 0., 0., ..., 0., 0., 0.],
[ 0., 0., 0., ..., 0., 0., 0.]]
>>>

```

æL'ÄæIJL'çŽDæŽóéĂŽæŞ■ä;IJèfYæYřäijŽăŘŇæŮüä;IJçTlăIJlæL'ÄæIJL'ăĚČt'ăäyŁiijŽ

```

>>> grid += 10
>>> grid
array([[ 10., 10., 10., ..., 10., 10., 10.],
       [ 10., 10., 10., ..., 10., 10., 10.],
       [ 10., 10., 10., ..., 10., 10., 10.],
       ...,
       [ 10., 10., 10., ..., 10., 10., 10.],
       [ 10., 10., 10., ..., 10., 10., 10.],
       [ 10., 10., 10., ..., 10., 10., 10.]])
>>> np.sin(grid)
array([[ -0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
       [ -0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
       [ -0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
       ...,
       [ -0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
       [ -0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111],
       [ -0.54402111, -0.54402111, -0.54402111, ..., -0.54402111,
        -0.54402111, -0.54402111]])
>>>

```

ăĚşăžŎ NumPy æIJL'äyĂçCzéIJĂēçAçL'zălŇçŽDäyžæĐRiijŇéCčârşæYřăóČæL'řăsTPythonăLŮealçŽD  
-çL'zălŇæYřărzăžŎăđ'Žçzt'æTřçzDăĂČ äyžăžĚçrt'æYŎæyĚæčŽiijŇăĚLăđĐéĂăäyĂäyłçŎĂ■TçŽDăžŇçzt'

```

>>> a = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
>>> a
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])

>>> # Select row 1
>>> a[1]
array([5, 6, 7, 8])

>>> # Select column 1
>>> a[:,1]
array([ 2,  6, 10])

```

```

>>> # Select a subregion and change it
>>> a[1:3, 1:3]
array([[ 6,  7],
       [10, 11]])
>>> a[1:3, 1:3] += 10
>>> a
array([[ 1,  2,  3,  4],
       [ 5, 16, 17,  8],
       [ 9, 20, 21, 12]])

>>> # Broadcast a row vector across an operation on all rows
>>> a + [100, 101, 102, 103]
array([[101, 103, 105, 107],
       [105, 117, 119, 111],
       [109, 121, 123, 115]])

>>> a
array([[ 1,  2,  3,  4],
       [ 5, 16, 17,  8],
       [ 9, 20, 21, 12]])

>>> # Conditional assignment on an array
>>> np.where(a < 10, a, 10)
array([[ 1,  2,  3,  4],
       [ 5, 10, 10,  8],
       [ 9, 10, 10, 10]])
>>>

```

## ěőłěőž

NumPy æŸřPythoněćĚāššäy■ā;Łād'ŽçġŚā■ēäyŌāũēćĹŇāžŚçŽĎāšžçāĀiijŇāŔŇæŮūāžšæŸřěćŇāžŁæšŽā■şä;ŁæēĆæ■đ'ijŇāĪĹāŁŽāijĀāġŇçŽĎæŮūāĀŽéĀŽēŁĠāyĀāžŽçōĀā■ŤçŽĎä;Ňā■ŔāŤŇçŌĹ'āĚūćĹŇāžŔāžš

éĀŽāyŷæĹŚāžŇārijāĚĚ NumPy æĹāāĪŮçŽĎæŮūāĀŽāijŽā;ŁçŤĹér■āŔĚ import numpy as np āĀĆ èŁŽæāũçŽĎērĹā;āāřsäy■çŤĹāĒ■ā;āçŽĎćĹŇāžŔéĠŇéĹcāyĀéĀ■éĀ■çŽĎæŤšāĚĚ numpy iijŇāŔĹēĪĀēēĀē;ŚāĚĚ np āřsēāŇāžĚiijŇēĹĆçĪĪāāžĚāy■āřŚæŮūēŮŤāĀĆ

āēĆæđĪæČšēŌūāŔŮæŽŤād'ŽçŽĎāŁæĀřijŇā;āā;ŚçĎūā;ŮāŌž NumPy āōŸç;ŚéĀŽéĀŽāžĚiijŇç;ŚāĪæŸřijŽ <http://www.numpy.org>

## 5.10 3.10 çšĹ'éŸŤäyŌçžŁæĀġāžçæŤřèŁŔçŌŮ

### éŮőéćŸ

ä;äēĪĀēēĀæĹġēāŇçšĹ' éŸŤāŤŇçžŁæĀġāžçæŤřèŁŔçŌŮiijŇāřŤāēĆçšĹ'éŸŤāžŸæšŤāĀāřžæĹ;èāŇāĹŮā

## èġċăEşæŮzæąĹ

NumPy äŻŞæIJĹäyÄäyġçŞĹ' éYġårzèşąăRřazèçŤĹăĹèġċăEşæŹăyĹéŮőécŸăĂĆ  
çŞĹ' éYġçşzäijijăŹŌ3.9ăRèĹĆăy■æŤřçzĎărzèşăijŇăĹEæŸréAġăĹçŻĹæĂġăzçæŤřçŹĎèőăçőŮèġĎăĹŹăĂ

```
>>> import numpy as np
>>> m = np.matrix([[1,-2,3],[0,4,5],[7,8,-9]])
>>> m
matrix([[ 1, -2,  3],
         [ 0,  4,  5],
         [ 7,  8, -9]])

>>> # Return transpose
>>> m.T
matrix([[ 1,  0,  7],
        [-2,  4,  8],
         [ 3,  5, -9]])

>>> # Return inverse
>>> m.I
matrix([[ 0.33043478, -0.02608696,  0.09565217],
        [-0.15217391,  0.13043478,  0.02173913],
         [ 0.12173913,  0.09565217, -0.0173913 ]])

>>> # Create a vector and multiply
>>> v = np.matrix([[2],[3],[4]])
>>> v
matrix([[2],
         [3],
         [4]])
>>> m * v
matrix([[ 8],
        [32],
         [ 2]])
>>>
```

ăRřazèăIJĹ numpy.linalg ä■RăŇĚäy■æĹĹăĹăŹt'ăd'ŹçŹĎæŞ■ăĹJăĠăŹăŤřijŇăŹŤăçĆijŹ

```
>>> import numpy.linalg
>>> # Determinant
>>> numpy.linalg.det(m)
-229.99999999999983

>>> # Eigenvalues
>>> numpy.linalg.eigvals(m)
array([-13.11474312,  2.75956154,  6.35518158])

>>> # Solve for x in mx = v
>>> x = numpy.linalg.solve(m, v)
```



```

>>> x
matrix([[ 0.96521739],
        [ 0.17391304],
        [ 0.46086957]])
>>> m * x
matrix([[ 2.],
        [ 3.],
        [ 4.]])
>>> v
matrix([[2],
        [3],
        [4]])
>>>

```

## èóíèõž

āĶĹæŸĶçĎŮçžĤæĀğăžçæŦræŸrăyléİđăyyăđ'ğçŽĎăyzécŸriijŇăũşçžRėúĒăĜžăžĒæĬŇăžęèĈ;èóíèõžçŽĎ  
 äĶĒæŸriijŇăęĈăđĬĶă;ăéĬĶăęęĀæŞ■ăĶĬæŦrçžĎăŇăŔŤéĜŦçŽĎēŦriijŇ NumPy  
 æŸrăyĀăylăy■éŦŽçŽĎăĒăŦŦççĈžăĀĈ āŦŦăžëèőféŮő NumPy āőŸç;Ś <http://www.numpy.org>  
 èŬăăŦŮăŽŦ'ăđ'ŽăŦăæĀŦăĀĈ

## 5.11 3.11 éŽŦæĬžéĀĹæŇŦ'

### éŬőéçŸ

äĶăæĈşăžŌăyĀăylăžŦăĹŮăy■éŽŦæĬžæĹ;ăŦŮăĒăžăžăĒĈçŦ'ăriijŇăĹŮăĒăæĈşçŦşæĹŦăĜăăyléŽŦæĬž

### èğĉăĒşæŮžæăĹ

random æĶăăĬŮăĬĹ'ăđ'ğéĜŦçŽĎăĜĶæŦrçŦĬăĬăžğçŦşéŽŦæĬžæŦŦăŇăĒŦŦæĬžéĀĹæŇŦ'ăĒĈçŦ'ăăĀĈ  
 æŦŦăĒĈriijŇăęĀæĈşăžŌăyĀăylăžŦăĹŮăy■éŽŦæĬžçŽĎăĹ;ăŦŮăyĀăylăĒĈçŦ'ăriijŇăŦŦăžăäĶçŦĬ  
 random.choice() ĩijŽ

```

>>> import random
>>> values = [1, 2, 3, 4, 5, 6]
>>> random.choice(values)
2
>>> random.choice(values)
3
>>> random.choice(values)
1
>>> random.choice(values)
4
>>> random.choice(values)
6
>>>

```

random.sample() **iiž**

```
>>> random.sample(values, 2)
[6, 2]
>>> random.sample(values, 2)
[4, 3]
>>> random.sample(values, 3)
[4, 3, 1]
>>> random.sample(values, 3)
[5, 4, 1]
>>>
```

random.shuffle() **iiž**

```
>>> random.shuffle(values)
>>> values
[2, 4, 6, 5, 3, 1]
>>> random.shuffle(values)
>>> values
[3, 5, 2, 1, 6, 4]
>>>
```

random.randint() **iiž**

```
>>> random.randint(0,10)
2
>>> random.randint(0,10)
5
>>> random.randint(0,10)
0
>>> random.randint(0,10)
7
>>> random.randint(0,10)
10
>>> random.randint(0,10)
3
>>>
```

random() **iiž**

```
>>> random.random()
0.9406677561675867
>>> random.random()
0.133129581343897
>>> random.random()
0.4144991136919316
>>>
```

random.getrandbits() iijŽ

```
>>> random.getrandbits(200)
335837000776573622800628485064121869519521710558559406913275
>>>
```

## ěóíeőž

random aĺaǎıŮä;ŁčŤı *Mersenne Twister* čóŮașŤaēlēőačóŮčŤšaĹŔěŽŔaēIJžaeŤřāĀČēŁžaeŸřäŸÄäŸŁč; ä;EaeŸřä;āāŔřäžēēĀŽēŁĜ random.seed() āĜ;aeŤřāŁōaeŤžāĹıāĝŊāŊŮčĝ■ā■ŔāĀČaēŤāēČııjŽ

```
random.seed() # Seed based on system time or os.urandom()
random.seed(12345) # Seed based on integer given
random.seed(b'bytedata') # Seed based on byte data
```

éŽd'āžEäŸŁēŁřāžŊč;■čŽDāŁšēČ;ııjŊrandomaĺaǎıŮēŁŸāŊĚāŔŋāšžāžŌāıĜāŊāĹEäŸČāĀAēŋŸaeŮřā aēŤāēČııjŊ random.uniform() ēőačóŮāıĜāŊāĹEäŸČēŽŔaēIJžaeŤřııjŊ random.gauss() ēőačóŮaē■čaeĀAāĹEäŸČēŽŔaēIJžaeŤřāĀČ āŕžāžŌāĚŮāžŮčŽDāĹEäŸČaeČĚāĒēŮāŔČēĀČāıĹčžŁaeŮĜaeāčāĀČ

āıĹı random aĺaǎıŮäŸ■čŽDāĜ;aeŤřäŸ■āžŤēŕēčŤıāıĹıāŊŋāŕEçāAā■ēçŽŸāĚșçŽDçıĹāžŔäŸ■āĀČ āēČaēđıĴa;āčāőāőđēıĴāēēAçšžäııjčŽDāŁšēČ;ııjŊāŔřäžēä;ŁčŤısslēaĺaǎıŮäŸ■čŽŸāžŤčŽDāĜ;aeŤřāĀČ aēŤāēČııjŊ ssl.RAND\_bytes() āŔřäžēčŤıāēčŤšaĹŔäŸÄäŸıāőĹāĚıĹčŽDēŽŔaēIJžā■ŮēŁČāžŔāĹŮāĀČ

## 5.12 3.12 āšžaeıĴŋčŽDaeŮēaeıĴšäŸŌaeŮúēŮŤ'ē;ŋae■č

### éŮēéčŸ

ä;āēıĴāēēAaeŁĝēāŊčóĀā■ŤčŽDaeŮúēŮŤ'ē;ŋae■čııjŊaēŤāēČāđŤāĹŕčĝšııjŊāŕŔaeŮúāĹŕāĹEēšșç■ŁčŽD

## ēĝčāEșaeŮžaeāĹ

äŸžāžEaeŁĝēāŊäŸ■āŔŊaeŮúēŮŤ'ā■Ťä;■čŽDē;ŋae■čāšŊēőačóŮııjŊēŕüā;ŁčŤı datetime aĺaǎıŮāĀČ aēŤāēČııjŊäŸžāžEaeāĹčđ'žäŸÄäŸıaeŮúēŮŤ'aeőııjŊāŕŔäžēāĹŽāžžäŸÄäŸŁ timedelta aőđä;ŊııjŊāŕšāČŔäŸŊēıĹēŁžaeāııjŽ

```
>>> from datetime import timedelta
>>> a = timedelta(days=2, hours=6)
>>> b = timedelta(hours=4.5)
>>> c = a + b
>>> c.days
2
>>> c.seconds
37800
>>> c.seconds / 3600
10.5
```

```
>>> c.total_seconds() / 3600
58.5
>>>
```

æCædIä;äæCšèáíçd'žæŇGåóŽçŽDæŮææIJšåŠŇæŮúéŮt'ijŇãĚĹáĹZázžäyÄäyĭ  
datetime åóðä;ŇçĎŮåŔŌä;ŁçŦĹæåĠåĠĖçŽDæŦŕå■èŁŔçóŮæĹæš■ä;IJåóČäzňãĀCærŦæČiijŽ

```
>>> from datetime import datetime
>>> a = datetime(2012, 9, 23)
>>> print(a + timedelta(days=10))
2012-10-03 00:00:00
>>>
>>> b = datetime(2012, 12, 21)
>>> d = b - a
>>> d.days
89
>>> now = datetime.today()
>>> print(now)
2012-12-21 14:54:43.094063
>>> print(now + timedelta(minutes=10))
2012-12-21 15:04:43.094063
>>>
```

åIJĹèøaçóŮçŽDæŮúåĀŽiijŇéIJĀèçAæşĹæĎŔçŽDæŸŕ  
äijŽèĠåĹĹåđ'ĎçŔĖéŮŕázŦ'ãĀCærŦæČiijŽ

datetime

```
>>> a = datetime(2012, 3, 1)
>>> b = datetime(2012, 2, 28)
>>> a - b
datetime.timedelta(2)
>>> (a - b).days
2
>>> c = datetime(2013, 3, 1)
>>> d = datetime(2013, 2, 28)
>>> (c - d).days
1
>>>
```

## èóĹèőž

åržåd'ğåd'ŽæŦŕåşžæIJŇçŽDæŮææIJšåŠŇæŮúéŮt'åd'ĎçŔĖéŮøécŸiijŇ  
æĹåĹåĹŮåšçžŔèŮşåd'şäžĖãĀC æCædIä;äéIJĀèçAæĹ'ğèaŇæŽŦ'åĹååđ'■æĹČçŽDæŮææIJšæš■ä;IJiijŇærŦæČ  
årŕfäzèèĀCèŽŠä;ŁçŦĹ dateutilæĹåĹŮ

èöyåd'ŽçşžäijijçŽDæŮúéŮt'èøaçóŮåŔŕäzèä;ŁçŦĹ  
åĠæŦŕäzçæŽĚãĀC ä;ĖæŸŕiijŇæIJĹäyĀçČzéIJĀèçAæşĹæĎŔçŽDærşæŸŕiijŇåóČäijŽåIJĹåđ'ĎçŔĖæIJĹäz;(è

```
>>> a = datetime(2012, 9, 23)
>>> a + timedelta(months=1)
```

```

Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: 'months' is an invalid keyword argument for this function
>>>
>>> from dateutil.relativedelta import relativedelta
>>> a + relativedelta(months=+1)
datetime.datetime(2012, 10, 23, 0, 0)
>>> a + relativedelta(months=+4)
datetime.datetime(2013, 1, 23, 0, 0)
>>>
>>> # Time between two dates
>>> b = datetime(2012, 12, 21)
>>> d = b - a
>>> d
datetime.timedelta(89)
>>> d = relativedelta(b, a)
>>> d
relativedelta(months=+2, days=+28)
>>> d.months
2
>>> d.days
28
>>>

```

## 5.13 3.13 èõäçóŮæIJĀăŘŮäÿĂäÿłăŚĺăžŤčŽĎæŮěæIJš

### éŮóécŸ

äĵăéIJĀěçAæšěæL'çæŸšæIJšäÿ■æšŘäÿĂăđ'l'æIJĀăŘŮăĜžçŎřçŽĎæŮěæIJšĭĵŊæŕŤăçĆæŸšæIJšăžŤă

### èğčăEşæŮžæąĹ

PythonçŽĎ datetime æĺăăĭŮäÿ■æIJĹăŭěăĚŭăĜĵæŦŕăŠŊçşăŕăžăÿŏăĹ'l'ă;ăæL'ğëăŊëçŽæăŭçŽĎëö.äÿŊéĹæŸŕăŕçşăĭĵĭççŽæăŭçŽĎëŮóécŸçŽĎäÿĂäÿłăĂžçŦĹèğčăEşæŮžæąĹĭĵŽ

```

#!/usr/bin/env python
# -*- encoding: utf-8 -*-
"""
Topic: æIJĀăŘŮçŽĎăŚĺăžŦ
Desc :
"""
from datetime import datetime, timedelta

weekdays = ['Monday', 'Tuesday', 'Wednesday', 'Thursday',
              'Friday', 'Saturday', 'Sunday']

```

```
def get_previous_byday(dayname, start_date=None):
    if start_date is None:
        start_date = datetime.today()
    day_num = start_date.weekday()
    day_num_target = weekdays.index(dayname)
    days_ago = (7 + day_num - day_num_target) % 7
    if days_ago == 0:
        days_ago = 7
    target_date = start_date - timedelta(days=days_ago)
    return target_date
```

āĪlāzd'āžŠāijRēgčēĠLāZlāy■ā;ĤçTlāçCāyNriiŽ

```
>>> datetime.today() # For reference
datetime.datetime(2012, 8, 28, 22, 4, 30, 263076)
>>> get_previous_byday('Monday')
datetime.datetime(2012, 8, 27, 22, 3, 57, 29045)
>>> get_previous_byday('Tuesday') # Previous week, not today
datetime.datetime(2012, 8, 21, 22, 4, 12, 629771)
>>> get_previous_byday('Friday')
datetime.datetime(2012, 8, 24, 22, 5, 9, 911393)
>>>
```

āRréĀLçŽĎ start\_date āRĈæTŗāRřazēçTśāRēad'ŪāyĀāyĭ datetime  
āōđā;ŊāēĪæRŔä;ZāĀĈærTāçĆrijŽ

```
>>> get_previous_byday('Sunday', datetime(2012, 12, 21))
datetime.datetime(2012, 12, 16, 0, 0)
>>>
```

## ēōlēōž

āyLēĪççŽĎçōŪæşTāŌşçRĒæYřēĤZæāūçŽĎriiŽāĒĻāřEāijĀāgNæŪēæIJšāŠNçŽōæāĠæŪēæIJšæYāārĎ.  
çĎŮāRŌēĀŽēĤĠāĻæĤRçōŪēōāçōŪāĠžçŽōæāĠæŪēæIJšēæAçzRēĤĠād'ŽārŠād'ĤæL■ēĈ;āĻrē;āijĀāgNæŪ

āēĈæđIJā;āēæĀāĈRēĤZæāūæL'gēāNād'gēĠRçŽĎæŪēæIJšēōāçōŪçŽĎērĪiijNā;āæIJĀāē;āōL'ēçĒēñnāyL  
python-dateutil æĪēāzçæŽēāĀĈ æřTāçĆrijNāyŊēĪæYřæYřā;ĤçTĪ dateutil  
æĻāāĪŪāy■çŽĎ relativedelta() āĠ;æTŗæL'gēāNāRŊæāūçŽĎēōāçōŪriiŽ

```
>>> from datetime import datetime
>>> from dateutil.relativedelta import relativedelta
>>> from dateutil.rrule import *
>>> d = datetime.now()
>>> print(d)
2012-12-23 16:31:52.718111

>>> # Next Friday
>>> print(d + relativedelta(weekday=FR))
2012-12-28 16:31:52.718111
```

```
>>>

>>> # Last Friday
>>> print(d + relativedelta(weekday=FR(-1)))
2012-12-21 16:31:52.718111
>>>
```

## 5.14 3.14 èóäçõÜä;ŞåL'æIJLäz;çŽDæUëæIJŞèŇCăŽt'

### éUóécŸ

ä;ăçŽDăzççăAéIJĀèçAăIJlă;ŞåL'æIJLäz;äy■ă;łçŎræfRäyĀăd'fiijŇæČşæL'ăLřäyĀäyłèóäçõUèŁZäyłæ

### èğçăEşæŮzæąŁ

ăIJłèŁZæăüçŽDæUëæIJŞäyŁă;łçŎrăzűéIJĀèçAăžŇăĒŁădĐéĀăyĀäyłăŇĒăŔŇăL'ĀæIJL'æUëæIJŞçŽD  
ä;ăăŔŕăžăĒĒŁèóäçõÜăĠzăijĀăğŇæUëæIJŞăŖŇçzŞæİŞæUëæIJŞiiŇ  
çDŭăŔŎăIJlă;ăæ■èŁZçŽDæUŭăĀŽă;ŁçŦÍ  
ărzèsăéĀŖăcđēŁZäyłæUëæIJŞăŔŸéĠŔă■şăŔŕăĀĆ  
datetime.timedelta

äyŇéİcăŸřäyĀäyłæŎăŔŮăžzæĐŔ datetime.ărzèsăăžűēŁŦăŽđäyĀäyłçŦśă;ŞåL'æIJLäz;ăijĀăğŇæU

```
from datetime import datetime, date, timedelta
import calendar

def get_month_range(start_date=None):
    if start_date is None:
        start_date = date.today().replace(day=1)
    _, days_in_month = calendar.monthrange(start_date.year, start_
→date.month)
    end_date = start_date + timedelta(days=days_in_month)
    return (start_date, end_date)
```

æIJL'ăžĒēŁZäyłăŕşăŔŕăžăăŁăőžæŸŞçŽDăIJłèŁŦăŽđçŽDæUëæIJŞèŇCăŽt'äyŁéİcăĀŽă;łçŎræŞ■ă;IJăž

```
>>> a_day = timedelta(days=1)
>>> first_day, last_day = get_month_range()
>>> while first_day < last_day:
...     print(first_day)
...     first_day += a_day
...
2012-08-01
2012-08-02
2012-08-03
2012-08-04
2012-08-05
2012-08-06
```

èóìèőž

```
>>> for d in date_range(datetime(2012, 9, 1), datetime(2012, 10, 1),
                        timedelta(hours=6)):
...     print(d)
...
2012-09-01 00:00:00
2012-09-01 06:00:00
2012-09-01 12:00:00
2012-09-01 18:00:00
2012-09-02 00:00:00
2012-09-02 06:00:00
...
>>>
```



èŁŻçġ■āōđçŎřāzŊæL'ĂäzèèŁŻāzŁçōĂā■ŤiijŊèŁŸāŁŮāŁŤāŁšāzŎPythonāy■čŽĐæŮæIJšāŤŊæŮéŮŤŮ

## 5.15 3.15 ā■ŮčņęäŸšèĭŋæ■ćäŸæŮěæIJš

### éŮōéćŸ

äĭăçŽĐāžŤčŤĭćĭŊāžŤæŎěāŤŮā■ŮčņęäŸšæāĭĭāĭŤčŽĐēŁŤšāĔēĭĭŊāĭĖæŸŤāĭăæČšāŤĖāōČāzŋēĭŋæ■ćäŸæ  
datetime āŤžēšāžēäĭŁāIJĭäŸŁēĭćæL'ġēāŊēĭđā■ŮčņęäŸšæš■āĭIJāĀČ

### èġčāĖšæŮzæāŁ

äĭŁçŤĭPythončŽĐæāĠāĠĖæĭāāĭŮ datetime āŤŤāžēāŁŤāōžæŸščŽĐēġčāĖšēŁŤāŸĭéŮōéćŸāĀČæŤāēČ

```
>>> from datetime import datetime
>>> text = '2012-09-20'
>>> y = datetime.strptime(text, '%Y-%m-%d')
>>> z = datetime.now()
>>> diff = z - y
>>> diff
datetime.timedelta(3, 77824, 177393)
>>>
```

### èŏĭèŏž

datetime.strptime() æŮzæšŤæŤŤæŊŤāāŁŤād'ŽčŽĐæāĭĭāĭŤāŊŮāzččāĖĭĭŊ  
æŤŤāēČ %Y äžčēāĭ4äĭ■æŤŤāžŤ'äžĭĭĭŊ %m äžčēāĭäŸd'äĭ■æŤŤæIJĭLāžĭāĀČ  
èŁŸæIJL'äŸĖčČāĖĭāĭāŁŮæšĭæĐŤčŽĐæŸŤēŁŤāžŽæāĭĭāĭŤāŊŮā■āĭ■čņęäžšāŤŤāžēāŤŮēŁĠæĭāĭŁçŤĭĭŊŊāŤŮ  
æŤŤāēČĭĭŊŊāĖġēŎĭäĭăçŽĐāžččāĖāŸ■čŤšæŁŤāžĖāŸĖāŸŮ datetime āŤžēšāĭĭŊ  
äĭăæČšāŤĖāōČæāĭĭāĭŤāŊŮāŸžæĭĖčāžŏæŸšēŤŤāĭčāĭŤāŤŮæŤĭāIJĭēĠāŁĭčŤšæŁŤŤčŽĐāŁāžŮæŁŮēĀĖæŁēā

```
>>> z
datetime.datetime(2012, 9, 23, 21, 37, 4, 177393)
>>> nice_z = datetime.strftime(z, '%A %B %d, %Y')
>>> nice_z
'Sunday September 23, 2012'
>>>
```

èŁŸæIJL'äŸĖčČzéIJĖēĖĖæšĭæĐŤčŽĐæŸŤĭĭŊ strftime()  
čŽĐæĀġēČĭēĖĖæŤŤāĭăæČšēšāŸ■čŽĐāŮŏāŁŤād'ŽĭĭŊ āŽāŸžāŏČæŸŤāĭŁçŤĭćŤŮPythonāōđçŎřĭĭŊŊāžŮāŸŤāŁŤ  
āēČādIJāĭāēĖĖĖĖĖččāĖāŸ■ēIJĖēĖĖĖġčæđŤād'ġēĠŤčŽĐæŮěæIJšāžŮāŸŤāŮščžŤšēēĖĖšāžĖæŮěæIJšā■Ů  
æŤŤāēČĭĭŊŊāēČādIJāĭāāŮščžŤšēēĖĖšāšæL'ĂäžēæŮěæIJšæāĭĭāĭŤæŸŤ YYYY-MM-DD  
ĭĭŊŊāĭăāŤŤāžēāČŤāŸŊēĭĭēŁŤæāŮāōđçŎřāŸĖāŸĭēġčæđŤāĠĖŤĭĭŊŽ

```
from datetime import datetime
def parse_ymd(s):
```

```
year_s, mon_s, day_s = s.split('-')
return datetime(int(year_s), int(mon_s), int(day_s))
```

åóðéŽĚæŧNèŕTäy■iijNëfZäyſaĜ;æTŕæfT datetime.strptime() åſn7åĀ■ad'ŽāĀĆ  
åĉĆæđIJä;äèĉAād'DĉŘĚād'ġéĜRĉŽĎæŭL'åŖLåĽŕæŮëæIJşĉŽĎæTŕæ■ōĉŽĎèŕiijNéĆcăžĽæIJĀāē;èĀĈèŽŚā

## 5.16 3.16 çŻŞåŖĽæŮūāNžĉŽĎæŮëæIJşæŞ■ä;IJ

### éŮóéĲ

ä;äæIJL'äyÄäyſaōL'æŎŚåIJĬ2012åžŕ'12æIJĬ21æŮëæŮſ'äyſ9:30çŽĎĉTŕèſſaijŽèōōiijNåIJŕĉĆžåIJĽèĽſåŁ  
èĀNä;äĉŽĎæIJNåŖNåIJĀ■řāžĉŽĎĉŖ■åĽă;ŮårTŕiijNéĆcăžĽăžŮăžTèŕĉåIJĭ;ŞåIJŕæŮüéŮŕ'åĜăĉĆžåŖĈăĽă

### èġĉÅĒşæŮžæąĽ

åržåĜăăžŎæL'ĀæIJL'æŭL'åŖLåĽŕæŮūāNžĉŽĎéŮóéĲYiijNä;äĉĈ;ăžTèŕĉă;ĲĉTĬ  
pytz æĽąăĽŮāĀĈĉĲZäyſaŊĒæŖŖă;ŽăžĒOlsonæŮūāNžæTŕæ■ōăžŞiijN  
åŎĈæŸŕæŮūāNžăĲæĀŕĉŽĎăžNăōđăyĽĉŽĎæăĜăĜĒiijNåIJĭ;Ľăđ'Žĕŕ■ĽĀăŞNæŞ■ă;IJşĉžĉžşĉĜNéĽĉĈ;ăŖ

pytz æĽąăĽŮäyÄäyſăyžĉĲĀĈſĽéĀTæŸŕăŖĒ datetime  
ăžŞăĽŽăžžĉŽĎĉŎĀ■TæŮëæIJşåržĉşæIJNåIJŕăNŮāĀĈ æŕTăĉĈiijNăyNéĽĉăĉĈă;TĕăĲĉđ'žăyÄäyſĽĽăĽăăŞă

```
>>> from datetime import datetime
>>> from pytz import timezone
>>> d = datetime(2012, 12, 21, 9, 30, 0)
>>> print(d)
2012-12-21 09:30:00
>>>

>>> # Localize the date for Chicago
>>> central = timezone('US/Central')
>>> loc_d = central.localize(d)
>>> print(loc_d)
2012-12-21 09:30:00-06:00
>>>
```

äyĀæŮĉæŮëæIJşĉĉnæIJNåIJŕăNŮăžĒiijN åŎĈăŕşăŖŕăžĉĉ;ŋæ■ĉăyžăĒŮăžŮæŮūāNžĉŽĎæŮüéŮŕ'ăžĒăĀ  
äyžăžĒă;ŮăĽŕĉŖ■ăĽă;ŮårTăŕžăžTĉŽĎæŮüéŮŕ'iijNä;ăăŖăžĉĉĲZăăŮăĀŽiijŽ

```
>>> # Convert to Bangalore time
>>> bang_d = loc_d.astimezone(timezone('Asia/Kolkata'))
>>> print(bang_d)
2012-12-21 21:00:00+05:30
>>>
```

åĉĆæđIJä;äæL'ŞĉŏŮåIJĽæIJNåIJŕăNŮæŮëæIJşäyſæL'ġĕăNĕŏăĉŏŮiijNä;äĉIJĀĉĲĀĈL'žăĽNăşĽăĎŖăđ'Ŗă  
æŕTăĉĈiijNåIJĬ2013ăžŕ'iijNĉ;ŎăŽ;æăĜăĜĒăđ'Ŗăžđ'æŮüæŮüéŮŕ'ăijĀăġNăžŎæIJNåIJŕæŮüéŮŕ'3æIJĬ13æŮ  
åĉĆæđIJä;äæ■ĉåIJĽæL'ġĕăNăĒæIJNåIJŕĕŏăĉŏŮiijNä;ăăiijŽă;ŮăĽŕăyÄäyſĽĽŽĕŕŕăĀĈæŕTăĉĈiijŽ

```
>>> d = datetime(2013, 3, 10, 1, 45)
>>> loc_d = central.localize(d)
>>> print(loc_d)
2013-03-10 01:45:00-06:00
>>> later = loc_d + timedelta(minutes=30)
>>> print(later)
2013-03-10 02:15:00-06:00 # WRONG! WRONG!
>>>
```

çŞædIJeŦZèrræYřaZääyžăăČăžúăşşæIJL'èĂĈèZŚaIjIæIJnăIJræŮúéŮř'äy■æIJL'äyĂăřRæŮúçŽĐěůşěů  
 äyžăžEăřôă■čěřZăyleŦZèrríjNăRřăžěă;ŁçŦlæŮúăNžăržěşă  
 æŮžæşŦăĂĈæřŦăĈiijŽ

```
>>> from datetime import timedelta
>>> later = central.normalize(loc_d + timedelta(minutes=30))
>>> print(later)
2013-03-10 03:15:00-05:00
>>>
```

**èóìèőž**

äyžāžEäy■èól'äjäècnèfZāžZāyIjäyIjäijDçŽDæŽTād't'è;ñāRŠiijNād'DçŘEæIJñāIJřāNŮæUëæIJşçŽDéĀ  
 āzūçTlāōCæIëæL'gëāNāæL'ĀæIJL'çŽDäy■éUt'ā■ŸāĆlāSŇæS■ā:IJāĀCærTæÇiijŽ

```
>>> print(loc_d)
2013-03-10 01:45:00-06:00
>>> utc_d = loc_d.astimezone(pytz.utc)
>>> print(utc_d)
2013-03-10 07:45:00+00:00
>>>
```

äyÄæÛè;ñæ■cäyžUTCijñNä;äärsäy■çTlāŌzæNĒāfCēušād'Rāzd'æUūçZyāEšçŽDēŪōécYāzEāĀC  
āZāæ■d'ijñNä;āāRāzēūšāzNāL■äyÄæāuāTlāfCçŽDæLgēāNāyÿègAçŽDæŪēāIJšèōaçōŪāĀC  
ā;Šā;āæČsārEè;ŠāGžāRŸäyžæIJñāIJræŪūēŪ'çŽDæUūāĀZiijñNä;£çTlāRĒLēĀCçŽDæUūāNžāŌzè;ñæ■cäyNā

```
>>> later_utc = utc_d + timedelta(minutes=30)
>>> print(later_utc.astimezone(central))
2013-03-10 03:15:00-05:00
>>>
```

ā;ŠæuL'āRĹāLræUūāNžæŠ■ā;IJçŽDæUūāĀŽījNæIJL'āyĹēUōēcYārsæYræLSāznāēCā;Tā;UāLræUūāN  
 ærTāēCīijNāIJlēfZāyĹā;Nā■Rāy■ījNāLSāznāēCā;TçšēēAŠāĀIJAsia/KolkataāĀlārsæYrā■rāžēārzāžTçŽDæ  
 āyžāEæšēēL';ījNāRfrazēā;fçTīISO 3166āZ;āōūāzççāAā;IJāyžāĒšēTōā■UāŌzæšēēYĒā■UāEy  
 pytz.country\_timezones āĀCærTāēCīijŽ

```
>>> pytz.country_timezones['IN']
['Asia/Kolkata']
>>>
```

æʃliijZā;Šä;æYÈèrzāLrèfZéGŇçŽDæUúāĀŽiijNæIJL'āRrèČ; pyt z  
ælaaiUāušçzRäy■āE■āzžèōōä;£çTlāžEiijNāZāāyžPEP431æRŘāGžāžEæŽt'āĚĹè£ŽçŽDæUúāNžæTŕæNāāĀ  
ä;EæYŕèfZéGŇèrLāLŕçŽDä;Lād'ŽéUóécYè£YæYŕæIJL'āRČèĀČzūāĀijçŽD(æŕTāçCä;£çTlUTCæUēæIJšç

## 6 çññāZZçñāiijŽè£■āzčāZlāyÖçTšæLŘāZl

è£■āzčæYŕPythonæIJĀiijžād'ğçŽDāLšèČ;āzNāyĀāĀCālIçIJNètuæIeriijNā;āāRŕèČ;āijŽçóĀā■TçŽDèōō  
çDūèĀNriijNçZlèlāžĒāzĒāŕsæYŕæČæ■d'riijNè£YæIJL'ā;Lād'Zā;āāRŕèČ;āy■çšèéAšçŽDriijN  
æŕTāçCālZāzā;æĒGāušçŽDè£■āzčāZlāŕžèšāiijNāIJlitertoolsælaaiUāy■ā;£çTlæIJL'çTlçŽDè£■āzčælaaijRriij  
è£ZāyĀçñāçZōçŽDāŕsæYŕāRŠä;āāsTçd'žèušè£■āzčæIJL'āĚšçŽDāRĎçg■āyÿègAèUóécYāĀC

Contents:

### 6.1 4.1 æL'NāLlÉæA■āŌEè£■āzčāZl

#### éUóécY

ä;āæČšéA■āŌEäyĀäyĹāRŕè£■āzčāŕžèšāy■çŽDæL'ĀæIJL'āĚČçt'āriijNā;EæYŕā■'āy■æČšä;£çTlforā;łçČ

#### èğčāEšæÚzæaĹ

äyžāžEæL'NāLlçŽDèA■āŌEāRŕè£■āzčāŕžèšāiijNā;£çTl next()  
āG;æTŕāžūāIJlāzčçāAäy■æ■TèŌū StopIteration āijČāyÿāĀC  
æŕTāçČriijNāyNéIççŽDä;Nā■RæL'NāLlèrzāRŪāyĀäyĹæŪGāžūāy■çŽDæL'ĀæIJL'èaNriijŽ

```
def manual_iter():  
    with open('/etc/passwd') as f:  
        try:  
            while True:  
                line = next(f)  
                print(line, end='')  
        except StopIteration:  
            pass
```

éĀŽāyÿæIèèōšriijN StopIteration çTlæIèæNĜçd'žè£■āzčçŽDçzŠār;āĀC  
çDūèĀNriijNāçCædIJā;āæL'NāLlā;£çTlāyĹéIçæijTçd'žçŽD next()  
āG;æTŕçŽDèŕriijNā;æ£YāRŕāžèéĀŽè£Gè£TāZdāyĀäyĹæNĜāōŽāĀijæIèæāGèōŕçzŠār;riijNæŕTāçC  
None āĀC āyNéIçæYŕçd'žā;NriijŽ

```
with open('/etc/passwd') as f:  
    while True:  
        line = next(f, None)  
        if line is None:  
            break  
        print(line, end='')
```

## èõléõž

ad' gad' Žæ Træ ČĚā Eṭāy Nrij Næ LŠāznāij Žā; ɛç Tl for ā; lç Őrēf ■ā Rēc Tlæ Iēē A ■ā ŐEāy Āāylā Rrēf ■āzčāržē.  
ā; Eæ Yřij Nā Aūār Tāz šé IJĀēç Aāržēf ■āzčā AŽæ Žt' ā Lāçš; çā oç ŽDæ Őgā Lūij Nēf Zæ Ūūā ĀZāž Eēgčāž Tās Cēf ■  
āy Nē Içç ŽDāžd' āž Šçd' žā; Nā RŠæ LŠāznāij Tçd' žāž Eēf ■āzčæ IJ šé Ūt' æ L' Āā RŠç Tšç ŽDāšžæ IJ nçz Eē LČij.

```
>>> items = [1, 2, 3]
>>> # Get the iterator
>>> it = iter(items) # Invokes items.__iter__()
>>> # Run the iterator
>>> next(it) # Invokes it.__next__()
1
>>> next(it)
2
>>> next(it)
3
>>> next(it)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
StopIteration
>>>
```

æ IJ nçāæ Őēāy Næ Iēā Gāār Rē LČāij Žæ Žt' æ ūsā Ēēç ŽDē ōšēgčēf ■āzčç Žyā Ēšæ L' Āæ IJ řij Nā L' ■ā RŘæ Yřā; ā  
æ L' Āāžēçā oāflā; āā ūšçz Ræ L' Lēf Žçnāç ŽDā EĀōžç L' çç L' cē ōrā IJ lāf Čāy ■ā ĀČ

## 6.2 4.2 āzčç RĒēf ■āžč

### éŪōécY

ā; āæ dDāžžāž Eāy Āāylē Gĥāō Žāz L' āōžā Žlāržēsārij Nē GŊē Içā NĒā Rŋæ IJ L' ā L' Ūēā lā ĀĀā ĒČçz Dæ L' Ūā Ēūāz Ū  
ā; āæ Čšç Žt' æ Őēā IJ lā; āç ŽDēf Žāylæ Ūrā ōžā Žlāržēsāy Læ L' gēā Nēf ■āzčæ Š ■ā; IJ ā ĀČ

### ēgčā Eşæ Ūzæā L

āōdē ŽĒāy Lā; āār lē IJĀēç Aāō Žāz L' āy Āāyl  
æ Ūzæş Třij Nār Eēf ■āzčæ Š ■ā; IJ āzčç RĒā L' rā ōžā Žlā EĒē Člç ŽDāržēsāy Lā Őžā ĀČær Tāē Čřij Ž

```
class Node:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)
```

```

def __iter__(self):
    return iter(self._children)

# Example
if __name__ == '__main__':
    root = Node(0)
    child1 = Node(1)
    child2 = Node(2)
    root.add_child(child1)
    root.add_child(child2)
    # Outputs Node(1), Node(2)
    for ch in root:
        print(ch)

```

Python 3.6.0 2017-12-23 14:00:00.000000
 `__iter__()`
 Python 3.6.0 2017-12-23 14:00:00.000000
 `__next__()`

## 6.3 4.3

Python 3.6.0 2017-12-23 14:00:00.000000
 `__iter__()`
 Python 3.6.0 2017-12-23 14:00:00.000000
 `__next__()`

## 6.3 4.3

## 6.3 4.3

Python 3.6.0 2017-12-23 14:00:00.000000
 `__iter__()`
 Python 3.6.0 2017-12-23 14:00:00.000000
 `__next__()`

## 6.3 4.3

Python 3.6.0 2017-12-23 14:00:00.000000
 `__iter__()`
 Python 3.6.0 2017-12-23 14:00:00.000000
 `__next__()`

```

def frange(start, stop, increment):
    x = start
    while x < stop:
        yield x
        x += increment

```

```
>>> for n in frange(0, 4, 0.5):
...     print(n)
...
0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
>>> list(frange(0, 1, 0.125))
[0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875]
>>>
```

äyÄäylāG;æTṛäy■IJÄëAæIJL'äyÄäyl y i e l d e r■āRēā■šāRfāEāEūē;ñæ■cāyžāyÄäylçTšæLRāZlāĀC  
 èušæZōēĀZāG;æTṛäy■āRŇçZDæYřiiJŇçTšæLRāZlāRlëČ;çTlāžŌëf■āžçæS■ā;IJāĀC  
 äyNéIcæYřäyÄäylāōdēlNīijNāRŠā;āāsTçd'žēfZæuōçZDāG;æTṛāžTāšCāuēä;IJæIJžāLūiiJZ

```
>>> def countdown(n):
...     print('Starting to count from', n)
...     while n > 0:
...         yield n
...         n -= 1
...     print('Done!')
...

>>> # Create the generator, notice no output appears
>>> c = countdown(3)
>>> c
<generator object countdown at 0x1006a0af0>

>>> # Run to first yield and emit a value
>>> next(c)
Starting to count from 3
3

>>> # Run to the next yield
>>> next(c)
2

>>> # Run to next yield
>>> next(c)
1
```

```
>>> # Run to next yield (iteration stops)
>>> next(c)
Done!
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
StopIteration
>>>
```

äyÄäyłçŤšæŁŔăŹlăĜjæŤrăyžèçAçŁ'žăĴAæŸřăóCăŔłajŹăŹđăžŤăIJlèf■ăžčăy■ă;ŕçŤlăŁŕçŽĐ  
 next æŠ■ă;IJăĂĆ äyĂæŮççŤšæŁŔăŹlăĜjæŤrèŕŤăŹđéĂĂăĜziiŃNèf■ăžčçzŁæ■ćăĂĆæŁŖăžňăIJlèf■ăžčăy■é.

## 6.4 4.4 ăóđçŎřèf■ăžčăŹlă■Ŕèőő

éŮőécŸ

ăjăæČšæđĐăžžăyĂăyłèČjæŤŕæŃĂèf■ăžčæŠ■ă;IJçŽĐèĜłăóŹăžŁ'ăržèšajijŃăžúăyŃæIJZæŁ'čăŁŕăyĂăył

èğčăĚşæŮžæąŁ

çŽăăŁ'■ăyžæ■ćiiŃăIJlăyĂăyłăržèšăyŁăóđçŎřèf■ăžčæIJĂçóĂă■ŤçŽĐæŮžăijŔăŸŕăjŕçŤlăyĂăyłçŤšæ  
 ăIJl4.2ărŔèŁĆăy■iiŃăjŕçŤlNodeçşžălèèăłçđ'žăăŖăjçæŤŕæ■óçžşæđĐăĂĆăjăăŔŕèČjæČşăóđçŎřăyĂăyłăžéă  
 äyŃéłćæŸŕăžčçăAçđ'žăĴŃiiŹ

```
class Node:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

    def depth_first(self):
        yield self
        for c in self:
            yield from c.depth_first()

# Example
if __name__ == '__main__':
    root = Node(0)
    child1 = Node(1)
    child2 = Node(2)
```



```

root.add_child(child1)
root.add_child(child2)
child1.add_child(Node(3))
child1.add_child(Node(4))
child2.add_child(Node(5))

for ch in root.depth_first():
    print(ch)
# Outputs Node(0), Node(1), Node(3), Node(4), Node(2), Node(5)

```

aIJlëfŽæõřäzççäAäy■iijNdepth\_first() æŰzæşTçõÄa■TçZt'èğCăĂĆ  
 aõČëęŰaĖĹëfTăZđëĞlăûsæIJñèznázűëf■äzçæfRăyĂäyĹa■ŘëĹĆçĆzăzű  
 éĂŽëfĞërČçŤĹa■ŘëĹĆçĆzçŽĎ depth\_first() æŰzæşT(ăĵçŤĹ yield from  
 ér■aŘë)ëfTăZđărzăžTăĖČçřăăĂĆ

## ëöĹëöž

PythonçŽĎëf■äzçă■ŘëõöëęAæśCăyĂäyĹ \_\_iter\_\_() æŰzæşTëfTăZđäyĂäyĹçL'zæõĹçŽĎëf■äzçăŽĹăržësaiijN èfŽäyĹëf■äzçăŽĹăržësaaõđçŎřăžE  
 \_\_next\_\_() æŰzæşTăzűéĂŽëfĞ StopIteration aijCăyÿæăĞërĖëf■äzççŽĎaõNæĹŘăĂĆ  
 äĵEæŸřiiijNăõđçŎřëfZăžŽéĂŽäyÿaijŽærTëçÇçzAçŘăĂĆ äyNéĹcæĹSăznæijTçd'žäyNëfŽçğ■æŰzaijŘiiijNă  
 depth\_first() æŰzæşTĵijŽ

```

class Node2:
    def __init__(self, value):
        self._value = value
        self._children = []

    def __repr__(self):
        return 'Node({!r})'.format(self._value)

    def add_child(self, node):
        self._children.append(node)

    def __iter__(self):
        return iter(self._children)

    def depth_first(self):
        return DepthFirstIterator(self)

class DepthFirstIterator(object):
    '''
    Depth-first traversal
    '''

    def __init__(self, start_node):
        self._node = start_node
        self._children_iter = None

```

```

        self._child_iter = None

    def __iter__(self):
        return self

    def __next__(self):
        # Return myself if just started; create an iterator for
        ↪ children
        if self._children_iter is None:
            self._children_iter = iter(self._node)
            return self._node
        # If processing a child, return its next item
        elif self._child_iter:
            try:
                nextchild = next(self._child_iter)
                return nextchild
            except StopIteration:
                self._child_iter = None
                return next(self)
        # Advance to the next child and start its iteration
        else:
            self._child_iter = next(self._children_iter).depth_
            ↪ first()
            return next(self)

```

DepthFirstIterator ċšzǎŠNǎyŁéÍcǎ;ŁçTÍçTšæLŘǎŽÍçŽĎçL'ŁæIJñǎûěǎ;IJǎŎšçŘĚçšzǎijijřijŇ  
 ä;EǎŸřǎŏČǎĚŽēũǎĹǎ;ŁçzAçRŘřijŇǎŽǎǎyžēŁ■ǎžčǎŽÍǎŁĚéǎzǎIJĹēŁ■ǎžčǎđ'ĎçŘĚēŁĜçÍŇǎy■çzt' æŁđ' ǎđ' ġéČ  
 ǎĹççŽ;ǎĹēēŏšřijŇǎšǎǎžžǎĎŁǎĎŘǎĚŽēŁŽǎžŁǎŽēǎũĹ' çŽĎǎžččǎAǎĀČǎřEǎ;ǎçŽĎēŁ■ǎžčǎŽÍǎŏŽǎžŁ'ǎyžǎyĀǎy

## 6.5 4.5 ǎŘǎŘŠèŁ■ǎžč

éŮŏécŸ

ǎ;ǎæČšǎŘ■ǎŮzǎŘŠēŁ■ǎžčǎyĀǎyĹǎžŘǎŁŮ

èġčǎĚšǎŮzǎǎŁ

ǎ;ŁçTÍǎĚĚç;ŏçŽĎ reversed() ǎĜ;ǎTřřijŇǎřTǎçČřijŽ

```

>>> a = [1, 2, 3, 4]
>>> for x in reversed(a):
...     print(x)
...
4
3
2
1

```

āRāRŠēfāzčāzĒāzĒā;ŠāfzēsāçŽĎād'gārRāRrēcĎāĒĹçāōōŽæĹŪēĀĒāfzēsāāōđçŌrāžE  
\_\_reversed\_\_()  
āçĈādIJāyđ'ēĀĒēĈ;āyāçñēāRĹiijNēĈcā;āāĒĒēāzāĒĹāRĒāfzēsāē;ñæāçäyžäyĀäyĹāĹŪēāĹæĹāēāNīiijNāfTāēĈ

```
# Print a file backwards
f = open('somefile')
for line in reversed(list(f)):
    print(line, end='')
```

ēçAæšĹæDRçŽĎæYrāçĈādIJāRrēfāzčārzēsāāĒĈçt'āāĹĹād'ŽçŽĎēfīiijNārĒāĒūēćĎāĒĹē;ñæāçäyžäyĀ

## èóìèõž

āĹĹād'ŽçĹNāzRāSŸāzūāyāçšēēAçŠāRrāzēēĀŽēĒGāIJĹēGĹāōŽāzĹçšzāyĹāōđçŌr  
\_\_reversed\_\_() æŪzæšTæĹēāōđçŌrāRāRŠēfāzčāĀĈærTāēĈīiijŽ

```
class Countdown:
    def __init__(self, start):
        self.start = start

    # Forward iterator
    def __iter__(self):
        n = self.start
        while n > 0:
            yield n
            n -= 1

    # Reverse iterator
    def __reversed__(self):
        n = 1
        while n <= self.start:
            yield n
            n += 1

for rr in reversed(Countdown(30)):
    print(rr)
for rr in Countdown(30):
    print(rr)
```

āōŽāzĹāyĀäyĹāRāRŠēfāzčāŽĹāRrāzēā;ĒāĹŪāzčçāĀēĹđāyççŽĎēnŸæTĹiijN  
āŽāäyžāōĈāyāĒēĹIJĀēçAārĒæTřæāāāñāĒĒāĹrāyĀäyĹāĹŪēāĹāyāçĎūāRŌāĒāŌzāRāRŠēfāzčēçZāyĹāĹ

## 6.6 4.6 āyēæIJĹād'ŪēĈĹçĹŪæĀAçŽĎçTšæĹRāŽĹāĠ;æTř

### éŪōēćŸ

ā;āæĈšāōŽāzĹāyĀäyĹçTšæĹRāŽĹāĠ;æTřīiijNā;ĒæYrāōĈāijŽērĈçTĹæšRāyĹā;āæĈšæŽt'ēIJšçzŽçTĹæĹūā

## èġċaEşæŮzæaġ

æĊædIJä;äæĊşèŎl'ä;ăçŽDçTşæLRăZÍæŽt' éIJsăd' ŮéĊlçLúæĂAçzŽçTlæLüiijŃ  
ăLńăĤŸăžEă;ăăRřăžèçŎĂă■TçŽDărEăŏĊăŏdçŎřăyžăyĂăylçşzŭiijŃçDŭăRŎăĤLçTşæLRăZÍăĠ;æTřæTġăĤ  
\_\_iter\_\_() æŮzæşTăy■èĤĠăŎžăĂĊæřTăeĊiijŽ

```
from collections import deque

class linehistory:
    def __init__(self, lines, histlen=3):
        self.lines = lines
        self.history = deque(maxlen=histlen)

    def __iter__(self):
        for lineno, line in enumerate(self.lines, 1):
            self.history.append((lineno, line))
            yield line

    def clear(self):
        self.history.clear()
```

ăyžăžEă;ĤçTlèĤŽăylçşzŭiijŃă;ăăRřăžèăřEăŏĊă;ŞăAŽæŸřăyĂăylæŽŏéĂŽçŽDçTşæLRăZÍăĠ;æTřăĂĊ  
çDŭăĂŃiijŃçTşăžŎăRřăžèăĤŽăžăyĂăylăŏdăġŃăřžèşăiijŃăžŎăŸřă;ăăRřăžèèŏĤéŮăăEĤéĊlăşđæĂġăĂiijŃŃ  
æřTăeĊ history ăşđæĂġăĤŮèĂĤæŸř clear() æŮzæşTăĂĊăžçăĂAçd'žăġŃăeĊăyŃiijŽ

```
with open('somefile.txt') as f:
    lines = linehistory(f)
    for line in lines:
        if 'python' in line:
            for lineno, hline in lines.history:
                print('{}:{}'.format(lineno, hline), end='')
```

## èŎlèŏž

ăĤşăžŎçTşæLRăZÍiijŃăġŤăŏžæŸşæŎL'èĤŽăĠ;æTřæŮăăĤŸăy■èĊ;çŽDèŽŭéŸşăĂĊ  
æĊædIJçTşæLRăZÍăĠ;æTřèIJĂèçAèŭşă;ăçŽDçĤŃăžRăĤŭăžŮéĊlăĤæĤŞăžd' éAşçŽDèřl(æřTăeĊæŽt' éIJsă  
ăRřèĊ;ăiijŽărijeĤt'ă;ăçŽDăžçăĂăiijĊăyŷçŽDăd'■ăĤăĂĊ æĊædIJæŸřèĤŽçġ■ăĊĤăEĤçŽDèřlŭiijŃăRřăžèèĂĊ  
ăIJġ \_\_iter\_\_() æŮzæşTăy■ăŏŽăžĤL'ă;ăçŽDçTşæLRăZÍăy■ăiijŽæTžăRŸă;ăăžăžă;TçŽDçŏŮăşTéĂžèġŞăĂĊ  
çTşăžŎăŏĊæŸřçşçŽDăyĂéĊlăĤEiijŃăĤŸăžăăĤEăŏŷă;ăăŏŽăžĤăRĤDçġ■ăşđæĂġăŞŃæŮzæşTăĤăăġŽçTlæĤ

ăyĂăylèIJĂèçAæşġăĤRçŽDărRăIJăŮzæŸřiijŃăeĊædIJă;ăăIJlèĤ■ăžçæŞ■ă;IJæŮŭăy■ă;ĤçTlforăġĤçŎřè  
iter() ăĠ;æTřăĂĊæřTăeĊiijŽ

```
>>> f = open('somefile.txt')
>>> lines = linehistory(f)
>>> next(lines)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'linehistory' object is not an iterator
```

```
>>> # Call iter() first, then start iterating
>>> it = iter(lines)
>>> next(it)
'hello world\n'
>>> next(it)
'this is a test\n'
>>>
```

## 6.7 4.7 è■āzčāZíāŁĠçŁ'Ġ

### éŮóécŸ

äĵæČşăĹ ŮăĹrăyĂăyĥčŤséŁ■āzčāZíçŤşæĹŖçŽĐăĹĠçŁ'ĠĠăŕzéşajĭjŊăĵEæŸŕæăĠăĠĠçŁ'ĠĠăş■ăĵJăž

### èğčăEşæŮzæąĹ

ăĠĵæŤŕitertools.islice() æ■čăĕĕéĂČŤĹăžŎăĴĹĕŁ■āzčāZíăŠŊçŤşæĹŖăZíăyĹăĂăZăĹĠçŁ'ĠĠăş

```
>>> def count(n):
...     while True:
...         yield n
...         n += 1
...
>>> c = count(0)
>>> c[10:20]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'generator' object is not subscriptable

>>> # Now using islice()
>>> import itertools
>>> for x in itertools.islice(c, 10, 20):
...     print(x)
...
10
11
12
13
14
15
16
17
18
19
>>>
```

## èõléõž

è£■āzčāZlāŠNčTšæLŔāZlāy■ēČjā;£çTlāāGāGĖçŽDāLĠçLĠGæŠ■ā;IŦiijNāZāāyžāōČāznčŽDēT£āžēāžN  
āĠj;æTŦislice() è£TāZđāyĀāyġāRŕāžēçTšæLŔæNĠāōZāĖČçt'āçŽDē£■āzčāZlīijNāōČēĀZē£ĠéA■āŌĖā  
çDūāRŌæL■āijĀāgNāyĀāyġāyġçŽDē£TāZđāĖČçt'āiijNāžūçŽt'āLŕāLĠçLĠĠçzŠæġšçt'ćāijTā;■çjōāĀČ

è£ŽéGŦēēAçĬĀéG■āijžērČçŽDāyĀçČzæYŕ islice()  
āijŽæŦLēĀŪæŌL'āijāāĖēçŽDē£■āzčāZlāy■çŽDæTŦæ■ōāĀČ ā£ĖēāžēĀČēŽSāLŕē£■āzčāZlāYŕāy■āRŕéĀĖçŽ  
æL'ĀāžēāçĀēđĬā;āēĬĀēçĀāžNāRŌāĖ■āēñāēō£ēŪōē£Zāyġē£■āzčāZlçŽDērīijNēČčā;āāŕsā;ŪāĖLāŕĖāōČēČ

## 6.8 4.8 èũşè£ĠāRŕè£■āzčāržèšāçŽDāijĀāgNéČlāĬ£

### éŪŌécY

ājāæČşēA■āŌĖāyĀāyġāRŕē£■āzčāržèšāijNā;ĖæYŕāōČāijĀāgNçŽDæšRāžZāĖČçt'āā;āāžūāy■æĐšāĖt'è

### èğčāĖşæŪzæāĬ

itertools æĬāāĬŪāy■æĬJL'āyĀāžZāĠj;æTŦāRŕāžēāōNæLŔē£ZāyġāžzāĬāāĀČ  
éēŪāĖLāžNçz■çŽDæYŕ itertools.dropwhile() āĠj;æTŦāĀČā;£çTlāŪŦiijNā;āçzZāōČāijāēĀŠāyĀāy  
āōČāijZē£TāZđāyĀāyġē£■āzčāZlāŕžèšāijNāyćāijČāŌšæĬJL'āžRāĬŪāy■çŽt'āLŕāĠj;æTŦē£TāZđFlaseāžNāL'■

āyžāžĖæijTçd'zīijNāĀĠāōZā;āāĬĬēržāRŪāyĀāyġāijĀāgNéČlāĬ£æYŕāĠāēāNæşĬéĠçŽDæžRæŪĠāžūā

```
>>> with open('/etc/passwd') as f:
...     for line in f:
...         print(line, end='')
...
##
# User Database
#
# Note that this file is consulted directly only when the system is_
↳running
# in single-user mode. At other times, this information is provided_
↳by
# Open Directory.
...
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
...
>>>
```

āçĀēđĬā;āæČşèũşè£ĠāijĀāgNéČlāĬ£çŽDæşĬéĠēāNçŽDērīijNāRŕāžèç£ZæāūāĀŽīijŽ

```
>>> from itertools import dropwhile
>>> with open('/etc/passwd') as f:
...     for line in dropwhile(lambda line: line.startswith('#'), f):
```

```
...         print(line, end='')
...
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
...
>>>
```

èfZäylä;Nā■RæYrāšzāžŌæāzæ■ōæ\$RäylætNërTāGjæTṛeūšèfGāijĀāgNçŽDāĒČčt'āāĀĆ  
 æĒĆädIJā;āāūšçzRæYŌçqōçšēēAšžEēēAēūšèfGçŽDāĒČčt'āçŽDäylæTṛçŽDērīijNéCčāzLāRfāzēā;£çTī  
 itertools.islice() ælēāzčæŽfāĀĆæfTāēČīijŽ

```
>>> from itertools import islice
>>> items = ['a', 'b', 'c', 1, 4, 10, 15]
>>> for x in islice(items, 3, None):
...     print(x)
...
1
4
10
15
>>>
```

āIJlēfZäylä;Nā■Räy■īijN islice() āGjæTṛæIJĀāRŌéCčāyl None  
 āRĆæTṛæNĠāōZāžEā;āēēAēŌūāRŪāzŌçnn3äylāLṛæIJĀāRŌçŽDæL'ĀæIJL'āĒČčt'āīijN  
 æĒĆädIJ None āšN3çŽDä;■ç;ōāržērČīijNæDṚæĀlāršæYrāzĒāzĒēŌūāRŪāL'■äyL'äylāĒČčt'āæAṛæAṛçŽyāR  
 (èfZäylēūšāLĠçL'ĠçŽDçŽyāR■æš■ā;IJ [3:] āšN [:3] āŌšçRĒæYrāyĀæāūçŽD)āĀĆ

## ēōlēōž

āGjæTṛdropwhile() āšN islice() āĒūāōdāršæYrāyd'äylāyōāL'āGjæTṛīijNäyžçŽDāršæYféA£ā

```
with open('/etc/passwd') as f:
    # Skip over initial comments
    while True:
        line = next(f, '')
        if not line.startswith('#'):
            break

    # Process remaining lines
    while line:
        # Replace with useful processing
        print(line, end='')
        line = next(f, None)
```

ēūšèfGāyĀäylāRfēf■āzčāržèšaçŽDāijĀāgNéČlāLĒēūšéĀžāyççŽDēfGæzd'æYrāy■āRŌçŽDāĀĆ  
 æfTāēČīijNäyLēfṛāzčçāAçŽDçñnāyĀäylēČlāLĒāRfēČ;āijŽèfZæāūēG■āEZīijŽ

```
with open('/etc/passwd') as f:
    lines = (line for line in f if not line.startswith('#'))
```





ä|çŤİitertools.combinations() âRřăŮăĹřèĶŞăĚéZĚăŔĹăŮ■ăĚÇçťăçŽDăĹĂăİJĹçŽDçŽĹ

```
>>> from itertools import combinations
>>> for c in combinations(items, 3):
...     print(c)
...
('a', 'b', 'c')

>>> for c in combinations(items, 2):
...     print(c)
...
('a', 'b')
('a', 'c')
('b', 'c')

>>> for c in combinations(items, 1):
...     print(c)
...
('a',)
('b',)
('c',)
>>>
```

árzážŎ combinations() æİëëõšījŇăĚĈçt'ăçŽĐéąžăžŔăũščžŔăy■éĜ■èèAăžEăĂĆ  
 äžšăřšăĚŕër'tījŇčžĐăŔĹ ('a', 'b') èũş ('b', 'a')  
 âĖŭăôđăĚŕăyĂăăũçŽĐ(æİĂçžĹăŔăĭăjŽèçšăĜžăĖŭăy■ăyĂăyĭ)ăĂĆ

ãIJġēoαçøŮçzǾDãRĹŁçŽǾæŮŮãĂŽiijŃăyĂæŮęãĚĈçť'æècńéĂĹ'ăRŮăřsăijŽăzŌăĂŽéĂĹ'ăy■ăĹ'ŤéŽd'ăŌĹ'  
 èĂŃăĜj;æŤř itertools.combinations\_with\_replacement()  
 ăĚăēōŷăRŃăyĂăyĹăĚĈçť'æècńéĂĹ'ăŃĹ'ăđ'ŽăŋăiijŃăŕŤăçĈiijŽ

```
>>> for c in combinations_with_replacement(items, 3):
...     print(c)
...
('a', 'a', 'a')
('a', 'a', 'b')
('a', 'a', 'c')
('a', 'b', 'b')
('a', 'b', 'c')
('a', 'c', 'c')
('b', 'b', 'b')
('b', 'b', 'c')
('b', 'c', 'c')
('c', 'c', 'c')
>>>
```

èóíèőž

itertools

æfZäyÄärRèLCæLŚazñāŘŠajaašTçd'žçŽDzĚäzĚæYř  
ælaaiUčŽDäyÄeĆlálLEaŁšëČ;ãĂ Ćr;ćoaq;ääzšăŔrăzeeĞłauśeL'NăŁlăođčŎřæŐśăLŮczĐăŘŁcŮUaesTiiǵNă

ā;ŠæĹŚāzñčřāĹŕçIJŇäyĹāŌzæIJĹ'āžŽāđ'■æĹCçŽDēf■āžcéŮóécŸæŮüiijŇæIJĀāē;āŖřāžčāĚĹāŌžçIJŇçIJŇŇ  
āēČæđIJēfŽāyĹēŮóécŸā;ĹæŽóéA■iijŇéČčāžĹā;ĹæIJĹ'āŖřèČ;āijŽāIJĹéGŇéĹæĹ;āĹŕèğčāEşæŮžæāĹiijA

## 6.10 4.10 āžŖāĹŮäyĹçŕ'čāijŢāĀijè£■āžč

### éŮóécŸ

ā;āæČşāIJĹē£■āžčāyĀäyĹāžŖāĹŮçŽDāŖŇæŮüèùşèyĹæ■čāIJĹēčŇāđ'ĐçŖEçŽDāĚČçŕ'ăçŕ'čāijŢāĀČ

### èğčāEşæŮžæāĹ

āEĚç;őçŽD enumerate() āĠ;æŢŕāŖřāžčā;Ĺāē;çŽDèğčāEşçēfŽāyĹēŮóécŸiijŽ

```
>>> my_list = ['a', 'b', 'c']
>>> for idx, val in enumerate(my_list):
...     print(idx, val)
...
0 a
1 b
2 c
```

äyžāžEæŇĹ'āijāçzşēāŇāŖüè;ŞāĠž(ēāŇāŖüāžŌĹāijĀāğŇ)iiijŇā;āāŖřāžčāijāēĀŠāyĀäyĹāijĀāğŇāŖČæŢŕ

```
>>> my_list = ['a', 'b', 'c']
>>> for idx, val in enumerate(my_list, 1):
...     print(idx, val)
...
1 a
2 b
3 c
```

è£Žçğ■æČĚāĒāIJĹā;āēA■āŌEæŮĠžāžūæŮüæČşāIJĹéŢŽèŕŕæūĹæAŕäy■ā;£çŢĹēāŇāŖüāđŽā;■æŮüāĀŽéĹ

```
def parse_data(filename):
    with open(filename, 'rt') as f:
        for lineno, line in enumerate(f, 1):
            fields = line.split()
            try:
                count = int(fields[1])
                ...
            except ValueError as e:
                print('Line {}: Parse error: {}'.format(lineno, e))
```

enumerate() āŕžāžŌèùşèyĹæşŖāžŽāĀijāIJĹāĹŮēāĹāy■āĠžçŖçŽDā;■ç;őæŸŕā;ĹæIJĹçŢĹçŽDāĀČ  
æĹĀāžēiijŇāēČæđIJā;āæČşāŕEäyĀäyĹæŮĠžāžūāy■āĠžçŖçŽDā■Ţēr■æŸāāŕĐāĹŕāóČāĠžçŖçŽDēāŇāŖüāy  
enumerate() æĹēāđŇāĹŖiijŽ

```
word_summary = defaultdict(list)

with open('myfile.txt', 'r') as f:
    lines = f.readlines()

for idx, line in enumerate(lines):
    # Create a list of words in current line
    words = [w.strip().lower() for w in line.split()]
    for word in words:
        word_summary[word].append(idx)
```

æĊædIĲăăd'ĎċŘĖăōŇæŮĠăzŭăŔŌăL'Şăŗ word\_summary  
 ĩijŇăijŽăŔŚċŎŕăōĈăŸŕăŸĂăŸlă■ŮăĚŷ(ăĠĖċăŏăĬĕĕŏŝăŸŕăŸĂăŸl defaultdict  
 )ĭijŇăŕzăžŎăŕŔăŸlă■Ŧĕŕ■ăĬĴ'ăŸĂăŸl key ĩijŇăŕŔăŸl key  
 âŕzăžŦċŽĎăĀĭăŸŕăŸĂăŸlċŦŝĕĤăŸlă■Ŧĕŕ■ăĠžċŎŕċŽĎăŇăŔŭċžĎăĹŔċŽĎăĹŮĕăĴăĀĈ  
 æĊædIĲăŝŔăŸlă■Ŧĕŕ■ăĬĴăŸĂĕăŇăŸ■ăĠžċŎŕĕĤĠăŷd' æŋăĭijŇĕĈăžĹĕĤăŸlĕăŇăŔŭăžŝăĭijŽăĠžċŎŕăŷd' æŋăĭ  
 âŔŇăŮăăžŝăŔŕăžĕăĬăŸăžăŮĠăĴŇċŽĎăŸĂăŸlċŦă■ŦċžŝĕŏăĴăĀĈ

## èŏĬĕŏž

ăĭŞăĭăæĈŝĕċĬăd' ŮăŏŽăžĹăŸĂăŸlĕŏăŕŦŕăŔŸĕĠŔċŽĎăŮăĂŽĭijŇăĭĤĭĬ  
 enumerate() âĠăŕŦŕăĭjŽăŽŦăĹăŏŦă■ŦăĀĈăĭăăŔŕĕĈĭĭjŽăĈŔăŸŇĕĬĕĖŽăăŭăĖŽăžċċăĀĭijŽ

```
lineno = 1
for line in f:
    # Process line
    ...
    lineno += 1
```

ăĭĖăŸŕăæĊædIĲăĭĤĬĬ enumerate() âĠăŕŦŕăĬăžăĈăŽăŕŝăŸăĭŮăŮăŽŦăĹăăĭijŸĕŽĖăžĖĭijŽ

```
for lineno, line in enumerate(f):
    # Process line
    ...
```

enumerate() âĠăŕŦŕĕĤŦăŽċŽĎăŸŕăŸĂăŸl enumerate âŕžĕŝăăŏđăĭŇĭijŇă  
 âŏĈăŸŕăŸĂăŸlĕŏă■ăžċăŽĭijŇĕĤŦăŽĎĕĤċž■ċŽĎăŇĕăŔŇăŸĂăŸlĕŏăŕŦŕăŝŇăŸĂăŸlăĀĭjċŽĎăĖĈċžĎĭijŇă  
 âĖĈċžĎăŸ■ċŽĎăĀĭjĕĂŽĕĤĠăĬăĭjăăĖĕăžŔăĹŮăŸĹĕŕĈċŦĬĬ next() ĕĤŦăŽĎăĀĈ

ĕĤŸăĬĴ'ăŸĂĈĈăŔŕĕĈĭăžŭăŸ■ăĭĹĕĠ■ĕĖĀĭijŇăĭĖăŸŕăžŝăĀĭjăĭŮăŝĹăĎŔĭijŇă  
 æĬĴ'ăŮăăĂŽăĭŞăĭăăĬĴăŸĂăŸlăŭŝċžŔĕġċăŎŇăŔŎċŽĎăĖĈċžĎăžŔăĹŮăŸĹăĭĤĬĬ  
 enumerate() âĠăŕŦŕăŮăăĭĹăŏžăŸŝĕŕĈăĖĕĖŽăĕŸŝăĀĈ  
 äĭăăĭŮăĈŔăŸŇĕĬĕă■ċċăŏċŽĎăŮăĭjŔĕĤăăŭăĖŽăžĭijŽ

```
data = [ (1, 2), (3, 4), (5, 6), (7, 8) ]

# Correct!
for n, (x, y) in enumerate(data):
    ...
```

```
# Error!
for n, x, y in enumerate(data):
    ...
```

## 6.11 4.11 áĤŃæŮúè£■āzčāđ'ŽäyłāžŔāĹŮ

### éŮóécŸ

äĵāæČšāŔŃæŮúè£■āzčāđ'ŽäyłāžŔāĹŮĭĵŃæŕŔæŋāāĹĒāĹŋāžŎäyĀäyłāžŔāĹŮäy■āŔŮäyĀäyłāĚČčŕ'āāĀ

### èğčāĒşæŮzæąĹ

äyžāžĒāŔŃæŮúè£■āzčāđ'ŽäyłāžŔāĹŮĭĵŃäĵčŦĭ zip() āĢĵæŦŕāĀĈæŕŦäęĈĭĵŽ

```
>>> xpts = [1, 5, 4, 2, 10, 7]
>>> ypts = [101, 78, 37, 15, 62, 99]
>>> for x, y in zip(xpts, ypts):
...     print(x, y)
...
1 101
5 78
4 37
2 15
10 62
7 99
>>>
```

zip(a, b) äĭjŽčŦşæĹŔäyĀäyłāŕŕèŦŦāŽđāĚČčžĎ (x, y)  
čŽĎè£■āzčāŽĭĵŃāĚŮäy■xæĹèèĢĭĵŃyæĹèèĢĭāĀĈ äyĀäŮęāĚŮäy■æşŔäyłāžŔāĹŮāĹŕāžŦčžşāŕĵĭĵŃè£■āz  
āŽāæ■đ'è£■āzčēŦ£āžęèŮşāŔĈæŦŕäy■ĪĀçş■āžŔāĹŮèŦ£āžęäyĀèĢŕ'āĀĈ

```
>>> a = [1, 2, 3]
>>> b = ['w', 'x', 'y', 'z']
>>> for i in zip(a,b):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
>>>
```

āęĈāđĪēŦŽäyłäy■æŸŕāĵāæČšęęĀçŽĎæŦĹæđĪĭĵŃēĈāžĹèŦŸāŕŕāžēäĵčŦĭ  
itertools.zip\_longest() āĢĵæŦŕæĹēāžčæŽĒāĀĈæŕŦäęĈĭĵŽ

```
>>> from itertools import zip_longest
>>> for i in zip_longest(a,b):
...     print(i)
```

```
...
(1, 'w')
(2, 'x')
(3, 'y')
(None, 'z')

>>> for i in zip_longest(a, b, fillvalue=0):
...     print(i)
...
(1, 'w')
(2, 'x')
(3, 'y')
(0, 'z')
>>>
```

èóíèőž

a1\$äjačšæĹřáržad'DčŘEæTřæ■óčŽDæUúáĀŽ zip()  
 aĜjæTřæYřaĹLæIJLčTĹčŽDāĀČ ærTæČiiĵNāAGēōĵä;āad't'āĹUēāĹāŠNäyĀäyĹāĀijaĹUēāĹiĵNāřsāČRäyNéĹ

```
headers = ['name', 'shares', 'price']
values = ['ACME', 100, 490.1]
```

ä:ʃcTÍzip()äRfrazëöl'ä:ääRĖaőČaznæL'SăNĚázúčTšæLRäyÄäyła■UäĚyijŽ

```
s = dict(zip(headers, values))
```

æŁŨèĂĚä;äázşăŔŕäzëăĈŔäyŊéİcèfŻæăũăžğçŦşè;ŞăĜziiŹ

```
for name, val in zip(headers, values):
    print(name, '=', val)
```

ěŽįćǦũäy■äyÿëġAīijŃäJĖæŸř z ĩ p ( ) āŖřäzēæŌēāŖŮād'ŽāžŌäyđ'äylčŽǦāžŖāĹŮčŽǦāŖĆæŤřāĀĆ  
 èĹžæŮūāĀžæĹ'ĀčŤšæĹŖčŽǦčzŠæđĪāĒĈčzǦäy■āĒĈct'ääylæŤřēušë;ŠāĒēāžŖāĹŮäylæŤřäyĀæūāĀĆæŖŤ

```
>>> a = [1, 2, 3]
>>> b = [10, 11, 12]
>>> c = ['x', 'y', 'z']
>>> for i in zip(a, b, c):
...     print(i)
...
(1, 10, 'x')
(2, 11, 'y')
(3, 12, 'z')
>>>
```

æIJÅŘŎajzērČäYĂçCzârſæYřijŇ zip() äijŽǎĹZǎzžyĂäylē■āzčǎZlæIēāIJäyžczŞædIjēŦǎZđāĂĆ  
ăĖĆædIJājăIEIJăĖĖAărĖçzŞărzçŽĐăAija■YăCǎIăIJǎĹŮēqläy■ijŇēĖAă;ŦçTĪ list()  
ăĖJæTřăĂĆærTăĖĆijŽ

```
>>> zip(a, b)
<zip object at 0x1007001b8>
>>> list(zip(a, b))
[(1, 10), (2, 11), (3, 12)]
>>>
```

## 6.12 4.12 äÿ■āŖŇéŽĚāŖĹäÿŁāĚČŧ'ăçŽĎèŁ■ăžč

### éŮóécŸ

äĵăæČšāIJĹăđ'ŽăÿĹăŕžèšăæL'gèāŇçŽÿāŖŇçŽĎăŞ■ăĵIJĵĵŇăĵEăŸŕèŁŽăžŽăŕžèšăāIJĹăÿ■āŖŇçŽĎăóžăŽĹă

### èğčăĚşăŮžăęĹ

itertools.chain() æŮžăşŧăŖŕăžèçŧĹăĹèçóĀăŇŮèŁŽăÿĹăžžăŁăăĂČ  
 āóČăŎěāŖŮăÿĂăÿĹăŖŕèŁ■ăžčăŕžèšăāĹŮèăĹăĴIJăÿžèŁŞăĚēĵĵŇăžžŭèŁŧăŽđăÿĂăÿĹèŁ■ăžčăŽĹĵĵŇăIJĹăŧĹçŽĎ  
 äÿžăžĚăĵĴçđ'žăÿĚăēŽĵĵŇèĂČèŽŚăÿŇéĹèŁŽăÿĹăĴŇă■ŖĵĴ

```
>>> from itertools import chain
>>> a = [1, 2, 3, 4]
>>> b = ['x', 'y', 'z']
>>> for x in chain(a, b):
...     print(x)
...
1
2
3
4
x
y
z
>>>
```

äĵĴçŧĹ chain() çŽĎăÿĂăÿĹăÿÿèğĀăIJžăŽŕăŸŕăĴşăĵăæČšăŕžăÿ■āŖŇçŽĎéŽĚāŖĹäÿ■ăL'ĂăIJĹăĚČç

```
# Various working sets of items
active_items = set()
inactive_items = set()

# Iterate over all items
for item in chain(active_items, inactive_items):
    # Process item
```

èŁŽçğ■èğčăĚşăŮžăęĹèĚĀăŕŧăČŖăÿŇéĹèŁŽăăŭăĴçŧĹăÿđ'ăÿĹă■ŧçŇçŽĎăĴçŎŕăŽŧ'ăŁăăĵĴŸéŽĚĵĵŇă

```
for item in active_items:
    # Process item
```

```

...

for item in inactive_items:
    # Process item
...

```

## ěóíěőž

`itertools.chain()` æŌěâŔŮäyÄäylæĹŮad'ŽäylâŔřef■äzcâržèsqæIJÄäyžèĭŞăĚěâŔCæŦřăĂĆ  
 çĎúâŔŌăĹZăzzäyÄäylæf■äzcâŽŕijNăĭIæñæfđcz■çŽĎěŦTăŽđæŦŔäylâŔřef■äzcâržèsqäy■çŽĎăĚĆçŦ'ăăĂĆ  
 èŦŽçg■æŮžaijŔëçAæŦTăĚĹăŦĚăžŔăĹŮăŔĹăžŭăĚ■èf■äzcèçAénŸæŦĹçŽĎăd'ŽăĂĆæŦTăçŦijŽ

```

# Inefficient
for x in a + b:
    ...

# Better
for x in chain(a, b):
    ...

```

çñnäyĂçg■æŮžæqĹäy■ijNă + b æŞ■ăĭIJaijŽăĹZăzzäyÄäylăĚĭæŮŕçŽĎăžŔăĹŮăžŭèçAæśCăăŦŦbçŽ  
 chian() äy■aijŽæIJĹèfŽäyĂæ■ëijNăĹĂăžěæÇæđIJèĭŞăĚěăžŔăĹŮăĬăyŷăđ'gçŽĎæŮŭăĂŽăijŽăĭĹçIJJA  
 âžŭäyŦăĭŞăŔřef■äzcâržèsqçşăđNăy■äyĂæăŭçŽĎæŮŭăĂŽ chain()  
 âŦŦNăăŭăŦŕăžèăĭĹăçĭçŽĎăŭăăĭIJăĂĆ

## 6.13 4.13 âĹZăzzæŦřæ■óad'ĎçŔĚçóæéAŞ

### éŮóécŸ

ăĭăæČşăžæŦřæ■óçóæAŞ(çşžăijijUnixçóæéAŞ)çŽĎæŮžaijŔëf■äzcăđ'ĎçŔĚæŦřæ■óăĂĆ  
 æŦTăçŦijNăĭăæIJĹăylăđ'gçĠŔçŽĎæŦřæ■óéIJăèçAăđ'ĎçŔĚĭijNăĭĚæŸŕăy■èÇĭârĚăđCăžñäyĂæñæĂğæŦĭ

### èğcăĚşæŮžæqĹ

çŦşæĹŔăŽĭăĠĭæŦřæŸŕăyÄäylăôđçŎŕçóæéAŞæIJžăĹŮçŽĎăçĭăĹđæşŦăĂĆ  
 äyžăžĚæijŦçđ'žijNăAĠăđŽăĭăèçAăđ'ĎçŔĚäyÄäylăĬăyŷăđ'gçŽĎæŮèăfŮăŮĠăžŭçŽóăĭŦijŽ

```

foo/
  access-log-012007.gz
  access-log-022007.gz
  access-log-032007.gz
  ...
  access-log-012008
bar/
  access-log-092007.bz2

```

```
...
access-log-022008
```

åAĞeö;æŕRäylæŮëåŁŮæŮĞäzúåŇĖåŔñëŁZæăŭçŽĐæŦŕæ■ŏiijŽ

```
124.115.6.12 - - [10/Jul/2012:00:18:50 -0500] "GET /robots.txt ..."
↳200 71
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /ply/ ..." 200
↳11875
210.212.209.67 - - [10/Jul/2012:00:18:51 -0500] "GET /favicon.ico ..
↳." 404 369
61.135.216.105 - - [10/Jul/2012:00:20:04 -0500] "GET /blog/atom.xml
↳..." 304 -
...
```

äyžāZĖād'ĐçŔĖëŁŽăžZæŮĞäzūiijŇă;ăăŔŕäzëăőŽăzL'äyĂäylçŦsăd'ŽăylæL'ğëąŇçL'żăőŽăzżăŁaçŇňçŇŇ

```
import os
import fnmatch
import gzip
import bz2
import re

def gen_find(filepat, top):
    '''
    Find all filenames in a directory tree that match a shell
    ↳wildcard pattern
    '''
    for path, dirlist, filelist in os.walk(top):
        for name in fnmatch.filter(filelist, filepat):
            yield os.path.join(path, name)

def gen_opener(filenamees):
    '''
    Open a sequence of filenames one at a time producing a file
    ↳object.
    The file is closed immediately when proceeding to the next
    ↳iteration.
    '''
    for filename in filenamees:
        if filename.endswith('.gz'):
            f = gzip.open(filename, 'rt')
        elif filename.endswith('.bz2'):
            f = bz2.open(filename, 'rt')
        else:
            f = open(filename, 'rt')
        yield f
        f.close()

def gen_concatenate(iterators):
    '''
```



çÖřǺIJłǻǻāŔřäzēăĹŁăőžæÿŞşÇŽĐărÈęēŻăżZăĞĭæȚrēfđētũæİeăLŽăzzăyĂăylăd’ĐçŘEçőăéAșăĂĆ  
ærȚăeÇiiJňăyžăžæEăşěæLĭăŇĖăŔňă■Țēr■pythonçŽĐæL’ĂæIJL’æŮěăfŮeăNiiJňăǻāŔřäzēēēŻăăũăAŽiiJŽ

æCædIjârEæiëçŽDæUûaĀZä;æČšæL’ásTçóæAŞijŇä;áčŤŽèGšâRřæzâIjčŤšæLŘâZléaèççâijRäy  
 ærTæCrijŇäyNéiçèfZäyčL’LæIjñèðaçŮâGžaijæ;šČŽDâUèŁCæTrázüèðaçŮâEûæĀzâŠŇāĀĆ

èóíèőž

äzëçðæAŞæŨzâijRâd'DçŘæTŗæ■ôâRřazëçTlælēëgçâEşâRĎçşzaĖŭazŨéŨôécYrijNâNĖæNñëgçædR  
 äyžāZĖçŘĖĖgçāyLēfřāzççāAīijNēĠçĆzæYřēæAæYŎçZj yield  
 èř■āRēā;IJāyžæTŗæ■ôçŽĎçTşāžgèAĖèĀN for ā;łçŎřèř■āRēā;IJāyžæTŗæ■ôçŽĎæŭLet'zèĀĖāĀĆ  
 ā;ŞēfZāzŽçTşæLŔāŽlēcñēfđāIJāyĀetūāRŎīijNæfRāyl yield  
 āijŽārĖāyĀāylā■TçNñçŽĎæTŗæ■ôāĖĈçt'āāijāēĀŞçzZēf■āzçād'DçŘĖçðæAŞçŽĎāyNāyĀēYŭæðtāĀĆ  
 āIJlā;Nā■RæIJāāRŎēĆlāLērijN sum() āĠ;æTŗæYřæIJāçzŁçŽĎçlNāžRēf'sāLlēĀĖīijNæfRāæñāžŎçTşæL  
 èfZçg■æŨzâijRāyĀāylēlđāyŷāē;çŽĎçL'zçĆzæYřæfRāylçTşæLŔāŽlāĠ;æTŗā;LārRāzŭāyTēČ;æYřçNñ  
 ā;Łād'ZæŨūāĀZrijNēfZāzZāĠ;æTŗæÇæđIJæfTē;ČēĀŽçTlçŽĎēlāRřāzēāIJlāĖŭazŨāIJzæZřēĠād'■ā;łç  
 āzŭāyTæIJāçzLārĖēfZāzŽçZĎāzŭçDāRĬlētūælēcŽĎāzççāAçIJNāyLāŎzēlđāyŷçōĀā■TijjNāzşā;ŁāōzæYŞ

ä;£çTlèfZçg■æÚzàiRçZDàEĖā■YæTlçŌGāzšāy■ā; Ūāy■æRŘāĀCāyLèfřāzččāAā■sä;£æYřāIJlāyĀāy  
āzNāōđāyLūijNçTśāzŌā;£çTlāzEēf■āzčæÚzàiRād'DçŘEīijNāzččāAēfRēāNēfGçlNāy■āRlēIJĀēēAā;LārRā

āIJlērCçTl gen\_concatenate() āG;æTřçZDæŪūāĀZā;āāRrēČ;āijZæIJL'āzZāy■ād'læYŌçZ;āĀC  
èfZāyġāG;æTřçZDçZōçZDæYřārEē;SāĖēāzRāLŪāNijæŌēæLŘāyĀāyġā;LéTfçZDēāNāzRāLŪāĀC  
itertools.chain() āG;æTřāRŊNæāūæIJL'çśzāijijçZDāLšèČ;īijNā;EæYřāōČēIJĀēēAārEæL'ĀæIJL'āRrē  
āIJlāyLēlçèfZāyġā;Nā■Rāy■īijNā;āāRrēČ;āijZāEŻçśzāijijēfZæāūçZDēr■āRē  
lines = itertools.chain(\*files) īijN ēfZārEārījēGt'  
gen\_opener() çTšæLŘāZlēcāRŘāL'■āĖlēcāūLèt'zæŌL'āĀC ā;EçTśāzŌ  
gen\_opener() çTšæLŘāZlērRæñaçTšæLŘāyĀāyġāL'SāijĀēfGçZDæŪGāzūīijN  
ç■L'āLřāyNāyĀāyġēf■āzčæ■ēēl'd'æŪūæŪGāzūārśāĖsēŪ■āzEīijNāZāæ■d' chain()  
āIJlēfZēGŊāy■ēČ;ēfZæāūā;£çTlāĀC āyLēlççZDæŪzæāLārřāzēēAāĖ■ēfZçg■æČĖāEġāĀC

gen\_concatenate() āG;æTřāy■āGzçŌrēfG yield from ēr■āRēīijNāōČārE  
yield æS■ā;IJāzčçRĖāLřçLūçTšæLŘāZlāyLāŌzāĀC ēr■āRē yield from  
it çŌĀā■TçZDēfTāZđçTšæLŘāZl it æL'ĀāzğçTšçZDæL'ĀæIJL'āĀijāĀC  
āĖšāzŌēfZāyġāLŚāzñāIJl4.14ārRēLČāijZæIJL'æZt'ēfZāyĀæ■ēçZDæRŘēfřāĀC

æIJāRŌēfYæIJL'āyĀçČzéIJĀēēAæślæDŘçZDæYřīijNçŌāēAşæŪzàiRāzūāy■æYřāyGēČ;çZDāĀC  
æIJL'æŪūāĀZā;āæČşçñNā■şād'DçŘEæL'ĀæIJL'æTřæ■ōāĀC çDūēĀNīijNā■sä;£æYřēfZçg■æČĖāEġīijNā;£

David Beazley āIJlāzŪçZD Generator Tricks for Systems Programmers  
æTřçlNāy■ārřāzŌēfZçg■æL'ĀæIJræIJL'ēlđāyÿæūsāĖēçZDēōšēgčāĀCāRřāzēāRČēĀČēfZāyġāTřçlNēŌūār

## 6.14 4.14 āśTāijĀātNāēŪçZDāzRāLŪ

### éŪŌēćY

ā;āæČşārEāyĀāyġād'ZāsČātNāēŪçZDāzRāLŪāsTāijĀæLŘāyĀāyġā■TāsČāLŪēāġ

### ēgčāEşæŪzæāġ

ārřāzēāEŻāyĀāyġāNēĀRñ yield from ēr■āRēçZDēĀšā;ŠçTšæLŘāZlāēē;zæġēgčāEşēfZāyġēŪŌēć

```
from collections import Iterable

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_
→types):
            yield from flatten(x)
        else:
            yield x

items = [1, 2, [3, 4, [5, 6], 7], 8]
# Produces 1 2 3 4 5 6 7 8
for x in flatten(items):
    print(x)
```

```

    if isinstance(x, Iterable):
        yield from flatten(x, ignore_types)
    else:
        yield x

```

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

```

>>> items = ['Dave', 'Paula', ['Thomas', 'Lewis']]
>>> for x in flatten(items):
...     print(x)
...
Dave
Paula
Thomas
Lewis
>>>

```

## 6.15

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

## 6.15 4.15

### 6.15

```

def flatten(items, ignore_types=(str, bytes)):
    for x in items:
        if isinstance(x, Iterable) and not isinstance(x, ignore_types):
            for i in flatten(x):
                yield i
        else:
            yield x

```

## èġċaEşæÚzæaĹ

heapq.merge() aĜjæTŗāRfāzēāyōājæġċaEşæfZāyĹéUōécYāĀCærTāeĆrijZ

```
>>> import heapq
>>> a = [1, 4, 7, 10]
>>> b = [2, 5, 6, 11]
>>> for c in heapq.merge(a, b):
...     print(c)
...
1
2
4
5
6
7
10
11
```

## éőĹéőZ

heapq.merge aRrēf■āzčĹ'zæĀġæDŖāŚşçİĀāōČāy■āijZçñNēĹ' nērzaŖŪæL' ĀæIJL' āzŖāĹŪāĀĆ  
ēfZārsæDŖāŚşçİĀājāāRfāzēāIJĹéīdāyŷēTfçZDāzŖāĹŪāy■āj;fçTĹāōĆrijNēĀNāy■āijZæIJL' ād' ĩad' ġçZDāijĀe  
ærTāeĆrijNāyNēīcæYŖāyĀāyĹā;Nā■ŖāĹēāijTçd' zāeČājTāŖĹāzūāy'd' āyĹæŌŠāzŖæŪĜāzūrijZ

```
with open('sorted_file_1', 'rt') as file1, \
    open('sorted_file_2', 'rt') as file2, \
    open('merged_file', 'wt') as outf:

    for line in heapq.merge(file1, file2):
        outf.write(line)
```

æIJL'āyĀçĆzēēAāijžērČçZDæYŖheapq.merge() éIJĀēēAæL' ĀæIJL'è;ŞāĒēāzŖāĹŪāfĒēāzæYŖæŌŠ  
çĹ'zāĹnçZDrijNāōČāzūāy■āijZēcDāĒĹērzaŖŪæL' ĀæIJL'æTŗæ■ōāĹŖāāEæāĹāy■æĹŪēĀĒēcDāĒĹæŌŠāzŖrij  
āōČāzĒāzĒæYŖæčĀæşēæL' ĀæIJL' āzŖāĹŪçZDāijĀāġNēČĹāĹēāzūēfTāZdæIJĀārŖçZDēĆçāyhijNēfZāyĹēfČ

## 6.16 4.16 èĤ■āzčāZĹāzčæZĤwhileæŪāéZŖā;ĹçŌŖ

### éUōécY

ājāāIJĹāzčçāAāy■āj;fçTĹ while āj;ĹçŌŖæĹēēf■āzčād' DçŖEæTŗæ■ōrijNāZāyŷzāōČéIJĀēēAērČçTĹæşŖāy  
èČjāy■èČjçTĹēf■āzčāZĹāĹēēĜ■āEŖēfZāyĹāj;ĹçŌŖāŚçrijş

## èġċaEşæÚzæaĹ

āyĀāyĹāyŷēēAçZDIOæŞ■ājIJĹĹNāzŖāŖŖēČjāijZæČşāyNēīcèfZæāūrijZ

```
CHUNKSIZE = 8192

def reader(s):
    while True:
        data = s.recv(CHUNKSIZE)
        if data == b'':
            break
        process_data(data)
```

èŁŻçġ■āzčċăĀéĀŽāÿÿăŔŕāzēă;ŁçŦĬ iter() æĬēāzčæŽĭijŇăĉCăÿŇăĽĀčđ'žĭijŽ

```
def reader2(s):
    for chunk in iter(lambda: s.recv(CHUNKSIZE), b''):
        pass
    # process_data(data)
```

ăĉĆăđĬă;ăăĀĀĈŨŖăŏĈăĽŕăžŦēĈ;ăÿ■ēĈ;æ■čăÿÿăŭēă;ĬĭijŇăŔŕāzēērŦēĭŇăÿŇăÿĀăÿĽĉŏĀă■ŦĉŽĎă;Ňă

```
>>> import sys
>>> f = open('/etc/passwd')
>>> for chunk in iter(lambda: f.read(10), ''):
...     n = sys.stdout.write(chunk)
...
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/
↪uucico
...
>>>
```

## èŏĬēŏž

iter   ăĜ;æŦŕăÿĀăÿĽēŖĬăÿžăžžçšĉŽĎĈĽ'žăĀĝăŸŕăŏĈăĖŏăŔŨăÿĀăÿĽăŔŕēĀĽ'çŽĎ  
callable   ăržēŖăăŖŇăÿĀăÿĽăăĜēŏŕ(çžŖăŕĭ)ăĀĭă;Ĭăÿžē;ŖăĖēăŔĈăŦŕăĀĈ  
ă;ŖăžēēŁŻçġ■æŨžăĭŔă;ŁçŦĬçŽĎæŨŭăĀŽĭijŇăŏĈăĭjŽăĽŽăžžăÿĀăÿĽē■āzčăŽĭijŇ  
èŁŽăÿĽē■āzčăŽĭăĭjŽăÿ■æŨēŕĈŦĬ callable āržēŖăçŽŦ'ăĽŕēŦŦăŽđăĀĭăŖŇăăĜēŏŕăĀĭjçŽÿç■Ľăÿžă■čăĀ  
èŁŻçġ■ĈĽ'žăŏĽçŽĎæŨžăŖŦăržăžŖŏăÿĀăžŽçĽ'žăŏŽçŽĎăĭjŽēĉnéĜ■ăđ'■ēŕĈŦĬçŽĎăĜ;æŦŕăĭĽăĬĽ'æŦŦ  
ăÿ;ăĭŇăĬēēŏŭĭijŇăĉĆăđĬă;ăăĈŖăžŖŏăŖŨăĖŏă■ŨăĽŨăŨĜăžŭăÿ■āzēæŦŕă■ŏăĭŨçŽĎæŨžăĭŔēŕžăŔŨăŦŕă  
read() æĽŨ recv() ĭijŇăžŭăĬĭăŔŖŏēĭçŦ'ĝēŭŖăÿĀăÿĽăŨĜăžŭçžŖăŕĭ;ætŦŇērŦăĬēăĖŖăŏŽăŸŕăŖēçžĽă■čă  
iter()   ēŕĈŦĬŕŖăŖăžēăŕĖăÿđ'ēĀĖçžŖăŖĽēŭăĬēăŖĖăĀĈ   ăĖŭăÿ■   lambda  
ăĜ;æŦŕăŖĈăŦŦŕăŸŕăÿžăžĖăĽŽăžžăÿĀăÿĽăŨăăŖĈçŽĎ callable āržēŖăĭijŇăžŭăÿž recv  
æĽŨ read() æŨžăŖŦăŖŖă;ŽăžĖ sizeăŖĈăŦŕăĀĈ

## 7 ċñňăžŤčňăiijŽæŮĠăžŭăÿŎŎ

æL'ĂæIJL'çl'NăžŔéÇ;èçAăd'ĐçŔEè;ŖSăĚăŠŇè;ŖSăĠžăĂĆ  
èĚŽăÿĂçňăăŕEăŭŭçŽŮăd'ĐçŔEăÿ■ăŔŇçşzădŇçŽĐæŮĠăžŭiijŇăŇĚæŇňæŮĠæIJňăŖŇăžŇèĚŽăŬăŮĠăžŭi  
ăŕžæŮĠăžŭăŔ■ăŖŇçŽăă;ŤçŽĐæŖ■ă;IJăžşăiijŽæŭL'ăŔăŬăŕăĂĆ

Contents:

### 7.1 5.1 èŕžăEŽæŮĠæIJňæŤŕæ■ŏ

#### éŮŏécŸ

ăĵăéIJĂèçAèŕžăEŽăŔĐçğ■ăÿ■ăŔŇçijŮçăAçŽĐæŮĠæIJňæŤŕæ■ŏiijŇæŕŤăçCĂŖŭiijŇUTF-  
8ăŬŮUTF-16çijŮçăAç■L'ăĂĆ

#### èğčăEşşæŮžæăŬ

ăĵ;ĚŤŬăÿæIJL' rt æŬăăijŔçŽĐ open () âĠ;æŤŕèŕžăŔŮæŮĠæIJňæŮĠăžŭăĂĆăçCăÿŇæL'Ăçd'žiiž

```
# Read the entire file as a single string
with open('somefile.txt', 'rt') as f:
    data = f.read()

# Iterate over the lines of the file
with open('somefile.txt', 'rt') as f:
    for line in f:
        # process line
    ...
```

çşzăiijçŽĐiijŇăÿžăžEăEŽăĚăÿĂăÿŬæŮĠæIJňæŮĠăžŭiijŇăĵ;ĚŬăÿæIJL' wt  
ăŬăăijŔçŽĐ open () âĠ;æŤŕiijŇăçCădIJăžŇăL'■æŮĠăžŭăEĚăŏžă■ŸăIJăŬăŬăŬăÿĚăŽd'ăžŭèçEçŽŮăŎŬăăĂĆ

```
# Write chunks of text data
with open('somefile.txt', 'wt') as f:
    f.write(text1)
    f.write(text2)
    ...

# Redirected print statement
with open('somefile.txt', 'wt') as f:
    print(line1, file=f)
    print(line2, file=f)
    ...
```

ăçCădIJăŸŕăIJăŭăă■ŸăIJăŮĠăžŭăÿ■ăŭăăŬăăEĚăŏžiiijŇăĵ;ĚŬăŬăăijŔăÿž at çŽĐ  
open () âĠ;æŤŕăĂĆ

æŮĠăzŭçŽĎërŷăĚŹæŠ■ă;IJézŸëôđ'ă;ŁçŦłçşzçzşçijŮčăAġijŃăŔŕăzëéĂŽëŁĠërĈçŦł  
sys.getdefaultencoding() æłĕă;ŮăĹŕăĂĈăĲłăđ'ġăđ'ŽæŦŕăĲŷăŹłăŷŁéłĕćĈ;æŸŕutf-  
8çijŮčăAăĂĈăĕĈăđĲă;ăăŭşçzŔçŞëéAŞă;ăëĕAĕŕŷăĚŹçŽĎăŮĠăĲăŸŕăĔŷăŷŮçijŮčăAæŮŷăĲŕĲijŃ  
éĈăŷăĹăŔŕăzëéĂŽëŁĠăĲăĕĂŞăŷĂăŷłăŔŕéĂĹçŽĎ encoding  
ăŔĈăŦŕçžŽopen()ăĠăŦŕăĂĈăĕĈăŷŃăĹĂĈđ'žĲijŽ

```
with open('somefile.txt', 'rt', encoding='latin-1') as f:  
    ...
```

PythonæŦŕăŃĂéłđăŷŷăđ'ŽçŽĎăŮĠăĲăŷçijŮčăAăĂĈăĠăăŷłăŷŷëġAçŽĎçijŮčăAæŸŕascii,  
latin-1, utf-8ăŦŦutf-16ăĂĈăĲłăŷăŷŦçŦłçłŃăŷŔăŷ■éĂŽăŷŷëĈ;ă;ŁçŦłçŽĎăŸŕUTF-8ăĂĈ  
asciiŕŕŷăŷŦăŹŦăŹŮŮ+0000ăĹŦŮ+007ŦëŃĈăŹŦăĔĔçŽĎ7ă;■ăŮçņăăĂĈ latin-1æŸŕă■ŮëĹĈ0-  
255ăĹŦŮ+0000ëĠŮ+00ŦŦëŃĈăŹŦăĔĔUnicodeă■ŮçņçŽĎçŽŦæŮŕăŷăăŕĎăĂĈ  
ă;ŦŕŷăŔŮăŷĂăŷłăĲçşççijŮčăAçŽĎăŮĠăĲăŮŷă;ŁçŦłlatin-  
1çijŮčăAæŷŷëĔĲăŷ■ăĲŷăžġçŦŦşġçăĂĕŦŦŕŕăĂĈ ä;ŁçŦłlatin-  
1çijŮčăAĕŕŷăŔŮăŷĂăŷłăŮĠăzŭçŽĎăŮŷăĂŽăžşëôŷăŷ■ĕ;ăžġçŦŦşăôŃăĔĹă■ĈçăôçŽĎăŮĠăĲăŷççăĂæŦŕ  
ă;ĔæŸŕăôĈăžşëĈ;ăžŮăŷ■ăĕŔŕăŔŮăĠžëŷăđ'şăđ'ŽçŽĎăĲłçŦłæŦŕă■ôăĂĈăŔŃăŮŷĲijŃăĕĈăđĲă;ăăžŃăŔŕă

## ëőłéőž

ĕŕŷăĚŹæŮĠăĲăŮĠăzŭăŷĂĕĹŃăĕëôŷæŸŕăŕŦĕ;ĈçôĂă■ŦçŽĎăĂĈă;ĔæŸŕăžşăĠăçĈăŷŦŕéĲĂĕĕAæş  
éĕŮăĔĲijŃăĲłăŮă■ŔçłŃăŷŔăŷ■çŽĎwithĕŕ■ăŔĕçžŽĕćŃă;ŁçŦłăĹŕçŽĎăŮĠăzŭăĹŷăžŷăĔăŷĂăŷłăŷŁăŷŃă  
ă;Ĕ with æŮġăĹŷăĲçşşæĲşæŮŷĲijŃăŮĠăzŭăŷŦĕĠăĹăĔşĕŮ■ăĂĈă;ăăžşăŔŕăzëăŷ■ă;ŁçŦł  
with ĕŕ■ăŔĕĲijŃă;ĔæŸŕĕŦŹæŮŷăĂŽă;ăăŕŷăĔĔăžëôŕăĲŮăĹŃăĹăĔşĕŮ■æŮĠăzŭĲijŽ

```
f = open('somefile.txt', 'rt')  
data = f.read()  
f.close()
```

ăŔĕăđ'ŮăŷĂăŷłĕŮŕéĈŷæŸŕăĔşăžŮă■ĕăŃçņççŽĎĕŕĔăĹŃĕŮŕéĈŸĲijŃăĲłUnixăŦŦWindowsăŷ■æŸŕăŷŮ  
\\năŦŦ \\r\\năĂĈăĕžŸëôđ'æĈĔăĔăŷŮŲĲijŃăPythonăĲŷăžĕçžşăŷĂăĹăăĲŕăđ'ĎçŔĔæ■ĕăŃçņăăĂĈ  
ĕŁŹçġ■ăĹăăĲŕăŷŮŲĲijŃăĲłĕŕŷăŔŮăŮĠăĲăŷçijŮčăŮŷăĂŽŲĲijŃăPythonăŔŕăzëĕŕĔăĹŃăĹ'ĂăĲłçŽĎăŽŕéĂŽæ■  
\\nă■ŮçņăăĂĈăçşăĲijŷçŽĎĲijŃăĲłĕ;şăĠăŷæŮŷăĲŷăŔĔæ■ĕăŃçņĕ \\n  
ĕĲŃă■ăŷŷçşçzşçéžŸëôđ'çŽĎă■ĕăŃçņăăĂĈăĕĈăđĲă;ăăŷ■ăŷŮăĲŷĕŁŹçġ■éžŸëôđ'çŽĎăđ'ĎçŔĔæŮŷăĲŕăŮ  
open()ăĠăŦŕăĲijăăĔĔăŔĈăŦŕ newline=' 'ĲijŃăŕŷăĈŔăŷŃĕłĕĕŁæăŷĲijŽ

```
# Read with disabled newline translation  
with open('somefile.txt', 'rt', newline='') as f:  
    ...
```

ăŷŷăžĔĕŕŦ'æŸŮăŷđ'ĕĂĔăžŃĕŮŦ'çŽĎăŷôăĲĲijŃăŷŮĕłĕăĹŮăĲłUnixăĲŷăŹłăŷŁéłĕćŕŷăŔŮăŷĂăŷłWind  
hello world!\\r\\nĲijŽ

```
>>> # Newline translation enabled (the default)  
>>> f = open('hello.txt', 'rt')  
>>> f.read()  
'hello world!\\n'  
  
>>> # Newline translation disabled
```



```
>>> g = open('hello.txt', 'rt', newline='')
>>> g.read()
'hello world!\r\n'
>>>
```

æIJĀāRŌäyÄäyléUőécYārsæYřæŮĜæIJñæŮĜäzúäy■āRřèĈ;āĜžçŎřçŽĎcijŮčāAéŤŽèřřāĀĆ  
ä;Eä;äerzāRŮāLŮēAēĀEZāĒēäyÄäylæŮĜæIJñæŮĜäzúæŮüijNä;āāRřèĈ;äijZéAĜāLřäyÄäylçijŮčāAæLŮē

```
>>> f = open('sample.txt', 'rt', encoding='ascii')
>>> f.read()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/local/lib/python3.3/encodings/ascii.py", line 26, in _
    ↪ decode
    return codecs.ascii_decode(input, self.errors)[0]
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc3 in position
12: ordinal not in range(128)
>>>
```

āēĈæđIJāĜžçŎřèfZäyléŤŽèřřijNēĀŽäyÿēāfçd'zä;äerzāRŮāŮĜæIJñæŮüæNĜāōŽçŽĎcijŮčāAäy■æ■çç  
ä;āæIJĀāē;äzŤçZĒéYĒèřzèřt'æYŎāzúçāōēōd'ä;āçŽĎæŮĜäzúçijŮčāAæYřæ■ççāōçŽĎ(æřŤāēĈā;fçŤŤUTF-  
8ēĀNäy■æYřLatin-1çijŮčāAæLŮāĒūāzŮ)āĀĆ āēĈæđIJçijŮčāAéŤŽèřřèfYæYřā■YāIJçŽĎèřřijNä;āāRřäzē  
open() āĜ;æŤřäijäēĀšäyÄäylāRřéĀLçŽĎ errors āRĈæŤřælēād'ĎçŘĒēfZäzZéŤŽèřřāĀĆ  
äyNēlĀēYřäyÄäzZād'ĎçŘĒäyÿēgAéŤŽèřřçŽĎæŮzæšŤijŽ

```
>>> # Replace bad chars with Unicode U+fffd replacement char
>>> f = open('sample.txt', 'rt', encoding='ascii', errors='replace')
>>> f.read()
'Spicy Jalape?o!'
>>> # Ignore bad chars entirely
>>> g = open('sample.txt', 'rt', encoding='ascii', errors='ignore')
>>> g.read()
'Spicy Jalapeo!'
>>>
```

āēĈæđIJā;āçzRāyÿä;fçŤŤ errors āRĈæŤřælēād'ĎçŘĒçijŮčāAéŤŽèřřijNāRřèĈ;äijZèōl'ä;āçŽĎçŤšæt  
ārzāzŎæŮĜæIJñād'ĎçŘĒçŽĎēçŮēçAāŎšāLZæYřçāōāfĪä;āæĀzæYřä;fçŤŤçŽĎæYřæ■ççāōçijŮčāAāĀĆā;šæ  
8)āĀĆ

## 7.2 5.2 æL'Sā■rèçŞāĜžèĜşæŮĜäzúäy■

### éUőécY

ä;āæĈşārE print() āĜ;æŤřçŽĎèçŞāĜžèĜ■āōZāRŠāLřäyÄäylæŮĜäzúäy■āŎzāĀĆ



èġčǎẸșæŮźæǻŁ

!J!print() åĜ;æTṛäy■æNĜăoŽ file åĖšetŏa■ŮaRĆæTṛijNăČRäyNe!cèfZæăuiijŽ

```
with open('d:/work/test.txt', 'wt') as f:
    print('Hello World!', file=f)
```

èóìèőž

ǎĖšǎžŎëǝŞǎĜzéĜ■ǎǎŏZǎŘSǎĹrǎŮĜǎžŭǎy■ǎřsǝfZǎžZǎžEǎǎĆǎǝEǎŸrǎIJLǎyǎĈĆžǝǎǎǝǝlǎĎRǝŽǎĎǎřǎǎĈĆǎĎIJǎŮĜǎžŭǎŸrǎžNǝfZǎĹŭǎǝlǎǎijRǝŽǎĎǎřǎijZǎĜzéTǎZǎǎĆ

### 7.3 5.3 ä;ŁçŦíaĖŰäzŰáŁĖēŽŦçņæŁŰēàŦçžŁæ■ćçņæŁ'Šà■

éŮőécŸ

ä:äăČsä:řčŤlprint()ăĜ:ăŤrēčŚăĜzăŤră■ōijNă:EăYŕăČsăŤzărŸézYēōđ'čŽDăLĚčŽŤčņăLŮē.

èġčǎẸșæŮźæąŁ

```

    ĀŖāzēä;ŁćŦĭāĬĬ      print()      āĠ;æŦŕäy■ä;ŁćŦĭ      sep      āŠŅ      end
    āĖšēŦōā■ŪāŖĀēŦŕĭĭjŅāzēä;āāĈšēēAćŽDæŨzāĭŖē;ŠāĠzāĀĆæŕŦāēĈĭĭjŽ

```

```
>>> print('ACME', 50, 91.5)
ACME 50 91.5
>>> print('ACME', 50, 91.5, sep=', ')
ACME,50,91.5
>>> print('ACME', 50, 91.5, sep=', ', end='!!\n')
ACME,50,91.5!!
>>>
```

äꞥçŦĭ end āŦĈæŦřāzšāŦŦřāzēāĭJlē;ŠāGžäy■çēAæ■cæ■cēāNāĈCærŦāēCiiJž

```
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
>>> for i in range(5):
...     print(i, end=' ')
...
0 1 2 3 4 >>>
```

èóìèőž

```

    a;S;ä;äeČš;ä;fçTlélçf'zæäijäLēēZTçñælēēçŠaGžæTṛæ■ōçŽDæUūāĀZiijNçizZ
print()    āG;æTṛāijāēĀŠāyĀäyĭ    sep    āRĆæTṛæYṛæIJĀçōĀā■TçŽDæŪzæāLāĀĆ
æIJL'æUūāĀZā;āaijŽçIJNāLṛāyĀāzZçlNāžRāSŸaijŽā;fçTl                                str.join()
æIēāōNæLṚāRŃæāūçŽDāžNæČĒāĀĆæfTāçĆiijŽ

```

```
>>> print(','.join(('ACME', '50', '91.5')))
ACME,50,91.5
>>>
```

```
str.join()  çŽĎeŮóécŸăİJlăžŎăőČăzĚăžĚěĂĈĉŤlăžŎă■ŮčņęăŷšăĂĈĉfZăĎŔăŜșĭĂă;ăéĂŽăŷŷéĬ
```

```
>>> row = ('ACME', 50, 91.5)
>>> print(','.join(row))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: sequence item 1: expected str instance, int found
>>> print(','.join(str(x) for x in row))
ACME,50,91.5
>>>
```

ä;äãŞçĐũăŔăzëäy■čŤléĈčázĽëzzčĈëijŇăŔŤĽĬăĕĖĀăĈŔăyŇéŭĕĕĹæăăĖĹziĭŽ

```
>>> print(*row, sep=',')
ACME, 50, 91.5
>>>
```

## 7.4 5.4 èrzãĖŽã■ŮèŁĆæŦřæ■ó

éŮőécŸ

ä:äăĈșërzãEŻăžŇefŽăĹúæŮĜăžũijŇăřŤăeĆăŻ;çL'ĜiijŇăċřěšşæŮĜăžũç■L'ç■L'ăĂĈ

èġčǎẸșæŮźæąŁ

ä;fcTlælaaijRävž rb æLŪ wb çŽĐ open ( ) aĠ;æTɾæiëērzaRŪæLŪaEŽaĖēazNefZaLúaTɾæ■oāĀCæTl

```
# Read the entire file as a single byte string
with open('somefile.bin', 'rb') as f:
    data = f.read()

# Write binary data to a file
with open('somefile.bin', 'wb') as f:
    f.write(b'Hello World')
```

āIjIērzāRŪāžNēŁZāŁūāTŗā■ōāŪūiijNēIJĀēēAāNĠāYŌčŽDāYřāL'ĀāIJL'ēŁTāŽdčŽDāTŗā■ōēČ;āY  
 čsžāijijčŽDiiJNāIJĀēZāĒēčŽDāŪūāĀŽiiJNāŁĒēāzāfIērAāRČāTŗāYřāzēā■ŪēŁČā;čāijRāřzād'ŪāŽt'ēIJsāTŗā

## èóìèőž

ǎIJlérzǎRŰázÑeƒZǎLúæTŗæ■óçŽĐæŰúǎĂZrijŃǎ■ŰèŁĆǎ■ŰçņęäÿśǎŠŃæŰĜæIJǎǎ■ŰçņęäÿşçŽĐér■ǎžŁ  
çL'zǎLnéIJǎèçAæşlæĐRçŽĐæŸřijŃçť cáijTǎŠÑeƒ■ǎžčǎLǎ;IJèƒTǎŽđçŽĐæŸřǎ■ŰèŁĆçŽĐǎĀijèǎŃäÿ■æŸř

```
>>> # Text string
>>> t = 'Hello World'
>>> t[0]
'H'
>>> for c in t:
...     print(c)
...
H
e
l
l
o

...
>>> # Byte string
>>> b = b'Hello World'
>>> b[0]
72
>>> for c in b:
...     print(c)
...
72
101
108
108
111

...
>>>
```

ǎeĆæđIJǎ;ǎæČşǎžŎǎžÑeƒZǎLúæÍǎǎijRçŽĐæŰĜǎžúäÿ■érzǎRŰæLŰǎEǎZǎĖæŰĜæIJǎæTŗæ■órijŃǎǎĖéǎ

```
with open('somefile.bin', 'rb') as f:
    data = f.read(16)
    text = data.decode('utf-8')

with open('somefile.bin', 'wb') as f:
    text = 'Hello World'
    f.write(text.encode('utf-8'))
```

ǎžÑeƒZǎLúI/OèƒŸæIJL'äÿĂäÿłéšIJäÿžǎžžçşççŽĐçL'zæĂĝǎřsæŸřæTŗçžĐǎŠŃCçzŞæđĐǎ;ŞçşzǎđŃèÇ;ç

```
import array
nums = array.array('i', [1, 2, 3, 4])
with open('data.bin', 'wb') as f:
    f.write(nums)
```

èƒZǎÿłéĂĆçTłǎžŎǎžǎ;TǎőđçŎřǎžEęėćńçĝřǎžŃäÿžǎĀİçijŞǎEşæŎèǎRçǎĀİçŽĐǎřzèşǎijŃeƒZçĝ■ǎřzèşǎij

æžŇëƒŽǎĹŭæŦŕæ■ōçŽĎæŽǎĚĚǎŕsæŸŕëƒŽçsžæŠ■äĵIJǎžŇäŸǺǎĀĆ

ǎĴĹǎđ'ŽǎŕžèšǎèƒŸǎĚAèöŸéǺŽèƒĞǎĵƒçŦĹæŨĞǎžŭǎŕžèšǎçŽĎ readinto()  
æŨžæšŦçŽŦ'æŐĚŕžǎŦŨǎžŇëƒŽǎĹŭæŦŕæ■ōǎĴŕǎĚŨǎžŦǎšĆçŽĎǎĚĚǎŸäŸ■ǎŐžǎĀĆæŕŦǎèĆĵijŽ

```
>>> import array
>>> a = array.array('i', [0, 0, 0, 0, 0, 0, 0, 0])
>>> with open('data.bin', 'rb') as f:
...     f.readinto(a)
...
16
>>> a
array('i', [1, 2, 3, 4, 0, 0, 0, 0])
>>>
```

ǎĵæŸŕǎĵƒçŦĹëƒŽçg■æĴǺæIJŕçŽĎæŨŭǎǺŽéIJǎèèAæǎĵǎđ'ŨǎŕŦǎƒĆĵijŇǎŽǎäŸžǎōĆéǺŽǎŸŸǎĚŨæIJĴǎž  
ǎŦŕǎžèæšèçIJŇ5.9ǎŕŦèĴCǎŸ■ǎŦèǎđ'ŨǎŸǺǎŸŕèžǎŦŨǎžŇëƒŽǎĹŭæŦŕæ■ōǎĴŕǎŦŕǎƒōæŦžçijŠǎĚsǎŇžçŽĎǎĴŇǎ

## 7.5 5.5 æŨĞǎžŭäŸ■ǎŸǎĴĴæĴ■èĆĵǎĚŽǎĚĚ

### éŨŏécŸ

ǎĵǎæĆšǎĴŕǎŸǺǎŸŕæŨĞǎžŭäŸ■ǎĚŽǎĚĚæŦŕæ■ōĵijŇǎĵæŸŕǎĴ■æŦŦǎƒĚéǎžæŸŕëƒŽǎŸŕæŨĞǎžŭǎĴĴæŨĞ  
ǎžšǎŕsæŸŕǎŸ■ǎĚAèöŸèèĚçŽŨǎŸsǎŸǎĴĴçŽĎæŨĞǎžŭǎĚĚǎōžǎĀĆ

### èğçǎĚşæŨžæǎĴ

ǎŦŕǎžèǎĴĴ open() ǎĞĵæŦŕǎŸ■ǎĵƒçŦĴ x æĴǎĵŦŕǎĴèǎžçæŽƒ w  
æĴǎĵŦŕçŽĎæŨžæšŦǎĴèèğçǎĚşèƒŽǎŸŕéŨŏécŸǎĀĆæŕŦǎèĆĵijŽ

```
>>> with open('somefile', 'wt') as f:
...     f.write('Hello\n')
...
>>> with open('somefile', 'xt') as f:
...     f.write('Hello\n')
...
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
FileExistsError: [Errno 17] File exists: 'somefile'
>>>
```

ǎèĆǎđIJæŨĞǎžŭæŸŕǎžŇëƒŽǎĴŭçŽĎĵijŇǎĵƒçŦĴ xb æĴèǎžçæŽƒ xt

### èŏŕéŏž

èƒŽǎŸǺǎŕŦèĴCǎĵŦçđ'žǎžĚǎĴĴǎĚŽæŨĞǎžŭæŨŭéǺŽǎŸŸǎĵŦžéAĞǎĴŕçŽĎǎŸǺǎŸŕéŨŏécŸçŽĎǎŕŇçĴŐğğ  
ǎŸǺǎŸŕæŽƒǎžçæŨžæǎĴæŸŕǎĚĴǎŦŦèŦŦèƒŽǎŸŕæŨĞǎžŭæŸŕǎŦèǎŸǎŸǎĴĴĵijŇǎĴŕǎŸŦéŕèèƒŽæǎŸĵijŽ

```
>>> import os
>>> if not os.path.exists('somefile'):
...     with open('somefile', 'wt') as f:
...         f.write('Hello\n')
... else:
...     print('File already exists!')
...
File already exists!
>>>
```

æŸçèĀŃæŸŞèğĀiijŃă;fcŦĪxæŨĜăzŭăĭăiJRæZt'ăŁăçŏĀăŦăĀCèèAæşĭăĎŔçŽĎæŸřxăĭăiJRæŸřăŸŦ  
 open()ăĜ;æŦřçL'žăIJL'çŽĎæL'řăŝŦăĀCăIJĪPythonçŽĎæŨĝçL'LăIJŃăLŨèĀĖæŸřPythonăŏđçŎřçŽĎăŹŦ

## 7.6 5.6 ăŦŨçņęäŸçŽĎĪ/OæŞăĪĪ

### éŨŏécŸ

ăĵăæČşă;fcŦĪăŞăĪĪçşzæŨĜăzŭăřzèşççŽĎçĪŃăžŔăĭăæŞăĪĪæŨĜăĪŃăLŨăžŃèŁZăĹŭăŦŨçņęäŸşăĀ

### èğçăEşæŨzæăĹ

ă;fcŦĪio.StringIO()ăŝŃio.BytesIO()çşzăĭăăĹZăžžçşzæŨĜăzŭăřzèşçæŞăĪĪăŦŨçņęäŸşăĀŦ

```
>>> s = io.StringIO()
>>> s.write('Hello World\n')
12
>>> print('This is a test', file=s)
15
>>> # Get all of the data written so far
>>> s.getvalue()
'Hello World\nThis is a test\n'
>>>

>>> # Wrap a file interface around an existing string
>>> s = io.StringIO('Hello\nWorld\n')
>>> s.read(4)
'Hell'
>>> s.read()
'o\nWorld\n'
>>>
```

io.StringIOăŔĭèČ;çŦĪăžŎăŨĜăĪŃăĀCăèČăđĪăĵăèèAæŞăĪĪăžŃèŁZăĹŭăŦŕăăŏiijŃèèAă;fcŦĪ  
 io.BytesIOçşzăĭăăžçăŽŁăĀCăŕŦăèČiijŽ

```
>>> s = io.BytesIO()
>>> s.write(b'binary data')
>>> s.getvalue()
```

```
b'binary data'
>>>
```

## èõléõž

å¡Šä¡æČšæÍæNšäyÄäyÍæŽóéÄŽčŽDæŮGäzŭčŽDæŮŭäÄŽ StringIO åŠŇ  
BytesIO çšzæÝřäĹæIJL'çTÍčŽDäÄČ æŕTæČřijŇNäIJÍäTäEČæŧNèŕTäy■īijŇä¡ääŕřäzëä¡£çTÍ  
StringIO æÍæäLŽäzžäyÄäyÍæNĚäŕNæŧNèŕTæTŕæ■óčŽDçšzæŮGäzŭäŕfžèšajijŇ  
è£ŽäyÍäŕfžèšäqäŕřäzëècñajčžZæšŕäyÍäŕČæTŕäyžæŽóéÄŽæŮGäzŭäŕfžèšäçŽDäG¡æTŕäÄČ  
éIJÄèèAæšÍæDŕČŽDæYřijŇ StringIO åŠŇ BytesIO  
åõđäĹNäzŭæšqæIJL æ■čçäóčŽDæTŕ æTŕçšzäđNčŽDæŮGäzŭæŕŕè£řçñæÄČ  
äZäæ■d'īijŇNäõČäznäy■èČ¡äIJléČčäžZéIJÄèèAä¡£çTÍIJšåõđčŽDçšzçzšçžgæŮGäzŭäèČæŮGäzŭīijŇçõæéAš

## 7.7 5.7 èŕzäEŽäŮŇçijl'æŮGäzŭ

### éŮóécŸ

ä¡æČšèŕzäEŽäyÄäyÍgzipæLŮbz2æāijāijŕčŽDäŮŇçijl'æŮGäzŭäÄČ

### èğčäEşæŮzæaĹ

gzip åŠŇ bz2 æÍäaÍŮäŕřäzëäĹLäõzæYšçŽDäđ'ĐçŕEè£ŽäzŽæŮGäzŭäÄČ  
äyđ'äyÍæÍäaÍŮèČ¡äyž open() åG¡æTŕæŕŕäĹZäzEäŕæäđ'ŮčŽDäõđčŎŕæÍèèğčäEşè£ŽäyÍéŮóécŸäÄČ  
æŕTæČřijŇäyžäzEäžæŮGæIJŇä¡čāijŕèŕzäŕŮäŮŇçijl'æŮGäzŭīijŇNäŕřäzëè£ŽæäŭäÄŽīijŽ

```
# gzip compression
import gzip
with gzip.open('somefile.gz', 'rt') as f:
    text = f.read()

# bz2 compression
import bz2
with bz2.open('somefile.bz2', 'rt') as f:
    text = f.read()
```

çšzāijijčŽDīijŇäyžäzEäEŽäĚèäŮŇçijl'æTŕæ■ōīijŇNäŕřäzëè£ŽæäŭäÄŽīijŽ

```
# gzip compression
import gzip
with gzip.open('somefile.gz', 'wt') as f:
    f.write(text)

# bz2 compression
import bz2
```

```
with bz2.open('somefile.bz2', 'wt') as f:
    f.write(text)
```

æĈäÿŁiijŇæL'ÄæIJL'čŽĐI/OæŠ■ä;IJéČ;ä;ŁçŤlæŮĜæIJŇæłäiijRāzŭæL'ĝèaŇUnicodečŽĐčijŮčăA/èĝčç  
çszăijijčŽĐiijŇæĈăđIJä;ăæĈşæŠ■ä;IJăžŇèŁZăLŭæŤræ■ōiijŇă;ŁçŤl rb æLŮèĂĚ wb  
æŮĜăzŭæłäiijRă■şăRřăĂĈ

## èõlèõž

ăđ'ĝéČlăLĚæĈĚăĚŤăÿŇérzăĚŽăŮŇčijl' æŤræ■óéČ;æŸrăŁçőĂă■ŤčŽĐăĂĈă;ĚæŸrèçAæşlæĎRčŽĐæŸ  
æĈăđIJä;ăäÿ■æŇĜăőZăłäiijRiijŇéĈčăžLéžŸèđ'čŽĐăřsæŸřăžŇèŁZăLŭæłäiijRiijŇæĈăđIJèŁZăŮŭăĂŽç  
gzip.open() äŇŇ bz2.open() æŮěăRŮèŭşăĚĚç;őçŽĐ open()  
ăĜ;æŤrăÿĂæăŭçŽĐăRČæŤriijŇ äŇĚæŇŇ encodingiijŇerrorsiijŇnewline  
ç■L'ç■L'ăĂĈ

ă;ŞăĚŽăĚĚăŮŇčijl' æŤræ■óæŮŭiijŇăRřăžăä;ŁçŤl compresslevel  
èŁZăÿlăRřăĂL'čŽĐăĚşéŤőă■ŮăRČæŤræłæŇĜăőZăÿĂăÿlăŮŇčijl' çžĝăLŇăĂĈæŤăçCiiž

```
with gzip.open('somefile.gz', 'wt', compresslevel=5) as f:
    f.write(text)
```

ézŸèđđ'čŽĐç■L'çžĝæŸr9iijŇăžşæŸræIJăénŸčŽĐăŮŇčijl' ç■L'çžĝăĂĈç■L'çžĝèŭLă;ŮăĂĝèČ;èŭLăè;ij  
æIJăăRŮăÿĂçČziiŇ gzip.open() äŇŇ bz2.open()  
èŁŸæIJL'ăÿĂăÿlăŁăRşĈéçŇşééAşçŽĐçL'žăĂĝiijŇăőČăžŇăRřăžăä;IJçŤlăIJlăÿĂăÿlăŭşă■ŸăIJlăžŭăžăžŇèŁ

```
import gzip
f = open('somefile.gz', 'rb')
with gzip.open(f, 'rt') as g:
    text = g.read()
```

èŁZăăŭăřşăĚĂèőÿ gzip äŇŇ bz2 æłäăłŮăRřăžăăŭëä;IJăIJlèőÿăđ'ŽçşszæŮĜăzŭăřžèşăÿŁiijŇăŕŤăçĂă

## 7.8 5.8 ăŽžăőŽăđ'ĝăřRèőřă;ŤčŽĐæŮĜăžŭèŁ■ăžč

### éŮóécŸ

ă;ăæĈşăIJlăÿĂăÿlăŽžăőŽéŤŁăžçèđřă;ŤæLŮèĂĚæŤræ■óăłŮčŽĐéŽĚăRĬăÿŁèŁ■ăžčriijŇèĂŇăÿ■æŸřăIJ

### èĝčăĚşæŮžæăĬ

éĂŽèŁĜăÿŇéłçèŁZăÿlăřRăŁăăŭĝă;ŁçŤl iter äŇŇ functools.partial()  
ăĜ;æŤriijŽ

```
from functools import partial

RECORD_SIZE = 32
```

```
with open('somefile.data', 'rb') as f:
    records = iter(partial(f.read, RECORD_SIZE), b'')
    for r in records:
        ...
```

èĚZäylä;Nā■Räy■çŽĎ records áržēsæYřäYÄäyläRřē■āzčáržēsaiijNāōČaijŽäy■æŮ■çŽĎäžgčTšāŽž  
èĚAæslæDŘčŽĎæYřæČædIJæÄžèōřā;Tāđ' gārRäy■æYřaiŮāđ' gārRčŽĎæTř' æTřāA■çŽĎēřliijNæIJĀāRŌäy/

## èóìèőž

iter() āG;æTřæIJL'äyÄäylēšIJäyžāžžçšĚčŽĎçL'žæĀgārśæYřiiijNāēČædIJā;āçžŽāōČaijāēĀŠäyÄäylā  
èĚZäylēē■āzčāŽlaijŽäyĀçŽt'ērČçTlaijāāĚĚçŽĎāRřērČçTlāržēsāçŽt' āLřāōČēfTāžđæāGēōřāĀijäyžæ■ciijNēf

āIJlä;Nā■Räy■iiijN functools.partial çTlālēāLZāžžäyÄäylāēRāēāēēēēēČçTlāŮüāžŌæŮGāžüā  
æāGēōřāĀij b' ' ' āřśæYřā;ŠāLřē;æŮGāžüçžŠār;æŮüçŽĎēfTāžđāĀijāĀČ

æIJĀāRŌāE■æRŘäyĀçČžiiijNäyLēīčçŽĎä;Nā■Räy■çŽĎæŮGāžüāŮüāžēāžNēfZāLūāēāaijRæL'SaijĀç  
āēČædIJæYřēržāRŮāŽžāōŽāđ' gārRčŽĎēōřā;TiiijNēfZēĀŽäyYæYřæIJæZōēA■çŽĎæČĚāEřāĀČ  
èĀNāržāžŌæŮGæIJnæŮGāžüiiijNäyĀēāNäyĀēāNçŽĎēržāRŮ(ēžYēōđ' çŽĎēf■āzčēāNäyž)æŽt' æZōēA■çŽā

## 7.9 5.9 èřžāRŮāžNēfZāLūæTřæ■ōāLřāRřāRŸçijŠāEšāNžäy■

### éŮóécŸ

ā;āæČšçŽt' æŌēēřžāRŮāžNēfZāLūæTřæ■ōāLřäyÄäylāRřāRŸçijŠāEšāNžäy■iiijNēĀNäy■ēIJĀēĚAāAžžā  
æLŮēĀĚā;āæČšāŌšāIJřāfōæTžæTřæ■ōāžüārEāōČāEžāŽđāLřäyÄäylāēŮGāžüāy■āŌžāĀČ

### ègčāEšæŮžæāŁ

äyžāžEēržāRŮæTřæ■ōāLřäyÄäylāRřāRŸæTřçžDäy■iiijNā;fçTlāŮGāžüāržēsāçŽĎ  
readinto() æŮžæšTāĀČæfTāēČiiijŽ

```
import os.path

def read_into_buffer(filename):
    buf = bytearray(os.path.getsize(filename))
    with open(filename, 'rb') as f:
        f.readinto(buf)
    return buf
```

äyNēīcæYřäYÄäylāēijTčđ'žēfZäylāG;æTřā;fçTlāŮžæšTçŽĎä;Nā■RiiijŽ

```
>>> # Write a sample file
>>> with open('sample.bin', 'wb') as f:
...     f.write(b'Hello World')
... 
```



```
>>> buf = read_into_buffer('sample.bin')
>>> buf
bytearray(b'Hello World')
>>> buf[0:5] = b'Hallo'
>>> buf
bytearray(b'Hallo World')
>>> with open('newsample.bin', 'wb') as f:
...     f.write(buf)
...
11
>>>
```

èóíèőž

æŮĜäzũärzèšaçŽĎ readinto() æŮzæšTèĈj;èċñĈTlæIèäyžécĎĎǼĹǻĹEèĚ■ǻEĚǻ■ŸĈŽĎæTŕĈzĎĎǻǻǻǻ  
array æĹǻǻŮǻĹŮ numpy āžšǻĹŽāžžĈŽĎæTŕĈzĎĎǻǻǻ āšNæŽōèǻŽ read()  
æŮzæšTǻy■ǻRŇĈŽĎæŸriijŇ readinto() ǻǻǻǻĚǻǻšǻ■ŸǻĹJĹĈŽĎĈijšǻEšǻNžèǻNǻy■æŸřǻyžæŮřǻržèšǻèĈ  
ǻžǻǻ■d'riijNǻ;ǻǻRřǻžèǻj;ĸĈTlǻōĈǻIèèǻǻǻǻǻd'gèGRĈŽĎĎEĚǻ■ŸǻĹEèĚ■æš■ǻjIǻǻǻǻ  
ǻrTǻèĈriijNǻèĈǻdIǻj;ǻèřžǻRŮǻyǻǻǻyĸTšĈŽyǻRŇǻd'gǻřĈŽĎèōǻǻTĈzĎǻĹRĈŽĎǻžNèĸžǻĹŮǻŮĜäzũǻŮŮri

```
record_size = 32 # Size of each record (adjust value)

buf = bytearray(record_size)
with open('somefile', 'rb') as f:
    while True:
        n = f.readinto(buf)
        if n < record_size:
            break
        # Use the contents of buf
    ...
```

ǎŘęǎđ' ŪæIJL' äŷÄäŷlæIJL' èũččL' zæĀğǎřsæŸř                      memoryview                      iijŃ  
 ǎőČǎŘřǎzéeĀŽēfĠēZŭǎđ' ■ǎŁŭčŽĐæŪžǎijRǎřzǎuǎř■ŸǎIJłčŽĐcijŠǎEšǎŃžæL' ġeǎŃǎŁĠçL' ĠǎS■ǎjIJiijŃçTžZ

```
>>> buf
bytearray(b'Hello World')
>>> m1 = memoryview(buf)
>>> m2 = m1[-5:]
>>> m2
<memory at 0x100681390>
>>> m2[:] = b'WORLD'
>>> buf
bytearray(b'Hello WORLD')
>>>
```

ä;fçTÍ f . readinto ( ) æUúéIJAðeAæşÍæDRçZDæYřiiJNä;ääfÉéazæcĂæşcăoČçZDèfTăZdăĂijriiJNă  
 æCædIJA■UðŁCæTřřRřZŔŔçijŞăEşăNžad' gărRriiJNeqlæYŔæTřæ■oècŋæLlæŮ■æLŮèĂĚècŋçar' aiRăZĚ  
 æIJAăRŔŔiiJNçTZăfCègCărşăŔŮăzŮăG;æTřăZşăŞNælaaiŮăy■ăŞN into

çZÿaĖşçZĎĎaĜjæTř(æřTăęĆ      recv\_into()      iijŃ      pack\_into()      çL')ăĂĆ  
PythonçZĎĎaĜLăđ'ZăĖŮăžŮéĈlăĖăŮşçZřëĈjæTřăŃAçZt'æŎëçZĎI/OăĹŮăTřăăőëőĖŮăŞăjIiijŃëřZă  
ăĖşăžŎëğçăđŘăžŃëřZăĹŮçZŞăđĎăŞŃ      memoryviews  
ăjççTlăŮăşçTçZĎăZt'énŸçžgăĹŃăŔiijŃëřăăŔĈëăĂĈ6.12ăřŔëĹĈăăĂĈ

```
>>> # Verify that changes were made
>>> with open('data', 'rb') as f:
...     print(f.read(11))
...
b'Hello World'
>>>
```

mmap () eŦTāZdçZĐ mmap áržesqāRŊăuăzşăRăřăză;IJăyžăyĂăyľăyŁăyŊăŮĠçőăçŔĚă  
eŦZăŮăăĂZăŧŦăsĆçZĐăŮĠăzăuăiŷZècñeĠăŁăĹăĚşéŮăăĂĆăřŦăeĆiiŷ

```
>>> with memory_map('data') as m:
...     print(len(m))
...     print(m[0:10])
...
10000000
b'Hello World'
>>> m.closed
True
>>>
```

```

    ézYèòd' æČĚăĖtăyŇrijŇ memory_map() āĠ;æTṛæLŠajĀčŽDæŮĠăzŭāŔŇăŮŭăTṛă
    äzzā;TṣŽDăŁôăTăăĖĚăôžēČ;ăijŽăd'■ăĹăăZăăŌšăĭēčŽDæŮĠăzŭăy■ăĀČ
    âĖČădĬĖĬĬăĖĀăŔĭēŕčŽDēôĤĖŮôăĭăijŔijŇăŔăŕăžēčzŽăŔČăTṛ      access    èŧŇăĀijăyž
    mmap.ACCESS_READ āĀČăŕTăēČijŽ

```

```
m = memory_map(filename, mmap.ACCESS_READ)
```

æĆæđIJä;äæČšåIJæIJñåIJřæŁæŤzæŤræ■ōijNä;EæŸřáRLäy■æČšårEæŁæŤzæEŹăZďáŁ  
mmap.ACCESS\_COPY ijŽ

```
m = memory_map(filename, mmap.ACCESS_COPY)
```

èòlèőž

```

äyžāžEēŽRæIJžēōēUōāŨĜāzūčŽDāEĖĀōžīijŅā;£çTĪ                                     mmap
ārEāŨĜāzūāYāārDālŖāEĖā■Yāy■āYŕāyĀāylénYāŤLāŠŅāijYēŽEçŽDāŨzæšŤāĀĆ
ä;ŅāēCīijŅā;āāUāēIJāēL'ŠaijĀāyĀāylæŨĜāzūāzūāL'gēaŅād'gēGRçŽD      seek()   iijŅ
read() iijŅ write() ērČçŤīiijŅāŖŖēIJĀēēAçōĀā■ŤçŽDāYāārDāŨĜāzūāzūā;£çŤīāŖLŖGçL'

```

äyÄēĽñæĭēōšijŃ mm̩ap ( ) æĽÄæŽt' éIJšČŽDāEĲā■ŸçIJŃäyĽāŌžārsæŸräyÄäyĽazŃēfZ  
ä;EæŸriijŃā;āāRfrazēa;ŁçTĽāyÄäyĽāEĲā■ŸēgEāZ;æĭēēgčædRāEūäy■ČŽDæTṛæ■ōāĀCærTāčĆ

```
>>> m = memory_map('data')
>>> # Memoryview of unsigned integers
>>> v = memoryview(m).cast('I')
>>> v[0] = 7
>>> m[0:4]
b'\x07\x00\x00\x00'
>>> m[0:4] = b'\x07\x01\x00\x00'
```

```
>>> v[0]
263
>>>
```

éIJÀèèAäijžèrČčŽDäyÄçCzæYřijNāEĚā■YæYāārDäyÄäylæŮGäzúázúäy■äijŽārijèĜt'æTt'äylæŮGäzúázšārsæYřèrt'ijNæŮGäzúázúæsqæIJL'ècnād'■āLūāLrāEĚā■YçijŠā■YæLŮæTřčzDäy■āĀČçŽyāR■ijNæŠ■ā;ā;Šā;æèðéŮðæŮGäzúçŽDäy■āRñāNžāššæŮüijNèfZāžZāNžāššçŽDāEĚāóžæL■æāžæ■óéIJÀèèAèèñèrZāRèĀNéCčāžZāžŌæšqèèèðéŮðāLřčŽDēCīāLĚèfYæYřçTŽāIJlçčAçŽYäyLāĀČæL'ĀæIJL'èfZāžZèfĜçlNæY

æÇCādIJād'ŽäyPythonèĝcéĜLāZlāEĚā■YæYāārDāRñäyÄäylæŮGäzūijNā;ŮāLřčŽD mmap āřzèsqèČ;ād'šècncTlæIēāIJlèĝcéĜLāZlçŽt'æŌèāžd'æ■cæTřæ■ōāĀČ äžšārsæYřèrt'ijNæL'ĀæIJL'èĝcéĜLāZlèC;èČ;āRñæŮüèrZāEŽæTřæ■ōijNāzūäyTāEüäy■äyÄäylèĝcéĜLāZlā;LæYŌæY;ijNèfZéĜNéIJÀèèAèĀČèZSāRñæ■èçŽDēŮóéYāĀČā;EæYřèfZçĝ■æŮzæşTæIJL'æŮūāZāR

èfZäyÄārRèLCäy■āĜ;æTřār;éGRāEŽā;Ůā;LéĀŽçTlrijNāRñæŮüéĀČçTlāžŌUnixāŠNWindowsāzšāR èèAæslæDRçŽDæYřā;ççTl mmap ( ) āĜ;æTřæŮüäijZāIJlāzTāsCæIJL'äyÄāžZāzšāRřçŽDāūōāijCæĀĝāĀČ āRēād'ŮüijNèfYæIJL'äyÄāžZéĀLēāzāRřāžèçTlæIēāLZāžZāNfāR■çŽDāEĚā■YæYāārDāNžāššāĀČ æÇCādIJā;āāržèfZäylæDšāĒt'èüçrijNçqōāfIā;āāzTçzEçāTèrZāžEPythonæŮĜæaçäy■ èfZæŮzélcçŽDāEĚāóž āĀČ

## 7.11 5.11 æŮGäzúèùrā;DāR■çŽDæŞ■ä;IJ

### éŮóéçY

ä;äéIJÀèèAä;ççTlèùrā;DāR■æIèèŌūāRŮæŮGäzúāR■ijNçZōā;TāR■ijNçZlāržèùrā;Dç■Lç■L'āĀČ

### èĝçAÈşæŮzæqL

ä;ççTl os.path ælāāIŮäy■çŽDāĜ;æTřæIèæŞ■ä;IJèùrā;DāR■āĀČ äyNéIcæYřāyÄäylāžd'āžSāijRā;Nā■RæIèæijTçd'žäyÄāžZāÈşéTōçŽDçL'zæĀĝüijZ

```
>>> import os
>>> path = '/Users/beazley/Data/data.csv'

>>> # Get the last component of the path
>>> os.path.basename(path)
'data.csv'

>>> # Get the directory name
>>> os.path.dirname(path)
'/Users/beazley/Data'

>>> # Join path components together
>>> os.path.join('tmp', 'data', os.path.basename(path))
'tmp/data/data.csv'

>>> # Expand the user's home directory
>>> path = '~/Data/data.csv'
```

```
>>> os.path.expanduser(path)
'/Users/beazley/Data/data.csv'

>>> # Split the file extension
>>> os.path.splitext(path)
('~/.Data/data', '.csv')
>>>
```

## èõléõž

årzäžÕäzzä;TçŽDæŮĠäzũāR■çŽDæŞ■ä;IJiijNä;äéÇ;āžTèrēā;£çTĪ os.path  
æĺāāĪŮiijNèĀNäy■æYřä;£çTĪæāĠāĠĠēā■ŮçñēäyşæŞ■ä;IJæĪædĎĒĀæĠāũşçŽDäzççāAāĀĆ  
çL'zāĹnæYřäyžāEāRřçğzæd'■æĀğèĀĆèŽŚçŽDæŮũāĀZæZt'āžTæÇæ■d'iijN āZāyž os.  
path æĺāāĪŮçŞēēAŞUnixāŠNWindowsçşçzçşāzNéŮt'çŽDāũōāijCāzũāyTèÇ;ād'şāRřēĪāĪJřād'ĎçRĒçşzāijij  
Data/data.csv āŠN Data\data.csv è£ZæāũçŽDæŮĠäzũāR■āĀĆ  
āĒŮāñāiijNä;āçIJŞçŽDäy■āžTèrēæŧèt'zæŮũéŮt'āŌzéĠ■ād'■éĀæ;ōā■RāĀĆéĀŽāyÿæIJĀāē;æYřçZt'æŌēā;t  
èēAæşĪæĎRçŽDæYř os.path è£YæIJL'æZt'ād'ŽçŽDāLşèÇ;āĪJĪè£ŽéĠNāzũæşāæIJL'āĹŮäy;āĠZæĪēā  
āRřāzèæşēēYĒāōYæŮzæŮĠæçæĪēèŌũāRŮæZt'ād'ŽāyŌæŮĠäzũæŧNèrTiiijNçñēāRũéŞ;æŌēç■L'çŽyāĒşçŽ

## 7.12 5.12 æŧNèrTæŮĠäzũæYřāRēā■YāĪĪ

### éŮóécY

ä;āæÇşætNèrTäyĀäyĪæŮĠäzũæĹŮçZōā;TæYřāRēā■YāĪĪāĀĆ

### èğçāEşæŮzæāĹ

ä;£çTĪ os.path æĺāāĪŮæĪæŧNèrTäyĀäyĪæŮĠäzũæĹŮçZōā;TæYřāRēā■YāĪĪāĀĆæŧTæÇiijŽ

```
>>> import os
>>> os.path.exists('/etc/passwd')
True
>>> os.path.exists('/tmp/spam')
False
>>>
```

ä;äè£YèÇ;è£ZäyĀæ■æŧNèrTè£ZäyĪæŮĠäzũæŮũāzĀāzĹçşzādNçŽDāĀĆ  
āĪĪāyNéĪcé£ZāžZætNèrTäy■iijNāēĆædIJætNèrTçŽDæŮĠäzũäy■ā■YāĪĪçŽDæŮũāĀZiijNçzŞædIJéÇ;āijŽæ

```
>>> # Is a regular file
>>> os.path.isfile('/etc/passwd')
True

>>> # Is a directory
>>> os.path.isdir('/etc/passwd')
```

```
False
```

```
>>> # Is a symbolic link
>>> os.path.islink('/usr/local/bin/python3')
True

>>> # Get the file linked to
>>> os.path.realpath('/usr/local/bin/python3')
'/usr/local/bin/python3.3'
>>>
```

æĈædIJă;æĕŸæĈşèŌuăRŪăĖĈæŢræ■ó(ærŢăĕĈæŪĜăzúăd'ğărRæĹŪèĂĖæŸrăĽăŢzæŪèæIJ§)ijŇăz  
os.path æĹăăĹŪæĹèĕğĉăĖşijŹ

```
>>> os.path.getsize('/etc/passwd')
3669
>>> os.path.getmtime('/etc/passwd')
1272478234.0
>>> import time
>>> time.ctime(os.path.getmtime('/etc/passwd'))
'Wed Apr 28 13:10:34 2010'
>>>
```

## èőĹèőŹ

ă;ĕĈŢĹ os.path æĹèĕŸæăŇæŪĜăzúăŢŇærŢæŸrăĹĽăŌĂă■ŢĉŹĎăĂĈ  
ăIJăĖŹæĕŹăzŹæĎŹæIJăŇæŪŷijŇăŖrèĈ;ăŢrăŷĂĖIJĂèĕĂæşĹæĎŖĉŹĎăŖşæŸră;ăĖIJĂèĕĂèĂĈèŹŖæŪĜăzúăĹĈ

```
>>> os.path.getsize('/Users/guido/Desktop/foo.txt')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/local/lib/python3.3/genericpath.py", line 49, in _
↳ getsize
    return os.stat(filename).st_size
PermissionError: [Errno 13] Permission denied: '/Users/guido/
↳ Desktop/foo.txt'
>>>
```

## 7.13 5.13 èŌuăRŪæŪĜăzúăd'zăŷ■ĉŹĎæŪĜăzúăĹŪèăĹ

### éŪŏéĈŸ

ă;ăæĈşèŌuăRŪæŪĜăzúĉşzĉzşăŷ■æşŖăŷĽĉŹăă;ŢăŷŇĉŹĎæĹĂæIJĹæŪĜăzúăĹŪèăĹăĂĈ

èġčǎẸșæŮźæǻŁ

```
ä;fçŦl os.listdir() äĜ|æŦræIëËŦuâRŦæşŦrâyłçZôâ;Ŧäy■çŦDæŦŦĜäzûâLŦèa!iijZ
```

```
import os
names = os.listdir('somedir')
```

[illegible]

```
import os.path

# Get all regular files
names = [name for name in os.listdir('somedir')
         if os.path.isfile(os.path.join('somedir', name))]

# Get all dirs
dirnames = [name for name in os.listdir('somedir')
            if os.path.isdir(os.path.join('somedir', name))]
```

a■ŮçņēäÿšçŽĎ                      startswith()                      āŠŅ                      ends with()  
 æŮzæšŦārzāžŌēŁĠæzd'äÿĂäÿłŻōāŦçŽĎāĖĖāōzāž\$æŸřāŦŁāJLčŦłçŽĎāĀĀcærŦæĈijŽ

```
pyfiles = [name for name in os.listdir('somedir')
            if name.endswith('.py')]
```

árřăžŎæŮĜăžăăŘ■čŽďăŇžéĚ■ijŇăĵăăŔêĈĵijžĚĂĈĚŽŚăĵčť glob æĹŮ fnmatch  
 æĹăăĹŮăĂĈăŕŤăĉĈijŽ

```
import glob
pyfiles = glob.glob('somedir/*.py')

from fnmatch import fnmatch
pyfiles = [name for name in os.listdir('somedir')
            if fnmatch(name, '*.py')]
```

èóìèőž

eŌuāRŪçZŏā;Täy■çŽDāLŪeālaēYřā;LāōzæYŞçŽDīijNā;EæYřāĒūēŦāŽdçzŞædIJāRlæYřçZŏā;Täy■āō  
 āçCædIJā;jäēYæCşēŌuāRŪāĒūāzŪçŽDāĒCāfææAřiiijNærŦāçCæŪGāzūāđ gārRiiijNāfŏæŦzæŪūēŪŦç■Lç■  
 ä;āæLŪēōyēŦYēIJĀēçAä;ŦçŦlāLř os.path ælāālŪāy■çŽDāĠ;æŦræLŪçIĀ os.stat()  
 āĠ;æŦrælēæŦūēZEæŦræ■ōāĀCærŦāçCīijŽ

```
# Example of getting a directory listing

import os
import os.path
import glob
```

```

pyfiles = glob.glob('*.py')

# Get file sizes and modification dates
name_sz_date = [(name, os.path.getsize(name), os.path.
    ↳ getmtime(name))
    for name in pyfiles]
for name, size, mtime in name_sz_date:
    print(name, size, mtime)

# Alternative: Get file metadata
file_metadata = [(name, os.stat(name)) for name in pyfiles]
for name, meta in file_metadata:
    print(name, meta.st_size, meta.st_mtime)

```

æIJĀŖŌèƒŸæIJL'äyĀçĆžèeAæşlæĐŖçŽĐāŕsæŸŕijNæIJL'æŪŭāĀŽāIJlād'ĐçŖEæŪŖGāzŭāŖ■çijŪçāAé  
 éĀŽāyyæİèøšijNāĠ;æŦŕ os.listdir() èƒŦāŽđçŽĐāóđä;ŞāLŪèalāijŽæāzæ■ōçşçzçşézŸèød'çŽĐæŪŖGā  
 ā;EæŸŕæIJL'æŪŭāĀŽāzşāijŽççŕāLŕāyĀāžŽāy■èÇ;æ■cāyyèğççāAçŽĐæŪŖGāzŭāŖ■āĀĆ  
 āĖşāžŌæŪŖGāzŭāŖ■çŽĐād'ĐçŖEçŪŕóécŸŕijNāIJl5.14āŞN5.15āŕŖèŁĆæIJL'æŽŦ'èŕççzEçŽĐèøšèğçāĀĆ

## 7.14 5.14 āŖçŦŖæŪŖGāzŭāŖ■çijŪçāA

### éŪŕóécŸ

ā;āæČşā;ƒçŦŦāŌşāğNæŪŖGāzŭāŖ■æL'ğèaŖæŪŖGāzŭçŽĐl/OæŞ■ā;IJŕijNāzşāŕsæŸŕèŦŦ'æŪŖGāzŭāŖ■āzŭæŸ

### èğçāEşæŪzæāĻ

ézŸèød'æČĖāEŦāyNŕijNæL'ĀæIJL'çŽĐæŪŖGāzŭāŖ■éÇ;āijŽæāzæ■ŕ sys.  
 getfilesystemencoding() èƒŦāŽđçŽĐæŪŖGæIJŖçijŪçāAæİèçijŪçāAæLŪèğççāAāĀĆæŦŦāeÇŕijŽ

```

>>> sys.getfilesystemencoding()
'utf-8'
>>>

```

āeĆæđIJāŽāāyžæşŖçğ■āŌşāŽāā;āæČşāƒçŦŦèƒŽçğ■çijŪçāAŕijNāŖŕāzèā;ƒçŦŦāyĀāyŦāŌşāğNā■ŪèŁĆā

```

>>> # Write a file using a unicode filename
>>> with open('jalape\xflo.txt', 'w') as f:
...     f.write('Spicy!')
...
6
>>> # Directory listing (decoded)
>>> import os
>>> os.listdir('.')
['jalapeĀśo.txt']

```



```
>>> # Directory listing (raw)
>>> os.listdir(b'.') # Note: byte string
[b'jalapen\xcc\x83o.txt']

>>> # Open file with raw filename
>>> with open(b'jalapen\xcc\x83o.txt') as f:
...     print(f.read())
...
Spicy!
>>>
```

æ■čæĆä;äæL'ÀëġAīijNāIJlæIJĀāRŌäyd'äylæS■ä;IJäy■īijNā;Šä;ăczZæŪĠäzūčZÿāĖšāĠ;æTŗæĆ  
open() āŠN os.listdir() āijjæĀŠā■ŪèŁĆā■ŪçņęäyšæŪīīijNæŪĠäzūāR■čZĎād'DčŘEæŪzāijRāijZčĹ

## èőléőž

éĀŽāyÿæĹèëőīijNā;āäy■éIJĀèeAæNĖāfCæŪĠäzūāR■čZĎcijŪçāAāŠNèġčçăAīijNæŽōéĀZčZĎæŪĠäz  
ä;EæYŗīijNæIJL'ăžZæS■ä;IJçşzçzşāĖAèőÿčTlæLūéĀŽēfĠāŪčĎūāLŪæAūæĎRæŪzāijRāŌzāLZăžzāR■ā■  
èfZăžZæŪĠäzūāR■āRfèC;āijZčēđçġYāIJŗäy■æŪ■éCăžZéIJĀèeAād'DčŘEād'ġéĠRæŪĠäzūčZĎPythončĹN

èrzaRŪčZōā;TāzūéĀŽēfĠāŪčĎāġNæIJèġčçăAæŪzāijRād'DčŘEæŪĠäzūāR■āRfäzēæIJL'æTlčZĎéAġā  
ār;çōæfZæāūāijZāyæĹèäyĀăőZčZĎcijŪčĹNéŽ;ăžæāĀĆ

āĖšāžŌæLŠā■ŗäy■āRfèġčçăAçZĎæŪĠäzūāR■īijNērūāRCèĀĆ5.15ārRèŁĆāĀĆ

## 7.15 5.15 æL'Sā■ŗäy■āRĹæşTçZĎæŪĠäzūāR■

### éŮóécŸ

ā;ăçZĎčĹNāžRèŌūāRŪāžEäyĀäylčZōā;Tāy■čZĎæŪĠäzūāR■āLŪèāīijNā;EæYŗā;ŠāŏČērTçĹĀăŌzæL'S  
āĠçŌŗāžE UnicodeEncodeError āijCāyÿāŠNāyĀæĹāăēĠāčZĎæŪLæAŗāĀTāĀT  
surrogates not allowed āĀĆ

## èġcāEşæŪzæqĹ

ā;ŠæL'Sā■ŗæIJłçşēçZĎæŪĠäzūāR■æŪīīijNā;łçTlāyNéĹčZĎæŪzæşTāRfäzēéAġāĖ■èfZæāūčZĎéTŽè

```
def bad_filename(filename):
    return repr(filename)[1:-1]

try:
    print(filename)
except UnicodeEncodeError:
    print(bad_filename(filename))
```

## èòléõž

èŁŻäýÄärRèŁCèòléõžçŽDæÝřáIJłçijŰâEZâŁĚéazâd'ĐçŘEæŰĞäzúçşçzşçşçŽDçłNâžRæŰűäýÄäýłäý■ád  
ézÝëóđ'æČĚâĚťäýNřijNřPythonâĀĠăôŽæŁ'ĂæIJŁ'æŰĞäzúâR■éČ;ăűşçžRæăžæ■ó  
sys.getfilesystemencoding() çŽDâĀijçijŰçăĀēŁĠžĚăĀĆ  
ăĵEæÝřřijNæIJŁ'ăýĂăžŽæŰĞäzúçşçzşçşăžúăşæIJŁ'ăĵžăŁűēĉĀæśCēŁŽæăűăĀŽřijNăŽăæ■d'ăĚĀēőýăŁŽăžă  
ēŁŽçġ■æČĚâĚťäý■ăđ'łăýŷēġĀřijNăĵEæÝřæĂžăĵŽæIJŁ'ăžŽçŰłăŁűăĚŚéŽł'ēŁŽæăűăĀŽæŁŰēĂĚæÝřæŰăæŁ  
ăRřēČ;æÝřáIJłäýÄäýłæIJŁ'çijžéŽűçŽDăžçčăĀăý■çžŽ open()  
ăĠĵæŰřăĵăēĂşăžĚäýÄäýłäý■ăRŁēġĐēNČçŽDæŰĞäzúâR■)ăĂĆ

ăĵŞæŁġēăNçşăĵij os.listdir() èŁŽæăűçŽDâĠĵæŰřæŰřřijNēŁŽăžŽäý■ăRŁēġĐēNČçŽDæŰĞäzú  
ăýĂæŰžéłçřijNăôČăý■ēČ;ăžĚăžĚăRłæÝřăýçăĵČēŁŽăžŽäý■ăRŁæăĵçŽDâR■ăŰăĂĆēĀNăRēăýĂæŰžéłçřij  
PythonăřžēŁŽäýłēŰőéçÝçŽDēġčăĚşæŰžæăŁæÝřăžŎæŰĞäzúâR■ăý■ēŎăRŰŰæIJłēġçčăĀçŽDâŰēŁČăĀĵæ  
\\xhhăžúâRĚăôČæÝăârDæŁRUnicodeăŰŰçņē \\udchhēăłçđ'žçŽDæŁ'ĂēřŞçŽDâĀłăžçčŘĚçijŰçăĀăĀłăĂĆ  
ăýNēłçăýÄäýłăçNă■RăĵŰçđ'žăžĚăĵ;ŞăýÄäýłäý■ăRŁæăĵçŽDăĵŰłŰēăłäý■ăRŋæIJŁ'ăýÄäýłæŰĞäzúâR■ăýžł  
łēĀNăý■æÝřUTF-8çijŰçăĀ)æŰűçŽDæăűăR■řijŽ

```
>>> import os
>>> files = os.listdir('.')
>>> files
['spam.py', 'b\\udce4d.txt', 'foo.txt']
>>>
```

ăēČăđIJăĵăæIJŁ'ăžçčăĀēIJĂēĉĀæŞ■ăĵIJæŰĞäzúâR■æŁŰēĂĚârĚæŰĞäzúâR■ăĵăēĂŞçžŽ  
open() èŁŽæăűçŽDâĠĵæŰřřijNăýĂăŁĠēČ;ēČ;æ■čăýŷăűēăĵIJăĂĆ  
ăRłæIJŁ'ăĵ;ăæČşēĚĀēçŞăĠŽæŰĞäzúâR■æŰűăŁ'■ăĵŽççřăŁřăžŽēžçČē(ăřŰăēČăŁŞă■rēçŞăĠžăŁřăşRăž  
çŁ'žăŁŋçŽDřijNăĵ;ăĵăæČşæŁŞă■řăýŁēłççŽDæŰĞäzúâR■ăŁŰēăłăŰřřijNă;ăçŽDçłNâžRăřşăĵŽăŰ'æžČřijŽ

```
>>> for name in files:
...     print(name)
...
spam.py
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
UnicodeEncodeError: 'utf-8' codec can't encode character '\udce4' in
position 1: surrogates not allowed
>>>
```

çłNâžRăŰ'æžČçŽDăŎŞăŽăârşæÝřă■Űçņē \\udce4 æÝřăýÄäýłēłđæŞŰçŽDUni-  
codeăŰŰçņēăĂĆăôČăĚűăôđæÝřăýÄäýłēčŋçġřăýžăžçčŘĚăŰŰçņēăřžçŽDâRŋăŰŰçņēçžDâRŁçŽDâRŎă■ŁēČ  
çŰşăžŎçijžăřŞăžĚăŁ■ă■ŁēČłăĚĚřijNăŽăæ■d'ăôČæÝřăýłēłđæŞŰçŽDUnicodeăĂĆ  
æŁ'ĂăžēřijNăŰřăýĂēČ;æŁRăŁşēçŞăĠžçŽDæŰžæŞŰăřşæÝřă;ŞēĀĠăŁřăý■ăRŁæşŰæŰĞäzúâR■æŰűēĠĠăRł  
ăřŰăēČăRăřăžăârĚăýŁēŁřăžçčăĀăłôăŰžăēČăýNřijŽ

```
>>> for name in files:
...     try:
...         print(name)
...     except UnicodeEncodeError:
...         print(bad_filename(name))
...
>>>
```

```
spam.py
b\udce4d.txt
foo.txt
>>>
```

åIJÍ bad\_filename() åĜ;æTträy■æÅŒæũad'Đç;óåRÚåEşazŒä;æeĜłaušāĀĆ  
årĖad'ŪäyÄäyłéĀL'æNl'åršæYřéĀŽefĜæšRçg■æŪzâijRéĜ■æŪřçijŪčāAiiĵŅçd'žäĭNæĆäyŅiiĴ

```
def bad_filename(filename):
    temp = filename.encode(sys.getfilesystemencoding(), errors=
    ↳ 'surrogateescape')
    return temp.decode('latin-1')
```

èřSèĀĚæşĬ:

```
surrogateescape:
èfžçg■æYřPythonåIJłçziĀd'ĝeĀłāŁEęİcāŘSOSçŽĐAPIäy■æL'Āä;ŁçŤłçŽĐeŤŽèřřad'ĐçŘĖåŻłii.
åŏČèĀ;äžěäyĀçg■äijYřéŽĚçŽĐæŪzâijŘad'ĐçŘĖçŤśæş■ä;IJçşžçžşæŘŘä;žçŽĐæŤřæ■ŏçŽĐçijŪčāAe
åIJłèĝčçāAāĜžéŤŽæŪüäijžårĖāĜžéŤŽā■ŪeŁĀ■YāĆłāŁřäyĀäyłā;Łåršècñä;ŁçŤłāŁřçŽĐUnicode
åIJłçijŪčāAæŪüårĖeĆčāžžéŽŘèŪŘāĀijårŁeŁYāŒşāžđāŒşāĚŁeĝčçāAāđ'sèt'èçŽĐā■ŪeŁĀžŘāŁŪ
åŏČäy■äzĚåržäžŒŒS_
↳ APIéİdāyŷæIJL'çŤłiiĵŅžşèĀ;ā;ŁāŏžæYşçŽĐad'ĐçŘĖāĚüāžŪæĈēāĖŁäyŅçŽĐçijŪčāAēŤŽèřřā
```

ä;ŁçŤłēfZäyłçŁŁæIJñāžĝçŤşçŽĐe;şāĜžæĆäyŅiiĴ

```
>>> for name in files:
...     try:
...         print(name)
...     except UnicodeEncodeError:
...         print(bad_filename(name))
...
spam.py
bĀd'd.txt
foo.txt
>>>
```

èŁZäyĀårŘeŁĆäyžécYårŘèĀ;äijŽècñad'ĝeĀłāŁEęřzeĀĚæL'ĀāŁç;ŤēāĀĆä;ĖæYřæĆæđIJä;ååIJłçijŪāĖ  
åršāŁĚēāžā;ŪeĀĈeŽşāŁřèŁZäyłāĀĀŘēāŁZā;åårŘèĀ;äijŽåIJłæşŘäyłāŚłæIJñècñāŘñāŁřāŁđāĚñāŏđ'āŒžèřĀ

## 7.16 5.16 áćđāŁæŁŪæŤzāRŸaušæL'ŞâijĀæŪĜäzŭçŽĐçijŪčāA

éŪŒécŸ

ä;āæĈşåIJłäy■āĖşēŪ■äyĀäyłaušæL'ŞâijĀçŽĐæŪĜäzŭāL'■æŘŘäyŅāćđāŁæāŁŪæŤzāRŸāŏĈçŽĐUnicode

## èġċăEşşæŪzæąĹ

æĊăđIJă;ăæĊşşzŻăyĂăyĹăzēăžŊēĤZăĹŪăĹăijRăL'ŞăijĂşŻĐăŪĠăzŭăŭăăĹăUnicodeşijŪċăA/èġċăA  
ăŔăzēă;ĤċŤĹ io.TextIOWrapper() âŕzēsăăŊĖċĖăăŐCăĂĊăŕŤăĊĭijŻ

```
import urllib.request
import io

u = urllib.request.urlopen('http://www.python.org')
f = io.TextIOWrapper(u, encoding='utf-8')
text = f.read()
```

æĊăđIJă;ăæĊşăĹăŤzăyĂăyĹăŭşşzRăL'ŞăijĂşŻĐăŪĠăIJăăĹăijRċŻĐăŪĠăzŭċŻĐċijŪċăAăŪzăijR  
detach() æŪzæşŤċġzéZđ'æŐĹ'ăŭşăăŸăIJĹċŻĐăŪĠăIJăċijŪċăAăşĊĭijŊ  
ăžŭă;ĤċŤĹăŪŕċŻĐċijŪċăAăŪzăijRăzċăZĤăĂĊăyŊéĹăĤŕăyĂăyĹăIJĹ sys.stdout  
ăyĹăĤăĹăŏăŤzċijŪċăAăŪzăijRċŻĐăĹăŊăŔĭijŻ

```
>>> import sys
>>> sys.stdout.encoding
'UTF-8'
>>> sys.stdout = io.TextIOWrapper(sys.stdout.detach(), encoding=
↳ 'latin-1')
>>> sys.stdout.encoding
'latin-1'
>>>
```

ēĤZăăŭăăŹăŔŕēĊ;ăijŻăyăŪăă;ăċŻĐċzĹċŋŕĭijŊēĤZēĠăŹăĖăzĖăŸŕăyăzăžEăejŤċđ'žēĂăŭşăĂĊ

## ëŏĹëőž

I/OċşşzċşşĤŝăyĂċşşăĹŪċŻĐăşĊăŋăăđĐăzžēĂăŊăĹŔăĂĊă;ăăŔăzēēŕŤċĹĂēĤŔēăŊăyŊéĹċēĤZăyĹăŞăă

```
>>> f = open('sample.txt', 'w')
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> f.buffer
<_io.BufferedWriter name='sample.txt'>
>>> f.buffer.raw
<_io.FileIO name='sample.txt' mode='wb'>
>>>
```

ăIJĹēĤZăyĹăĹăŊăŔăyăĭijŊio.TextIOWrapper æŸŕăyĂăyĹċijŪċăAăŤŊēġċăăAŪ-  
nicodeċŻĐăŪĠăIJăăđ'ĐċŔĖăşĊĭijŊ io.BufferedWriter  
æŸŕăyĂăyĹăđ'ĐċŔĖăzŊēĤZăĹŪăŤŕăăŏċŻĐăyēċijŞăĖşċŻĐĹ/OăşĊĭijŊ io.FileIO  
æŸŕăyĂăyĹăĹăċđ'žăŞăă;IJşşzċşşăžŤăşĊăŪĠăzŭăŔŔēĤŕċŋċşŻĐăŐşăġŊăŪĠăzŭăĂĊ  
ăċđăĹăăĹŪăŤzăŔŸăŪĠăIJăċijŪċăAăijŻăŭĹ'ăŔĹăċđăĹăăĹŪăŤzăŔŸăIJăăyĹăĹéĹċşŻĐ  
io.TextIOWrapper âşĊăĂĊ

ăyĂăĹăŊăĹēċŏşŕĭijŊăĊŔăyĹéĹăĹăŊăŔēĤZăăŭăăŹžēĤĠēŏĹéŪŏăşđăăĠăăĭijăĹēċŹŤ'æŐēăŞăă;IJăyăăŔŊċ  
ăĹăŊăĊĭijŊăĊăđIJă;ăēŕŤċĹĂă;ĤċŤĹăyŊéĹċēĤZăăŭăăŻĐăĹăăIJăŕŤzăŔŸċijŪċăAċIJŊċIJăijŻăŔŤşĤŝăžĂă

```
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> f = io.TextIOWrapper(f.buffer, encoding='latin-1')
>>> f
<_io.TextIOWrapper name='sample.txt' encoding='latin-1'>
>>> f.write('Hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: I/O operation on closed file.
>>>
```

çŞæđIĴăĠžēŤŽăẼĲijŊăŽăăÿžfcŽĎăŎşăĝŊăĀijăũşçzŔēcńçăť'ăĲăžĒăžűăĔşēŬăăžĒăžŤăşĆçŽĎăŬĠăă  
 detach() æŬžæşŤăijŽăŬăĲijĂæŬĠăžűçŽĎăĲĂăăűăşĆăžűēŤăŽđçñăžŊăşĆĲijŊăžŊăŔŎăĲĂăăűăş

```
>>> f = open('sample.txt', 'w')
>>> f
<_io.TextIOWrapper name='sample.txt' mode='w' encoding='UTF-8'>
>>> b = f.detach()
>>> b
<_io.BufferedWriter name='sample.txt'>
>>> f.write('hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: underlying buffer has been detached
>>>
```

ăÿĂæŬēæŬăĲijĂæĲĂăăűăşĆăŔŎĲijŊăĵăăŕşăŔăžēççzŽēŤăŽđççŞæđIĴăũžăĹăăÿĂăÿŤăŬŕçŽĎăĲĂăăűăş

```
>>> f = io.TextIOWrapper(b, encoding='latin-1')
>>> f
<_io.TextIOWrapper name='sample.txt' encoding='latin-1'>
>>>
```

ăŕĲçŏăăũşçzŔăŔŞăĵăæĲijŤçđ'žăžĒăŤžăŔŶçĲijŬçăĂçŽĎăŬžæşŤĲijŊă  
 äĲĒăŶŕăĵăæŤŶăŔŕăžēăĹŕçŤĲēŤŽçĝăăĹĂăĲŕăĲăŤžăŔŶăŬĠăžűēăŊăđ'ĐçŔĒăĂĂăŤŽēŕŕăĲžăĹŭăžēăŔĹăă

```
>>> sys.stdout = io.TextIOWrapper(sys.stdout.detach(), encoding=
↳ 'ascii',
...                                     errors='xmlcharrefreplace')
>>> print('Jalape\u00f1o')
Jalape&#241;o
>>>
```

æşŕăĎŔăÿŊăĲĂăŔŎēĲşăĠžăÿăçŽĎăĲđASCIIăăŬçņē Āś æŶŕăēĆăĲŤēcń &#241;  
 äŔŬăžççŽĎăĂĆ

## 7.17 5.17 āĖā■ŮēŁĆāĖŻāĖĖāŮĖāĬñāŮĖāžŮ

### ēŮōēćŸ

äĭāāĈşāĬĬāŮĖāĬñāĭāĭĖāĖŁŸāĭĖāĈŻĖāŮĖāžŮāŷ■āĖŻāĖĖāŮşāĖŃĈŻĖā■ŮēŁĆāĖŤŖā■ōāĀĈ

### èġĉāĖşāŮžāēĭ

āĖā■ŮēŁĆāĖŤŖā■ōĉŹŤ æŮōāĖŻāĖĖāŮĖāžŮĉŻĖĭĭşāĖşāŃžā■şāŖŕĭĭñāĭŃāēĈĭĭż

```
>>> import sys
>>> sys.stdout.write(b'Hello\n')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str, not bytes
>>> sys.stdout.buffer.write(b'Hello\n')
Hello
5
>>>
```

ĉşžāĭĭĭĈŻĖĭĭñēĈĭāđŸēĀŻēĖĖĖŖžāŖŮāŮĖāĬñāŮĖāžŮĉŻĖ  
āśđāĖĖāĖĖāŖžāŖŮāžŃēĖŻāĭŮāŤŖā■ōāĀĈ

buffer

### èŮŮēŮž

ĬŮĉşžĉžşāžēāśĈĉžġĉžşāđĖĖĈŻĖāĭĉāĭĖāđĖāžēāŃāĖĖĖāĖĖāĀĈ  
āŮĖāĬñāŮĖāžŮāŷŖēĀŻēĖĖāĬĬāŷāāŷĭāēŃēāĬĬĈĭĭşāĖşĈŻĖāžŃēĖŻāĭŮāĭāĭĖāŮĖāžŮāŷĖāĀĉđāĖāāŷāāŷ  
buffer āśđāĖĖāĖĖāŖşāŖžāžŤĈŻĖāžŤāśĈāŮĖāžŮāĀĈāēĈāđĬāĭāĈŹŤ æŮēēŮēĖŮōāŮĈĈŻĖĖŖāŖşāĭĭĈĉžŤē

āĬñāŖŖēŁĈāĭŃā■ŖāśŤĉđŹĉŻĖ sys.stdout āŖŖēĈĭĭĬñēŮāĖēāĬĬĈĉžĈĖĖāĖāĀĈ  
ēžŸēŮđ' æĈĖāĖĭāŷŃĭĭñsys.stdout æĀžāŷŖāžēāŮĖāĬñāĭāĭĖāĖŁŸāĭĖāĈŻĖāĀĈ  
äĭĖāŷŖāēĈāđĬāĭāāĬĬāĖŻāŷāāŷĭēĬāēēāĖŁŸā■ŖāžŃēĖŻāĭŮāŤŖā■ōāĖŖāāĖāĖāĖēĭşāĖžĈŻĖēĖĖāĬñāŮĖāžŮ

## 7.18 5.18 āĖāŮĖāžŮāŖŖēĖŖĉņēāŃĖēĈĖāĖĖāŖāŖžēşā

### ēŮōēćŸ

äĭāāĬĬāŷāāŷĭāŖžāžŤāžŮāş■āĭĬĈşžĉžşāŷĖāŷāāŷĭāŷāĖŁŸāĭĖāĈŻĖāŮēĀŻēāş(āŖŤāēĈāŮĖāžŮāĀĈāĈ  
äĭāāĈşāŖĖāŮĈāŃĖēĈĖāĖĖāŷāŷĭāēŹŤ ēŃŸāśĈĈŻĖĖPythonāŮĖāžŮāŖžēşāāĀĈ

### èġĉāĖşāŮžāēĭ

āŷāāŷĭāŮĖāžŮāŖŖēĖŖĉņēāŃŃāŷāāŷĭāĖŁŸāĭĖāĈŻĖāŮēĀŻēāŮĖāžŮāŷŖāŷ■āŷāāŷĖāŷĖāĀĈ  
āŮĖāžŮāŖŖēĖŖĉņēāžēāžēāŷŖāŷāāŷĭĈŤśāş■āĭĬĈşžĉžşāŃĖāŮŻĈŻĖāŤŤ æŤŕĭĭĭĈŤĭāĖēāŃĖāžĉāşŖāŷĭĈş  
āēĈāđĬāĭāĈĉŖāŷāĖāĬĬēĖŻāžĖāŷāŷĭāŮĖāžŮāŖŖēĖŖĉņēĭĭñāĭāāŖŖāžēēāĀŻēĖĖāĭĭĈŤĭ

open()  
# Open a low-level file descriptor  
import os  
fd = os.open('somefile.txt', os.O\_WRONLY | os.O\_CREAT)  
  
# Turn into a proper file  
f = open(fd, 'wt')  
f.write('hello world\n')  
f.close()

# Create a file object, but don't close underlying fd when done  
f = open(fd, 'wt', closefd=False)  
...  
  
# Create a file object, but don't close underlying fd when done  
f = open(fd, 'wt', closefd=False)  
...  
  
# Create a file object, but don't close underlying fd when done  
f = open(fd, 'wt', closefd=False)  
...

## echo server

from socket import socket, AF\_INET, SOCK\_STREAM  
  
def echo\_client(client\_sock, addr):  
 print('Got connection from', addr)  
  
 # Make text-mode file wrappers for socket reading/writing  
 client\_in = open(client\_sock.fileno(), 'rt', encoding='latin-1',  
 closefd=False)  
  
 client\_out = open(client\_sock.fileno(), 'wt', encoding='latin-1',  
 closefd=False)  
  
 # Echo lines back to the client using file I/O  
 for line in client\_in:  
 client\_out.write(line)  
 client\_out.flush()  
  
 client\_sock.close()  
  
def echo\_server(address):  
 sock = socket(AF\_INET, SOCK\_STREAM)  
 sock.bind(address)  
 sock.listen(1)  
 while True:

```
client, addr = sock.accept()
echo_client(client, addr)
```

éIJĀēēAēG■ÇZāijžēČÇŽDāyĀçCZæYřijNāyLēlčçŽDāŁNā■ŘāzĚāzĚæYřāyžāzEāēijTčd'žāEĚçŁčŽD  
open() āĠ;æTřçŽDāyĀāyŁçL'zæĀġijNāzūāyTāzšāRlēĀČçTlāzŌāšžāzŌUnixçŽDçšçzçšāĀČ  
āēČāđIJā;āæČšāřEāyĀāyŁçszæŪĠāzūāēŌēāRčā;IJçTlāIJlāyĀāyŁāēŪāēŌēā■ŪāzūāyNāIJZā;āçŽDāzčçāAāRřā  
makefile() æŪzæšTāĀČ ā;EæYřāēČāđIJāy■ēĀČēZSāRfçġzæd'■æĀġçŽDēřlīijNēČčāyLēlčçŽDēġčāEšæ  
makefile() æĀġēČ;æŽt'āē;āyĀçCZāĀČ

ā;āāzšāRřāzēā;ŁçTlēfZçġ■æLĀæIJrāēāđDēĀāyĀāyŁāLŋāR■ijNāĚAēōyāzēāy■āRŊāzŌčŋāyĀæŋā  
āŁNāēČīijNāyNēlčēijTčd'žāēČā;TāLZāzžāyĀāyŁæŪĠāzūāřzēšāijNāōČāĚAēōyā;āēŁSāĠzāzNēŁZāLŪāTřāē

```
import sys
# Create a binary-mode file for stdout
bstdout = open(sys.stdout.fileno(), 'wb', closefd=False)
bstdout.write(b'Hello World\n')
bstdout.flush()
```

ār;čōāāRřāzēāřEāyĀāyŁāūsā■YāIJlçŽDæŪĠāzūāēRRēŁřçŋēāNĚēčĚāēLŘāyĀāyŁæ■čāyçŽDæŪĠāzūāřzē  
ā;EæYřēēAēšlāēDŘçŽDæYřāzūāy■æYřāēLĀæIJLçŽDæŪĠāzūāēlāāijRēČ;ēčŋāēTřāēNāijNāzūāyTāēšRāzŽç  
(çL'žāLŋāYřāēŪL'āRĀlāřēT'Zēřrād'DçRĚāĀāæŪĠāzūçzSāř;æĪāzūç■Lç■LçŽDæŪūāĀŽ)āĀČ  
āIJlāy■āRŊçŽDæš■ā;IJçšçzçšāyLēfZçġ■ēāNāyžāzšæYřāy■āyĀæāūijNçL'žāLŋçŽDīijNāyLēlčçŽDāŁNā■R  
āēLŠēřt'āzEēŁZāzLād'ŽīijNāēDŘāēĀlāřsæYřēōl'ā;āāĚĚāLĚāēŁNērTēĠāūsçŽDāōđçŌřāzčçāAīijNçāōāŁlāōČē

## 7.19 5.19 āLZāzžāy'tæŪūæŪĠāzūāšNāēŪĠāzūād'z

### ēŪōēčY

ā;āēIJĀēēAāIJlčlNāžRāēL'ġēāNāēŪūāLZāzžāyĀāyŁāy'tæŪūæŪĠāzūāēLŪçZōā;TīijNāzūāyNāIJZā;ŁçTlā

### ēġčāEšæŪzæāŁ

tempfile æĪāāIŪāy■æIJL'ā;Lād'ŽçŽDāĠ;æTřāRřāzēāōNāēLŘēŁZāzžāŁāāĀČ  
āyžāzEāŁZāzžāyĀāyŁāNēāR■çŽDāy'tæŪūæŪĠāzūūijNāRřāzēā;ŁçTlā tempfile.  
TemporaryFile īijŽ

```
from tempfile import TemporaryFile

with TemporaryFile('w+t') as f:
    # Read/write to the file
    f.write('Hello World\n')
    f.write('Testing\n')

    # Seek back to beginning and read the data
    f.seek(0)
    data = f.read()
```



```
# Temporary file is destroyed
```

æŁŨèĀĖijŃæĆæđIä;ääŮIJæñćijŃä;æŁŸäŔŕäzëäČŔèŁŹæăă;ŁçŦlăyt' æŮŭæŮĠăzŭijŹ

```
f = TemporaryFile('w+t')
# Use the temporary file
...
f.close()
# File is destroyed
```

TemporaryFile() çŽĐçññäyĀăyġlāŔĆæŦŕæŸŕæŮĠăzŭăġăijŔijŃëĀŽăyŷæĲèőšæŮĠăIJŃăġăijŔă  
w+t ĩijŃăžŃëŁŹăĲăġăijŔă;ŁçŦl w+b āĀĆ èŁŹăyġăġăijŔăŔŃæŮŭæŦŕæŃĀèŕzăŖŃăĲŹæŞă;IJijŃăIJlèŁŹ  
TemporaryFile() âŔëăđ'ŮèŁŸæŦŕæŃĀèŭşăĲĖĖ;őçŽĐ open()  
ăĠ;æŦŕăyĀăăŭçŽĐăŔĆæŦŕăĀĆæŦŕăĖĆijŹ

```
with TemporaryFile('w+t', encoding='utf-8', errors='ignore') as f:
    ...
```

ăIJlăđ'ġăđ'ŽæŦŕUnixçşzçşăyĲijŃëĀŽèŁĠ TemporaryFile()  
ăĲŹăžçŽĐæŮĠăzŭëĆ;æŸŕăŃăŔăçŽĐijŃçŦŹèĠşèđçŽőă;ŦéĆ;æşăăIJL'ăĀĆ  
ăĖĆæđIä;ăăČşæĲŖçăŕ'èŁŹăyġéŽŔăĲŭijŃăŔŕäzëä;ŁçŦl NamedTemporaryFile()  
æĲăžçæŽŕăĀĆæŦŕăĖĆijŹ

```
from tempfile import NamedTemporaryFile

with NamedTemporaryFile('w+t') as f:
    print('filename is:', f.name)
    ...

# File automatically destroyed
```

èŁŹéĠŃijŃëćŃæĲŖăijĀæŮĠăzŭçŽĐ f.name âşđæĀġăŃĖăŔŃăžĲèŕëăyt' æŮŭæŮĠăzŭçŽĐæŮĠăzŭăŔă  
ă;Şă;ăĖIJăĖĀăŕĲæŮĠăzŭăŔăăijăĖĀŞçžŽăĖŭăžŮăžççăĀæĲæĲŖăijĀèŁŹăyġæŮĠăzŭçŽĐæŮŭăĀŽijŃëŁŹă  
ăŖŃ TemporaryFile() äyĀăăŭijŃçžşđđIJæŮĠăzŭăĖşëŮŭæŮŭăijŹëćŃëĠăĲăĲăĲăéŽđ' æŖĲăĀĀĆ  
ăĖĆæđIä;ăăyăăČşèŁŹăžĲăĀŽijŃăŔŕäzëăijăĖĀŞăyĀăyġăĖşëŦőăŭăŔĆæŦŕ  
delete=False âşăŔŕăĀĆæŦŕăĖĆijŹ

```
with NamedTemporaryFile('w+t', delete=False) as f:
    print('filename is:', f.name)
    ...
```

ăyžăžĲăĲăžăyĀăyġăyt' æŮŭçŽőă;ŦijŃăŔŕäzëä;ŁçŦl tempfile.  
TemporaryDirectory() āĀĆæŦŕăĖĆijŹ

```
from tempfile import TemporaryDirectory

with TemporaryDirectory() as dirname:
    print('dirname is:', dirname)
    # Use the directory
```

```
...
# Directory and all contents destroyed
```

## ðóíèõž

TemporaryFile()                      ãÄÄNamedTemporaryFile()                      åŠŃ  
TemporaryDirectory()    åĜ;æŦř    åžŦëræŸřåd'ĐçŘĒäyt' æŮüæŮĜäzűçŽóå;ŦçŽĐæIJĀçóĀå■ŦçŽĐæŮ  
åIJĪäyÄäyĪæŽt'ä;ŎçŽĐçžgĀĻñijŃä;åāŦřäzëä;ŁçŦĪ    mkstemp()    åŠŃ    mkdtemp()  
æĪæĀĻŽāžžäyt' æŮüæŮĜäzűāŠŃçŽóå;ŦāĀĆæŦŦæĆřijŽ

```
>>> import tempfile
>>> tempfile.mkstemp()
(3, '/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-/tmp7fefhv')
>>> tempfile.mkdtemp()
'/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-/tmp5wvcv6'
>>>
```

ä;ĒæŸřijŃëŁŽāžŽāĜ;æŦřāžűäy■äijŽāÄžëŁŽäyĀæ■ëçŽĐçóaçŘĒäžĒæĀĆ  
ä;ŃāëĆřijŃāĜ;æŦř mkstemp()    äžĒäzĒāřsëŁŦāŽđäyÄäyĪāŎšāĝŃçŽĐDOSæŮĜäzűæŦŦëŁřçñëijŃä;æĒIJĀëç  
āŦŃæăüä;æëŁŸĒIJĀëçĀëĜĪăűsæyĒçŘĒëŁŽāžŽæŮĜäzűāĀĆ

éĀŽäyyæĪëðšřijŃäyt' æŮüæŮĜäzűāIJĪçşçzşéžŸëød'çŽĐä;■ç;őëćŋĀĻŽāžžijŃæŦŦæĆ  
/var/tmp æĻŮçşžäijijçŽĐāIJřæŮžāĀĆ äyžāžĒëŎūāŦŮçIJşăödçŽĐä;■ç;őijŃāŦřäzëä;ŁçŦĪ  
tempfile.gettempdir()    åĜ;æŦřāĀĆæŦŦæĆřijŽ

```
>>> tempfile.gettempdir()
'/var/folders/7W/7WZl5sfZEF0pljrEB1UMWE+++TI/-Tmp-'
>>>
```

æĻ'ĀæIJĻ'åŠŃäyt' æŮüæŮĜäzűçŽyāĒşçŽĐāĜ;æŦřëĆ;āĒæëöyā;æĒŽëŁĜä;ŁçŦĪĀĒşëŦōā■ŮāŦĆæŦř  
prefix āĀĀsuffix åŠŃ dir æĪëçĜĪăōŽāžŁçŽóå;ŦäžëāŦĻāŚ;āŦ■ëĝĐĀĻŽāĀĆæŦŦæĆřijŽ

```
>>> f = NamedTemporaryFile(prefix='mytemp', suffix='.txt', dir='/tmp
↳ ')
>>> f.name
'/tmp/mytemp8ee899.txt'
>>>
```

æIJĀāŦŎëŁŸæIJĻ'äyĀçĆžijŃāř;āŦřëĆ;äžææIJĀăĻ'āĒĪçŽĐæŮžāijŦä;ŁçŦĪ tempfile  
æĪăĪŮæĪæĀĻŽāžžäyt' æŮüæŮĜäzűāĀĆ āŦĒæŦñäžĒççŽā;ŞāĻ■çŦĪæĻüæŎĻæĪçëðŁëŮőäžëāŦĻāIJĀŮĜäzű  
ëçĀşĪæĐŦçŽĐæŸřäy■āŦŦçŽĐāžşāŦŦāŦřëĆ;äijŽäy■äyĀæăüāĀĆăŽāæ■d'ä;ææIJĀäë;éŸĒëřž  
ăōŸæŮžæŮĜæaç æĪëžĒëĝçæŽt'ād'ŽçŽĐçžĒëŁĀĀĆ

## 7.20 5.20 äÿŌäÿšëàŇçñráŔççŽĐæŦŕæ■óéĂžăĖą

### éŮóécŸ

äĳăæČšéĂžèĤĞäÿšëàŇçñráŔççŽĐæŦŕæ■ōīīĴŇăĚÿăđŇăĪJžæŽŕăŕšæŸŕăŤŇăÿĂăžŽçăñăžűèőĳăđ' ĠæĽŦ

### èğčăĖşæŮžæąĹ

ărĳčŏăĳăăŔŕăžžéĂžèĤĞăĳčŦĪPythonăĖĚçĳčŽĐĪ/OăĳăĳĪŮăĪăŏŇăĽŔèĤŽăÿĳăžžăĹăĳĳŇăĳĖăržăžŌăÿ  
pySerialăŇĚăĂČèĤŽăÿĳăŇĚçŽĐăĳčŦĪĪđăÿÿčŏĂă■ŦĳĳŇăĚĹăŏĹèčĖpySerialĳĳŇăĳčŦĪčşăĳĳĳăÿŇăĪčèĤŽă

```
import serial
ser = serial.Serial('/dev/tty.usbmodem641', # Device name varies
                    baudrate=9600,
                    bytesize=8,
                    parity='N',
                    stopbits=1)
```

èőĳăđ' ĠăŔ■ăržăžŌăÿ■ăŔŇçŽĐèőĳăđ' ĠăŤŇăş■ăĳĳçşçşşæŸŕăÿ■ăÿĂăăŭçŽĐăĂČ  
æŕŦăĖČĳĳŇăĪĪWindowsçşçşçşşăÿĹĳĳŇăĳăŔŕăžžăĳčŦĪŏ, ĳč■Ĺ'èăĳčđ'žçŽĐăÿĂăÿĹèőĳăđ' ĠăĪăĖĹ'ŞăĳĳĂéĂžă  
ăÿĂăŮĖçñráŔçæĹ'ŞăĳĳĂĳĳŇăČčăŕšăŔŕăžžăĳčŦĪ read() ĳĳŇreadline()ăŤŇ write()  
ăĢĳæŦŕŕŕăžăĖŽæŦŕæ■ăžĖĂăČăĴŇăĖČĳĳŽ

```
ser.write(b'G1 X50 Y50\r\n')
resp = ser.readline()
```

ăđ' ġăđ' ŽæŦŕæČĖăĖŦăÿŇĳĳŇçŏĂă■ŦçŽĐăÿšăŔçéĂžăĖăžŌăđ'ăŔŸăĳŮă■ĂăĹĖçŏĂă■ŦăĂČ

### èőĪèőž

ărĳčŏăĖăĳĪčăÿĹçĪŇĖŦăĪăĳĳčŏĂă■ŦĳĳŇăĚŮăŏđăÿšăŔçéĂžăĖăæĪĴăŮăăĂžăžşæŸŕăŇžéžçČĖçŽĐă  
æŌĪè■ŔăĳăăĳčŦĪčŇăÿĹăŮăžăŇĚăĖČ pySerial çŽĐăÿĂăÿĳăŌşăăžăæŸŕăŏČăŔŔăĳăžăžĖăržénŸçžğçĹ'žăĂ  
(æŕŦăĖČĖŮĖăŮŮĳĳŇăŖġăĹăăŦăĳĳŇçĳşăĖşăŇžăĹăăŮĳĳŇăŔăăĹŇă■Ŕèŏŏç■Ĺç■Ĺ')ăĂČăÿĳăÿĳăŇăŔĳĳ  
RTS-CTS æŔăăĹŇă■ŔèŏŏĳĳŇăĳăăŔĪĪĂĖĖĂçžŽ Serial()ăĳăăĂşăÿĂăÿĳă  
rtscts=True çŽĐăŔČæŦŕă■şăŔŕăĂČăĚŮăŏŸăŮăžăŮĢăăçĖĪđăÿÿăŏŇăŮđĳĳŇăžăăđ'ăĽşăĪĪĪĖĤéŽéĢŇă

ăŮăĹăĹèŏŕăĳăĹăĂăĪĴăăŮĹăŔĹăĹŔăÿšăŔççŽĐĪ/OĖČĳăŸŕăžŇĖĤŽăĹăĪăĳăĳĳŔçŽĐăĂČăžăăđ'ĳĳŇçă  
(ăĹŮăĪĴăŮăăĂžăĹăġăăŇăŮĢăĪŇçŽĐçĳĳŮčăĂ/èğččăĂăş■ăĳĳ)ăĂČ  
ăŔĖăđ'ŮăĳăĳăĖĪĂĖĖĂăĹăžăžăžŇĖĤŽăĹăŮçĳĳŮčăĂçŽĐăŇĢăžđ'ăĹŮăŦŕæ■ăăŇĚçŽĐăŮăăĂžĳĳŇstruct  
ăĳăĳĪŮăžşæŸŕĖĪđăÿÿăĪĴčŦĪçŽĐăĂČ

## 7.21 5.21 ăžŔăĹŮăŇŮPythonăŕžèşă

### éŮóécŸ

ăĳăĖĪĂĖĖĂăŕĖăÿĂăÿĳăPythonăŕžèşăăžŔăĹŮăŇŮăÿžăÿĂăÿĳă■ŮĖĹČăĖŦĳĳŇăžăăĳăŕăĖăŏČăĖĪăŮăĹŔăÿĂ

## èġċàEşæŨzæąŁ

årzāžŎāžRāŁŨāNŨæIJāæZŏéA■çŽDāAŽæşŤårşæŸřä;ŁçŤĪ pickle  
æłąąİŨāĀĆāyžāžEārEāyĀāyłāržèşąąŁā■ŸāŁřāyĀāyłæŨĠāzūāy■ījNāRřāžèèŁZæāūāAŽījŽ

```
import pickle

data = ... # Some Python object
f = open('somefile', 'wb')
pickle.dump(data, f)
```

āyžāžEārEāyĀāyłāržèşąąŁā■ŹāyžāyĀāyłā■ŨçñēāyşījNāRřāžèē;ŁçŤĪ pickle.  
dumps() īījŽ

```
s = pickle.dumps(data)
```

āyžāžEāžŎā■ŨēŁĆætAāy■æAćād'■āyĀāyłāržèşąījNā;ŁçŤĪ picle.load() æŁŨ  
pickle.loads() āĠ;æŤřāĀĆæŤæĆījŽ

```
# Restore from a file
f = open('somefile', 'rb')
data = pickle.load(f)

# Restore from a string
data = pickle.loads(s)
```

## èőłėőž

årzāžŎād'ġād'ŽæŤřāžŤçŤĪćĪNāžRæİēēőşījNdump() āšN load()  
āĠ;æŤřçŽDā;ŁçŤĪårşæŸřä;āæIJŁ'æŤŁä;ŁçŤĪ pickle æłąąİŨæŁ'ĀēIJĀçŽDāĒĪéĆĪāžEāĀĆ  
āőĆāRřéĀĆçŤĪāžŎçzĪād'ġéĆĪāŁEPythonæŤřæ■őçşzādNāšNçŤĪæŁūèĠāőŽāžŁ'çşzçŽDāržèşąąŁā■NāĀĆ  
āēĆādIJā;āçćřāŁřæşŘāyłāžŞāRřāžèèőł'ā;āāIJŁæŤřæ■őāžŞāy■āŁā■Ÿ/æAćād'■PythonāržèşąæŁŨēĀĒæŸřéĀ  
éĆčāžŁā;ŁæIJŁ'āRřéĆ;ēŁZāyłāžŞçŽDāžŤāsĆārşä;ŁçŤĪāžE pickle æłąąİŨāĀĆ

pickle æŸřāyĀçġ■PythonçŁ'žæIJŁ'çŽDēĠæRŘèŁřçŽDæŤřæ■őçijŨçāAāĀĆ  
éĀŽèŁĠæRŘèŁřījNēćnāžRāŁŨāNŨāRŎçŽDæŤřæ■őāNĒāRñæŤRāyłāržèşąāījĀāġNāšNçzŞāİşāžēāRŁāő  
āŽāæ■d'īījNā;āæŨāēIJĀæNĒāŁĆāržèşąēőřā;ŤçŽDāőŽāžŁ'īījNāőĆæĀžæŸřèĆ;āūēā;IJāĀĆ  
āy;āyłā;Nā■RīījNāēĆādIJēēAād'ĐçŘEād'ŽāyłāržèşąījNā;āāRřāžèèŁZæāūāAŽījŽ

```
>>> import pickle
>>> f = open('somedata', 'wb')
>>> pickle.dump([1, 2, 3, 4], f)
>>> pickle.dump('hello', f)
>>> pickle.dump({'Apple', 'Pear', 'Banana'}, f)
>>> f.close()
>>> f = open('somedata', 'rb')
>>> pickle.load(f)
[1, 2, 3, 4]
>>> pickle.load(f)
```

```
'hello'
>>> pickle.load(f)
{'Apple', 'Pear', 'Banana'}
>>>
```

ä;äæfYëC;äzRäLÜäNÜäG;æTrijNçszijNëfYæIJL'æÖëäRçijNä;EæYřçzŞæđIJæTřæ■öäzĚäzĚäřEäöČä

```
>>> import math
>>> import pickle.
>>> pickle.dumps(math.cos)
b'\x80\x03cmath\ncos\nq\x00.'
>>>
```

ä;ŞæTřæ■öäR■äzRäLÜäNÜäZðæIëçŽĐæUüäÄZrijNäijŽäĚLäAĞäöZæL'ÄæIJL'çŽĐæžŘæTřæ■öäUüäL  
æIäälÜäÄAçszäŠNäG;æTřäijŽëGłäLäēNLéIJÄärijäĚëëfZæĬäÄCärzäžŎPythonæTřæ■öëcnäy■äRNæIJžäŽIä  
æTřæ■öçŽĐäfiä■YäRřëC;äijZæIJL'éUöëcYrijNäZäyžæL'ÄæIJL'çŽĐæIJžäŽIëC;äfĚëäzëöfëUöäRNäyÄäyř  
æşI

ä■ČäyGäy■ëëAärzäy■äfaäzzçŽĐæTřæ■öä;ŁçTłpickel.load()äÄČ  
pickleäIJläläë;äUüäIJL'äyÄäyřäl'rä;IJçTłäřsæYřäöČäijŽëGłäläläë;çŽYäžTäIääIÜäž  
ä;EæYřæSřäyłäİRäžžäëCæđIJçSëëAŞpicklecçŽĐäüëä;IJäŎŞçŘĚijN  
äzÜäřsäräzëälZäzzäyÄäyłæAüäĐRçŽĐæTřæ■öärijëĜt'PythonæL'ğëaÑëŽRæĐRæNĞäöZçŽĐçşçç  
äZäæ■d' iijNäyÄäöZëëAäfiërApickleäRlälIJlçŽYäžŠäzNëÜt'äRřäzëëöd'ërAärzæÜççŽĐëğçæđR

æIJL'äzŽçszäđNçŽĐärzësæYřäy■ëC;ëcnäzRäLÜäNÜçŽĐäÄCëfZäzŽëÄŽäyÿæYřëCçäzZä;IëtÜäđ'Ué  
ærTäëCæL'ŞäijÄçŽĐæÜĞäzřijNç;ŞçzIJëfðæŎërijNçžŁNrijNëfZçlNrijNæäläyğç■Lç■L'äÄČ  
çTłæLüëGłäöZäZL'çszäRřäzëëÄŽëfGæRŘä;Z\_\_\_\_getstate\_\_\_\_()  
äŠN\_\_\_\_setstate\_\_\_\_()æÜzæşTæIëçzTëfGëfZäzŽëŽRäLüäÄČ  
äëCæđIJäöZäZL'äžEëfZäyd'äyłæÜzæşTrijNpickle.dump()  
äršäijŽërČçTl\_\_\_\_getstate\_\_\_\_()èŎüäRÜäzRäLÜäNÜçŽĐärzësäÄČ  
çszäijijçŽĐrijN\_\_\_\_setstate\_\_\_\_()äIJlär■äzRäLÜäNÜæUüëcnërČçTłäÄCäyžäžEæijTçd'žëfZäyłäüëä;IJäČ  
äyNëIëcæYřäyÄäyłäIJlæĚëČIäöZäZL'äžEäyÄäyłçžŁNä;Eäz■čDüäRřäzëäzRäLÜäNÜäŠNäR■äzRäLÜäNÜç

```
# countdown.py
import time
import threading

class Countdown:
    def __init__(self, n):
        self.n = n
        self.thr = threading.Thread(target=self.run)
        self.thr.daemon = True
        self.thr.start()

    def run(self):
        while self.n > 0:
            print('T-minus', self.n)
            self.n -= 1
            time.sleep(5)
```

```
def __getstate__(self):
    return self.n

def __setstate__(self, n):
    self.__init__(n)
```

èŕŦçĭĀēŁŖēąŇäÿŇēĭćçŽĐāžŔāĹŪāŇŪēŕŦēĭŇāžčċāĀĭĭž

```
>>> import countdown
>>> c = countdown.Countdown(30)
>>> T-minus 30
T-minus 29
T-minus 28
...

>>> # After a few moments
>>> f = open('cstate.p', 'wb')
>>> import pickle
>>> pickle.dump(c, f)
>>> f.close()
```

çĐūāŖŌēĀĀĠžPythonēğċæđŔāŽĭāžŭēĠāŖŕāŖŌāĒēŕŦēĭŇäÿŇĭĭž

```
>>> f = open('cstate.p', 'rb')
>>> pickle.load(f)
countdown.Countdown object at 0x10069e2d0>
T-minus 19
T-minus 18
...
```

äĭāāŖŕäžēçĭJŇāĹŕçžŁçĭŇāŖĹāēĠēŁēĹŇçŽĐēĠçŦšāžĒĭĭžŇāžŌäĭāçŇŇäÿĀæŇąāžŔāĹŪāŇŪāōĈçŽĐāĭJ

pickle āŕžāžŌād'ğāđŇçŽĐæŦŕæŋōçžšæđĐæŕŦæĈäĭŁçŦĭ array æĹŪ numpy  
æĭāāĭŪāĹŽāžçŽĐāžŇēŁŽāĹŭæŦŕçžĐæŦĹçŌĠāžŭäÿæŦŕäÿĀäÿĹēŇŦæŦĹçŽĐçĭĭŦçāĀæŪžāĭŔāĀĈ  
āēĈæđĭĭāĭæĭĭĀēēĀçğžāĹĭād'ğēĠŖçŽĐæŦŕçžĐæŦŕæŋōĭĭĭŇäĭāæĭĭĀāēĭæŦŕāĒĹāĭĭāÿĀäÿĹæŪĠāžŭäÿāŕĒāĒ  
(ēĭĭĀēēĀçŇŇäÿĹæŪžāžšçŽĐæŦŕæŇĀ)āĀĈ

çŦšāžŌ pickle æŦŕPythonçĹžæĭĭĹçŽĐāžŭäÿŦēŽĐçĭĭĀāĭĭāžŖçāĀäÿĹĭĭžŇæĹĀæĭĭĹæĈæđĭĭēĭĭĀēē  
äĭŇāēĈĭĭžŇāēĈæđĭĭæžŖçāĀāŖŦāĹāžĒĭĭžŇäĭāæĹĀæĭĭĹçŽĐāŦŦāĈĹæŦŕæŋōāŖŕēĈĭĭžēēçċāŦāĭŔāžŭäÿŦāŦ  
āĭēçŽĭæĭēēōĭĭžŇāŕžāžŌāĭĭĭæŦŕæŋōāžšāŦŇāŦŦæāçæŪĠāžŭäÿāŦŦāĈĹæŦŕæŋōæŪŦĭĭžŇäĭāæĭĭĀāēĭäĭŁçŦĹæŽ  
ēŁŽāžžçĭĭŦçāĀæĭĭĭĭĭŔæŽŦæāĠāĠēĭĭžŇāŖŕäžēēçŇäÿāŖŇçŽĐēŦēĭĀæŦŕæŇĀĭĭžŇāžŭäÿŦāžšēĈĭĭĹāēĭçŽĹ

æĭĭĀāŖŌäÿĀçĈžēēĀæšĭæĐŖçŽĐæŦŦŦ pickle æĭĭĹād'ğēĠŖçŽĐēĒçĭōēĀĹēāžāŦŇäÿĀāžžæçŦæĹŇ  
āŕžāžŌæĭĭĀäÿÿēğĀçŽĐāĭŁçŦĭĭĭæžŦĭĭžŇäĭāäÿēĭĭĀēēĀāŌžæŇēāŁĈēŁŽäÿĭĭžŇäĭĒæŦŕāēĈæđĭĭāĭæēēĀāĭĭĭ  
æĭĭĀāēĭāŌžæšēēŦēäÿĀäÿŇ āŦŦæŪžæŪĠæāç āĀĈ

## 8 ģņāĒ■ģņāīīĴæŦŕæ■ōçīĴŪčāAāŠŅad'DçŘĒ

ēĒZāyĀçņāāyžēēAēōlēōžā;ĒçŦĪPythonād'DçŘĒāŦŦĎçģ■āy■āŦŦæŪzāīĴçīĴŪčāAçŽĎæŦŕæ■ōīīŦæŦāē  
āŠŦæŦŕæ■ōçzŠæđDēČçāyĀçņāāy■āŦŦçŽĎæŦīīĴŦēĒZçņāāy■āīĴZēōlēōžçĴ'žæōĴçŽĎçōŪæşŦēŪōēēŦīīŦē.

Contents:

### 8.1 6.1 ērzāĒZCSVæŦŕæ■ō

#### ēŪōēēŦ

ā;āæČşērżāĒZāyĀāyĴCSVæāīĴāīĴŦçŽĎæŪĠāzŪāĀĆ

#### ēģčāĒşæŪzæāĴ

ārżāžŌād'ġād'ŽæŦŦçŽĎCSVæāīĴāīĴŦçŽĎæŦŕæ■ōērzāĒZēŪōēēŦīīŦēČ;āŦŦāzēā;ĒçŦĪ  
csv āžŠāĀĆ ā;ŦāēČīīĴāĀĠēō;ā;āāĴāyĀāyĴāŦ■āŦŦstocks.csvæŪĠāzŪāy■æĴĴ'āyĀāžZēČaçēĴāyČāĴžæŦŦ

```
Symbol,Price,Date,Time,Change,Volume
"AA",39.48,"6/11/2007","9:36am",-0.18,181800
"AIG",71.38,"6/11/2007","9:36am",-0.15,195500
"AXP",62.58,"6/11/2007","9:36am",-0.46,935000
"BA",98.31,"6/11/2007","9:36am",+0.12,104800
"C",53.08,"6/11/2007","9:36am",-0.25,360900
"CAT",78.29,"6/11/2007","9:36am",-0.23,225400
```

āyŦēĴčāŦŠā;āāšŦçđ'žāēČā;ŦārĒēĒZāžZæŦŕæ■ōērzāŦŪāyžāyĀāyĴāĒČçzĎçŽĎāžŦāĴŪīīĴ

```
import csv
with open('stocks.csv') as f:
    f_csv = csv.reader(f)
    headers = next(f_csv)
    for row in f_csv:
        # Process row
        ...
```

āĴĴāyĴēĴçŽĎāžççāAāy■īīŦ row āīĴZæŦŦāyĀāyĴāĴŪēāĴāĀĆāZāæ■đ'īīĴŦāyžāžĒēōēēŪōæşŦāyĴā■Ūæō  
row[0] ēōēēŪōSymbolīīŦ row[4] ēōēēŪōChangeāĀĆ

çŦšāžŌēĒZçģ■āyŦæāĠēōēēŪōēĀžāyāīĴāīĴŦçŦŦæŪŪæŪĒīīŦā;āāŦŦāzēēĀČēZŠā;ĒçŦĴāŠ;āŦ■āĒČçzĎæ

```
from collections import namedtuple
with open('stock.csv') as f:
    f_csv = csv.reader(f)
    headings = next(f_csv)
    Row = namedtuple('Row', headings)
    for r in f_csv:
        row = Row(*r)
```

```
# Process row
...
```

```
        row.Symbol      row.Change
äzcæŽfäyNæäGèøféÚõäÄĆ éIJÄëAæşlæĐRçŽĐæYřèŁŽäyłāRłæIJL'āIJlāLŪāR■æYřāŘŁæşTçŽĐPythonæä
ä;āāRřèC;éIJÄëAäēōæTžäyNāŌşāğNçŽĐāLŪāR■(āēCārEēIdæāGērEçñēā■ŪçñēæŽŁæ■cæLŘäyNāLŠçžŁä
āRēād'ŪäyÄäyłēĀL'æNl'ārśæYřārEæTřæ■ōērZāRŪāLŘäyÄäyłā■ŪāĚyāžRāLŪäy■āŌzāÄĆāRřäzèèŁŽæä
```

```
import csv
with open('stocks.csv') as f:
    f_csv = csv.DictReader(f)
    for row in f_csv:
        # process row
    ...
```

```
    āIJlēŁŽäyłçL'ŁæIJnäy■ñijNä;āāRřäzēä;ŁçTlāLŪāR■āŌžèøféÚōæfRäyÄēāNçŽĐæTřæ■ōäžEāÄĆæfTāēC
æLŪēÄĚ row['Change']
```

```
    äyžāžEāEŽāĚēCSVæTřæ■ōñijNä;āāz■çDūāRřäzēä;ŁçTlcsvæłāāIŪñijNäy■ēŁGèŁŽæŪūāÄŽāĚLāLŽäzā;
writer āržēšāāÄĆ;NāēC:
```

```
headers = ['Symbol', 'Price', 'Date', 'Time', 'Change', 'Volume']
rows = [('AA', 39.48, '6/11/2007', '9:36am', -0.18, 181800),
        ('AIG', 71.38, '6/11/2007', '9:36am', -0.15, 195500),
        ('AXP', 62.58, '6/11/2007', '9:36am', -0.46, 935000),
        ]

with open('stocks.csv', 'w') as f:
    f_csv = csv.writer(f)
    f_csv.writerow(headers)
    f_csv.writerows(rows)
```

```
    āēCædIJä;āæIJL'äyÄäyłā■ŪāĚyāžRāLŪçŽĐæTřæ■ōñijNāRřäzēāČRèŁŽæāūāÄŽñijŽ
```

```
headers = ['Symbol', 'Price', 'Date', 'Time', 'Change', 'Volume']
rows = [{ 'Symbol': 'AA', 'Price': 39.48, 'Date': '6/11/2007',
          'Time': '9:36am', 'Change': -0.18, 'Volume': 181800},
        { 'Symbol': 'AIG', 'Price': 71.38, 'Date': '6/11/2007',
          'Time': '9:36am', 'Change': -0.15, 'Volume': 195500},
        { 'Symbol': 'AXP', 'Price': 62.58, 'Date': '6/11/2007',
          'Time': '9:36am', 'Change': -0.46, 'Volume': 935000},
        ]

with open('stocks.csv', 'w') as f:
    f_csv = csv.DictWriter(f, headers)
    f_csv.writeheader()
    f_csv.writerows(rows)
```



## ěőłěőž

ä;ääžTěřěæĀzæYřäijYăĚĹéĀĹæŇŮ csvæĹaaĪŮăĹĚăĹšæĹŮěğčæđŘCSVæŤŕæ■őăĂĈăĹŊăĕĈiijŊă;ăăŖŮ

```
with open('stocks.csv') as f:
    for line in f:
        row = line.split(',')
        # process row
    ...
```

ä;ĤçŤĹěĹŽçğ■æŮžäijŘçŽĎäyĂäyĹçijžçĈžăŕšæYřă;ăäž■çĎŮéIJĂĕĕAăŮžăđ'ĎçŘĚăyĂăžŽæčYæĹŊçŽĎç  
æŕŤăĕĈiijŊăĕĈăđIJăšŘăžŽă■ŮăŕăăĀijĕĕŋăijŤăŖŮăŊĚăŽŤ'ĵiijŊă;ăäy■ăĹŮăy■ăŮžéŽđ'ĕĹŽăžŽăijŤăŖŮăĂĈ  
ăŖăăđ'ŮĵiijŊăĕĈăđIJăyĂäyĹĕĕŋăijŤăŖŮăŊĚăŽŤ'çŽĎă■ŮăŕăŕăăğăŖŊăĹJĹăyĂäyĹăŮăŖŮiijŊéĈčăžĹçĹŊăž

ézYĕód'æĈĚăĔăyŊĵiijŊcsv äžšăŖŕĕŕĚăĹŋMicrosoft Ex-  
celăĹĂă;ĤçŤĹçŽĎCSVçijŮçăAĕğĎăĹŽăĂĈ ĕĹŽăĹŮĕŕăžšæYřăIJĂăyŷĕğAçŽĎă;ĕăijŘĵiijŊăžŮăyŤăžšăijŽ  
çĎŮĕĂŊĵiijŊăĕĈăđIJă;ăăšĕçIJŊcsvçŽĎăŮĜăăçĵiijŊăŕšăijŽăŖšçŮŕăIJĹăĹăĹăđ'Žçğ■æŮžăšŤăŕĚăőĈăžŤçŤĹ  
ăĹŊăĕĈiijŊăĕĈăđIJă;ăăĈšĕŕžăŖŮăžĕtabăĹĚăĹšçŽĎăŤŕæ■ŕiijŊăŖŕăžĕĕĹŽăăŮăĂŽĵiijŽ

```
# Example of reading tab-separated values
with open('stock.tsv') as f:
    f_tsv = csv.reader(f, delimiter='\t')
    for row in f_tsv:
        # Process row
    ...
```

ăĕĈăđIJă;ăă■ĕăĹĹĕŕžăŖŮCSVæŤŕæ■őăžŮăŕĚăőĈăžŋĕ;ŋă■ĕăyžăš;ăŖ■ăĔĈçžĎĵiijŊéIJĂĕĕAăšĹăĎŖăŕž  
ăĹŊăĕĈiijŊăyĂäyĹCSVæăijăijŖăŮĜăžŮăIJĹăyĂäyĹăŊĚăŖŊéĹđăšŤăăĜĕŕĚçŋĕçŽĎăĹŮăđ'ŕĕăŊĵiijŊçšžăijĵiij

StreetĂăAddress,Num-Premises,Latitude,Longitude 5412ĂăŊĂăCLARK,10,  
→41.980262,-87.668452

ĕĹŽăăŮăIJĂçžĹăijŽăŕijĕĜŕ'ăĹĹăĹŽăžžăyĂäyĹăš;ăŖ■ăĔĈçžĎăŮăžğçŤšăyĂäyĹ  
ValueErrorăijĈăyŷĕĂŊăđ'sĕŕ'ĕăĂĈăyžăžĚĕğčăĔšĕĹŽĕŮĕĕYĵiijŊă;ăăŖŕĕĈ;ăy■ăĹŮăy■ăĔĹăŮžăĹăőăĕăĕă  
ăĹŊăĕĈiijŊăŖŕăžăĈŖăyŊéĹĕĕĹŽăăŮăIJĹéĹđăšŤăăĜĕŕĚçŋĕăyĹă;ĤçŤĹăyĂäyĹă■ĕăĹŽăĕĹĕĹăijŖăžĹăæ■ĈiijŽ

```
import re
with open('stock.csv') as f:
    f_csv = csv.reader(f)
    headers = [ re.sub('[^a-zA-Z_]', '_', h) for h in next(f_csv) ]
    Row = namedtuple('Row', headers)
    for r in f_csv:
        row = Row(*r)
        # Process row
    ...
```

ĕĹYăIJĹĕĜ■ĕĕAçŽĎäyĂçĈzéIJĂĕĕAăijžĕŕĈçŽĎăYřĵiijŊcsvăžğçŤšçŽĎăŤŕæ■őĕĈ;æYřă■Ůçŋĕăyšçšžă  
ăĕĈăđIJă;ăĕIJĂĕĕAăŽĕĹŽăăŮçŽĎçšžăđŊĕ;ŋă■ĕĈiijŊă;ăăĔĔĕăžĕĜĹăŮšăĹŊăĹăĹăŮžăăđçŮŕăĂĈ  
ăyŊéĹăĕăYřăyĂäyĹăIJĹCSVæŤŕæ■őăyĹăĹĹĜĕăŊăŮăŮăžŮçšžăđŊĕ;ŋă■ĕçŽĎăĹŊă■ŖiijŽ

```
col_types = [str, float, str, str, float, int]
with open('stocks.csv') as f:
    f_csv = csv.reader(f)
    headers = next(f_csv)
    for row in f_csv:
        # Apply conversions to the row items
        row = tuple(convert(value) for convert, value in zip(col_
→types, row))
    ...
```

āRēād' ŪriiŃäyŃēlċæŸřäyÄäylè;ñæ■ċā■ŪāĔyāy■çL'zāōŽā■ŪæōtçŽĎä;Ńā■ŘiijŽ

```
print('Reading as dicts with type conversion')
field_types = [ ('Price', float),
                 ('Change', float),
                 ('Volume', int) ]

with open('stocks.csv') as f:
    for row in csv.DictReader(f):
        row.update((key, conversion(row[key]))
                    for key, conversion in field_types)
    print(row)
```

éĀŽāyŷæĪēēōriiŃä;āāRřēČ;āžūāy■æČšèŁĠād'ŽāŌžēĀČēŽSèŁŽāžŽè;ñæ■céŪōécŸāĀĆ  
āĪĪāōđéŽĒæČĔāĔtāy■riiŃCSVæŪĠāžūēČ;æĹŪād'ŽæĹŪārŠæĪĪ'āžŽçijžād'sçŽĎæŢřæ■ōriiŃēēñċāt'āĪRçŽĪ  
āŽāæ■d'riiŃēŽd'ēĪdā;āçŽĎæŢřæ■ōçāōāōđæĪĪ'āĹēŽĪJæŸřāĠĔçāōæŪāēřřçŽĎiijŃāRēāĹŽā;āāĔĔēāžēĀČēŽ  
æĪĪāāRŌriiŃāēČæđĪJā;āēřžāRŪCSVæŢřæ■ōçŽĎçŽōçŽĎæŸřāĀŽæŢřæ■ōāĹĔæđRāŠŃçžšēōāçŽĎēřĪiij  
ā;āāRřēČ;ēĪĪāēēAçĪJŃäyĀçĪJŃ Pandas āŃĔāĀĆPandas  
āŃĔāRŃāžĔyāyŷēĪđāyŷæŪžā;ŁçŽĎāĠ;æŢřāRŃ pandas.read\_csv()  
riiŃ āōČāRřāžēāĹāē;ĪCSVæŢřæ■ōāĹřāyĀäyĹ DataFrame āržèšāy■āŌžāĀĆ  
çĎūāRŌāĹĹ'çĹĔēŁŽāyĹāržèšā;āāršāRřāžēçĹšæĹRāRĎçg■ā;ċāiŃRçŽĎçžšēōāāĀĀēĔĠæžd'æŢřæ■ōāžēāRĹæĹ  
āĪĪ6.13ārRēĹČāy■āiijŽæĪĪ'ēĔŽæāūāyĀäyĹä;Ńā■ŘāĀĆ

## 8.2 6.2 èřžāĔŽJSONæŢřæ■ō

### éŪōécŸ

ä;āæČšēržāĔŽJSON(JavaScript Object Notation)çijŪčāĀæāiijāiŃRçŽĎæŢřæ■ōāĀĆ

### èğċāĔşæŪžæāĹ

j s o n æĹāāĪŪāēRŘä;ŽāžĔyāyĀçg■ā;ĹçōĀā■ŢçŽĎæŪžāiŃRæĪēçijŪčāĀāŠŃèğċċāĪJSONæŢřæ■ōāĀĆ  
āĔŪāy■āyđ'āyĹāyžèēAçŽĎāĠ;æŢřæŸř json.dumps() āŠŃ json.loads()  
riiŃ ēēĀærŢāĔŪāžŪāžRāĹŪāŃŪāĠ;æŢřāžŠæČpicklēçŽĎæŌēāRċārSā;Ūād'ŽāĀĆ  
āyŃēĪċæiŃŢçd'žāēČā;ŢārĔyāyĀäyĹPythonæŢřæ■ōçžšæđĎē;ñæ■ċāyžJSONriiijŽ

```
import json

data = {
    'name' : 'ACME',
    'shares' : 100,
    'price' : 542.23
}

json_str = json.dumps(data)
```

äyÑéÍcæijTçd'zæCä;TärEäyÄäyJSONçijÛçäAçŽDā■Ûçñäyšè;ñæ■cāŽdäyÄäyPythonæTṛæ■õçzŠædĪ

```
data = json.loads(json_str)
```

æĈCædIJä;æēAād'DçŘEçŽDæYṛæŮĠäzűēĂŇäy■æYṛā■ÛçñäyšiiijNä;ääRfäzēä;ŁçTĪ  
 json.dump() āŠŇ json.load() æĪēçijÛçäAāŠŇèġççäAJSONæTṛæ■õāĈcä;NāēĆiiijŽ

```
# Writing JSON data
with open('data.json', 'w') as f:
    json.dump(data, f)

# Reading data back
with open('data.json', 'r') as f:
    data = json.load(f)
```

## èõĪèõž

JSONçijÛçäAæTṛæNĀçŽDā\$zæIJnæTṛæ■õçšzādNäyž None iiijN bool iiijN int iiijN  
 float āŠŇ str iiijN äzēāRĹāNĒāRñēŁZāžZçšzādNæTṛæ■õçŽDlistsiiijNtuplesāŠŇdictionariesāĈ  
 āržāžŌdictionariesiiijNkeysēIJĀēēAæYṛā■ÛçñäyšçšzādN(ā■ŮāĒyāy■āzžā;TēĪdā■ÛçñäyšçšzādNçŽDkeyāĪ  
 äyžāžEēAṭā;IJSONèġDēNĆiiijNä;ääžTèrēāRĪçijÛçäAPythonçŽDlistsāŠŇdictionariesāĈ  
 èĂŇäyTṛiiijNāĪIwebāžTçTĪçĪNāžRäy■iiijNēāūāsCāržēšæēćñçijÛçäAäyžäyÄäyĹā■ŮāĒyæYṛäyÄäyĹæāGāĠEāA

JSONçijÛçäAçŽDæäijäijRāržāžŌPythonèr■æšTēĂŇāūsāGāāžŌæYṛāōNāĒĪäyÄæāũçŽDiiijNéŽd'āžEäyÄ  
 æṛTāēĆiiijNTrueäijŽēcñæYāārDäyžtrueiiijNFalseēćñæYāārDäyž-  
 falseiiijNèĂŇNoneäijŽēcñæYāārDäyžnullāĈ äyÑéÍcæYṛäyÄäyĹä;Nā■RiiijNæijTçd'zæEçijÛçäAāRŌçŽDā■

```
>>> json.dumps(False)
'false'
>>> d = {'a': True,
...      'b': 'Hello',
...      'c': None}
>>> json.dumps(d)
'{"b": "Hello", "c": null, "a": true}'
>>>
```

æĈCædIJä;æērTçĪĀāŌzæĈĀæšēJSONèġççäAāRŌçŽDæTṛæ■õiiijNä;æēĂŽäyā;ĹēŽ;ēĂŽēŁĠçōĀā■TçŽĪ  
 çĹLžāĹnæYṛā;ŠæTṛæ■õçŽDā;NāēÛçzŠædDāsĆæñā;ĹæūsæĹŮēĂĒāNĒāRñād'ġēĠRçŽDā■ŮæōṭæŮūāĈ  
 äyžāžEēġçāEšēŁZäyĹēŮōēćYiiijNāRfäzēēĂĈēŽSā;ŁçTĪpprintēĪāĪŮçŽD

pprint()                      åGjæTřæİěäzçæŽŁæŽóéĂŽçŽĎ                      print()                      åGjæTřăĂĆ  
 åóČäijŽæŇL'çĚgkeyçŽĎă■Ůæř■éąžăžŘăžúăžěăŸĂçğ■æŽt'ăŁăçŁŮëğĆçŽĎæŮžăijRèŁŞăĜžăĂĆ  
 äŸŇéİçæŸřăŸĂäŸİæijTčd'žăçČăTăijČăžŏçŽĎæLŞă■řèŁŞăĜžTwitterăŸŁæŘİJçt'ćçžŞăđİJçŽĎăŁŇă■ŘİijŽ

```
>>> from urllib.request import urlopen
>>> import json
>>> u = urlopen('http://search.twitter.com/search.json?q=python&
↳ rpp=5')
>>> resp = json.loads(u.read().decode('utf-8'))
>>> from pprint import pprint
>>> pprint(resp)
{'completed_in': 0.074,
 'max_id': 264043230692245504,
 'max_id_str': '264043230692245504',
 'next_page': '?page=2&max_id=264043230692245504&q=python&rpp=5',
 'page': 1,
 'query': 'python',
 'refresh_url': '?since_id=264043230692245504&q=python',
 'results': [{'created_at': 'Thu, 01 Nov 2012 16:36:26 +0000',
               'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:14 +0000',
               'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:13 +0000',
               'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:07 +0000',
               'from_user': ...
             },
             {'created_at': 'Thu, 01 Nov 2012 16:36:04 +0000',
               'from_user': ...
             }],
 'results_per_page': 5,
 'since_id': 0,
 'since_id_str': '0'}
>>>
```

äŸĂèĹŇæİèèőšijŇJSONèğççăĂäijŽæăžæ■őæŘŘăŁŽçŽĎæTřæ■őăĹŽăžždictsæĹŮlistsăĂĆ  
 æçČăđİJăĵăæČşèçĂăĹŽăžžăĚŮăžŮčşžăđŇçŽĎăržèşăijŇŇăŘřăžèçžŽ                      json.  
 loads()                      äijăéĂşobject\_pairs\_hookæĹŮobject\_hookăŘĆæTřăĂĆ  
 äŁŇăçČİijŇăŸŇéİçæŸřăŸİæijTčd'žăçČăTăğççăĂJSONæTřæ■őăžŮăİJăŸĂäŸİOrderedDictăŸ■ăĹİçTŽăĚŮéąžăžŘ

```
>>> s = '{"name": "ACME", "shares": 50, "price": 490.1}'
>>> from collections import OrderedDict
>>> data = json.loads(s, object_pairs_hook=OrderedDict)
>>> data
OrderedDict([('name', 'ACME'), ('shares', 50), ('price', 490.1)])
>>>
```

äŸŇéİçæŸřăçČăTăřĚăŸĂäŸİJSONă■ŮăĚŸèĵŇæ■căŸžăŸĂäŸİPythonăržèşăăŁŇă■ŘİijŽ

```
>>> class JSONObject:
...     def __init__(self, d):
...         self.__dict__ = d
...
>>>
>>> data = json.loads(s, object_hook=JSONObject)
>>> data.name
'ACME'
>>> data.shares
50
>>> data.price
490.1
>>>
```

æIJĀāRŌäyĀäyĭä;Nā■Räy■iijNJSONègççāAāRŌçŽDā■ŪāĒyā;IJäyžäyĀäyĭā■TäyĭāRCæTŗaijæĀŠçžŽ  
 \_\_init\_\_() āĀĆ çDūāRŌiijNā;āārsāRřāžēēŽRāĤČæL'ĀæñšçŽDā;ĤçTĭāōČāžEŗijNæřTāēČā;IJäyžäyĀäyĭā  
 āIJĭcijŪčāAJSONçŽDæŪūāĀŽiijNēĤYæIJL'äyĀäžŽēĀL'ēāžā;ĤæIJLçTĭāĀĆ  
 āēČādIJā;āæČšēŌūā;ŪæijČāžōçŽDæāijāijRāNŪā■ŪçņēäyšāRŌē;ŠāGžiiijNāRřāžēā;ĤçTĭ  
 json.dumps() çŽDindentāRCæTŗāĀĆ āōČāijŽā;Ĥā;Ūē;ŠāGžāŠNpprint()āG;æTŗæTĭLæđIJçšžāijijāĀĆæřT

```
>>> print(json.dumps(data))
{"price": 542.23, "name": "ACME", "shares": 100}
>>> print(json.dumps(data, indent=4))
{
    "price": 542.23,
    "name": "ACME",
    "shares": 100
}
>>>
```

āržēšāōdā;NēĀŽāyŷāžūāy■æYřJSONāRřāžRāĤŪāNŪçŽDāĀĆā;NāēČiijŽ

```
>>> class Point:
...     def __init__(self, x, y):
...         self.x = x
...         self.y = y
...
>>> p = Point(2, 3)
>>> json.dumps(p)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    File "/usr/local/lib/python3.3/json/__init__.py", line 226, in _
    ↪dumps
        return _default_encoder.encode(obj)
    File "/usr/local/lib/python3.3/json/encoder.py", line 187, in _
    ↪encode
        chunks = self.iterencode(o, _one_shot=True)
    File "/usr/local/lib/python3.3/json/encoder.py", line 245, in _
    ↪iterencode
        return _iterencode(o, 0)
```

```

File "/usr/local/lib/python3.3/json/encoder.py", line 169, in _
    ↪default
        raise TypeError(repr(o) + " is not JSON serializable")
TypeError: <__main__.Point object at 0x1006f2650> is not JSON_
    ↪serializable
>>>

```

æĈæđIä;ăæĈşăŹŔăĹŮăŇŮăŕŹèşăăôđă;ŊiijŃă;ăăŔŕăžèæŔŔă;ŽăyĂăyĹăĜ;æŢŕiijŃăôĈĉŽĐè;ŞăĖëæŸŕ

```

def serialize_instance(obj):
    d = { '__classname__' : type(obj).__name__ }
    d.update(vars(obj))
    return d

```

æĈæđIä;ăæĈşăŔ■ēĜăĬēèŎŭăŔŮēĚăyĹăôđă;ŊiijŃăŔŕăžèèĚăăŭăAžiiž

```

# Dictionary mapping names to known classes
classes = {
    'Point' : Point
}

def unserialize_object(d):
    clsname = d.pop('__classname__', None)
    if clsname:
        cls = classes[clsname]
        obj = cls.__new__(cls) # Make instance without calling __
    ↪init__
        for key, value in d.items():
            setattr(obj, key, value)
        return obj
    else:
        return d

```

ăyŇéĬæŸŕăēĈă;Ţă;ĚĉŢĬēĚăžŽăĜ;æŢŕĉŽĐă;Ŋă■Ŕiijž

```

>>> p = Point(2,3)
>>> s = json.dumps(p, default=serialize_instance)
>>> s
'{"__classname__": "Point", "y": 3, "x": 2}'
>>> a = json.loads(s, object_hook=unserialize_object)
>>> a
<__main__.Point object at 0x1017577d0>
>>> a.x
2
>>> a.y
3
>>>

```

json æĹăăĬŮēĚŸæĬĹă;Ĺăđ'ŽăĖŭăžŮēĂĹ'éăžæĬæŎĝăĹŭæŽŦă;ŎĉžĝăĹŋĉŽĐæŢŕă■ŮăĂĂĉĹ'žæôĹăĂŕŕăžèăŔĈèĂĈăôŸæŮžæŮĜăăĉèŎŭăŔŮæŽŦăđ'ŽĉzĖèĹĈăĂĈ

Vasudev Ram: Wakari, Scientific Python **in** the cloud  
Sun, 18 Nov 2012 20:19:41 +0000  
<http://jugad2.blogspot.com/2012/11/wakari-scientific-python-in-cloud.html>

Jesse Jiryu Davis: Toro: synchronization primitives **for** Tornado\_  
→coroutines  
Sun, 18 Nov 2012 20:17:49 +0000  
[http://feedproxy.google.com/~r/EmptysquarePython/~3/\\_DOZT2Kd0hQ/](http://feedproxy.google.com/~r/EmptysquarePython/~3/_DOZT2Kd0hQ/)

åĲŁæŸĲçDũĲĲjNăeĆæđĲăĲæČşăAŽèŁZăyĂæ■ēçŽĐăđ'ĐçŘĒĲĲjNăĲăēĲĲĂēēAæŽŁæ■ć  
print () èř■ăŘēæĲēăđNăĲŘăĒŸăžŮæĲĲ'ēŮčçŽĐăžNăĂĆ

## ëóĲëőž

ăĲĲăĲŁăđ'ŽăžŤçŤĲĲĲNăžŘăy■ăđ'ĐçŘĒXMLçĲĲŮčăAæăĲĲĲĲĲçŽĐăŤřæ■óæŸřăĲŁăyŸēēĲçŽĐăĂĆ  
ăy■ăžĒăŽăăyžXMLăĲĲĲInternetăyĲēĲăŮşçžŘēćăăžŁæşŽăžŤçŤĲăžŮăŤřæ■ăăžđ'æ■ćĲĲĲ  
ăŘNăŮŮăđŮČăžşæŸřăyĂçğ■ă■ŸăĆĲăžŤçŤĲĲĲNăžŘăŤřæ■óçŽĐăyŸçŤĲăăĲĲĲĲĲ(æřŤăēĆă■Ůăđ'ĐçŘĒĲĲjNăēşşă  
æŮēăyNăĲēçŽĐēőĲëőžăĲĲăĒĲăĲăĲăđŽēřžēĂēăŮşçžŘăřžXMLăşžçăĂăřŤēĲçĲçĒşæĲĲ'ăžĒăĂĆ

ăĲĲăĲŁăđ'ŽăČĒăĒĲăyNĲĲjNăĲşăĲçŤĲXMLăĲēăžĒăžĒă■ŸăĆĲăŤřæ■óçŽĐăŮŮăĂŽĲĲjNăřžăžŤçŽĐăŮĞ  
ăĲNăēĆĲĲjNăyĲēĲăĲNă■Řăy■çŽĐRSSēőćēŸĒăžŘçşžăĲĲĲăžŮăyNăĲççŽĐăăĲĲĲĲĲĲĲ

```
<?xml version="1.0"?>
<rss version="2.0" xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel>
    <title>Planet Python</title>
    <link>http://planet.python.org/</link>
    <language>en</language>
    <description>Planet Python - http://planet.python.org/</
→description>
    <item>
      <title>Steve Holden: Python for Data Analysis</title>
      <guid>http://holdenweb.blogspot.com/...-data-analysis.
→html</guid>
      <link>http://holdenweb.blogspot.com/...-data-analysis.
→html</link>
      <description>...</description>
      <pubDate>Mon, 19 Nov 2012 02:13:51 +0000</pubDate>
    </item>
    <item>
      <title>Vasudev Ram: The Python Data model (for v2 and_
→v3)</title>
      <guid>http://jugad2.blogspot.com/...-data-model.html</
→guid>
      <link>http://jugad2.blogspot.com/...-data-model.html</
→link>
      <description>...</description>
      <pubDate>Sun, 18 Nov 2012 22:06:47 +0000</pubDate>
    </item>
    <item>
      <title>Python Diary: Been playing around with Object_
→Databases</title>
```



```

        <guid>http://www.pythondiary.com/...-object-databases.
</html>
        <link>http://www.pythondiary.com/...-object-databases.
</html>
        <description>...</description>
        <pubDate>Sun, 18 Nov 2012 20:40:29 +0000</pubDate>
    </item>
    ...
</channel>
</rss>

```

```

xml.etree.ElementTree.parse()
doc.find('channel/item')
doc.findtext('title')

```

```

doc.iterfind('channel/item')
doc.find('channel/item')
doc.find('title')

```

```

ElementTree
tag
get()

```

```

>>> doc
<xml.etree.ElementTree.ElementTree object at 0x101339510>
>>> e = doc.find('channel/title')
>>> e
<Element 'title' at 0x10135b310>
>>> e.tag
'title'
>>> e.text
'Planet Python'
>>> e.get('some_attribute')
>>>

```

```

xml.etree.ElementTree
from lxml.etree import
parse

```

## 8.4 6.4 áćđéĠŖáijŖèġčæđŖād'ġādŊXMLæŮĠzū

### éŮóécŸ

ä;äæČšä;ŁçŤlār;āŖŕēČ;ārŚçŽĐāĒĚā■ŸāzŎäyÄäyłēūĚād'ġçŽĐXMLæŮĠæaçäy■æŖŖāŖŮæŤŕæ■ōāĀĆ

### èġčāĒşæŮzæąŁ

äzzä;ŤæŮūāĀŽāŖlèçAä;ăéAĠāŁŕăćđéĠŖáijŖçŽĐæŤŕæ■ōād'ĐçŖĒæŮūiijŊçññäyĀæŮūéŮŮ'ārśāžŤèŕéā  
äyŊéíćæŸŕäyÄäyłā;ŁçōĀā■ŤçŽĐāĠæŤŕiijŊāŖlā;ŁçŤlā;ŁārŚçŽĐāĒĚā■ŸārśēČ;áćđéĠŖáijŖçŽĐād'ĐçŖĒæ

```
from xml.etree.ElementTree import iterparse

def parse_and_remove(filename, path):
    path_parts = path.split('/')
    doc = iterparse(filename, ('start', 'end'))
    # Skip the root element
    next(doc)

    tag_stack = []
    elem_stack = []
    for event, elem in doc:
        if event == 'start':
            tag_stack.append(elem.tag)
            elem_stack.append(elem)
        elif event == 'end':
            if tag_stack == path_parts:
                yield elem
                elem_stack[-2].remove(elem)
            try:
                tag_stack.pop()
                elem_stack.pop()
            except IndexError:
                pass
```

äyžāŖæŧŊērŤēſŽāyłāĠæŤŕiijŊā;ăéIJĀēçAāĒŁæIJL'äyÄäyłād'ġādŊçŽĐXMLæŮĠzūāĀĆ  
éĀŽāyŷä;āāŖŕāzēāIJæŤſāžIJç;ŚçñŽæŁŮāĒñāĒſæŤŕæ■ōç;ŚçñŽāyŁæŁ;āŁŕēſŽæāūçŽĐæŮĠzūāĀĆ  
ä;ŊāçČiijŊā;āāŖŕāzēäyŊē;XMLæäijäijŖçŽĐēŁlāŁāāſēāſŎäyČéAſſēūŕāĪſæt'ijæŤŕæ■ōāžſāĀĆ  
āIJlāĒēſŽæIJñāžççŽĐæŮūāĀŽiijŊäyŊē;æŮĠzūāūſçzŖāŊĒāŖŭēĒēſĠ00,000ēāŊæŤŕæ■ōiijŊçijŮçāA

```
<response>
  <row>
    <row ...>
      <creation_date>2012-11-18T00:00:00</creation_date>
      <status>Completed</status>
      <completion_date>2012-11-18T00:00:00</completion_date>
      <service_request_number>12-01906549</service_request_
↪number>
      <type_of_service_request>Pot Hole in Street</type_of_
↪service_request>
```

```

        <current_activity>Final Outcome</current_activity>
        <most_recent_action>CDOT Street Cut ... Outcome</most_
→recent_action>
        <street_address>4714 S TALMAN AVE</street_address>
        <zip>60632</zip>
        <x_coordinate>1159494.68618856</x_coordinate>
        <y_coordinate>1873313.83503384</y_coordinate>
        <ward>14</ward>
        <police_district>9</police_district>
        <community_area>58</community_area>
        <latitude>41.808090232127896</latitude>
        <longitude>-87.69053684711305</longitude>
        <location latitude="41.808090232127896"
        longitude="-87.69053684711305" />
    </row>
    <row ...>
        <creation_date>2012-11-18T00:00:00</creation_date>
        <status>Completed</status>
        <completion_date>2012-11-18T00:00:00</completion_date>
        <service_request_number>12-01906695</service_request_
→number>
        <type_of_service_request>Pot Hole in Street</type_of_
→service_request>
        <current_activity>Final Outcome</current_activity>
        <most_recent_action>CDOT Street Cut ... Outcome</most_
→recent_action>
        <street_address>3510 W NORTH AVE</street_address>
        <zip>60647</zip>
        <x_coordinate>1152732.14127696</x_coordinate>
        <y_coordinate>1910409.38979075</y_coordinate>
        <ward>26</ward>
        <police_district>14</police_district>
        <community_area>23</community_area>
        <latitude>41.91002084292946</latitude>
        <longitude>-87.71435952353961</longitude>
        <location latitude="41.91002084292946"
        longitude="-87.71435952353961" />
    </row>
</row>
</response>

```

åAĞèö;ä;äæČšâEŽäyÄäyİeĐŽæIJnäİæÑL'çĖğâİŞæt'ijæLěăŚŁæTřéGRæŎŠăĽŮéCőçijŮăRŭcăAăĂĆă

```

from xml.etree.ElementTree import parse
from collections import Counter

potholes_by_zip = Counter()

doc = parse('potholes.xml')
for pothole in doc.iterfind('row/row'):

```

ɛfZäyɪɓDŽæIJñäTṛäyAçŽDëŮóécYæYřáoČäijŽăĚĹârEæTɪʔäyɪXMLæŮĠGäzúăŁăè;ǀăĹrăĚĚăYäy■çDū.  
 ăIJăĹĤŤçŽDăIJžăZlăyŁiijNăyžăžEɛfRĕaŊĕfZäyɪɓNăžRĕIJăEeAçTĭăĹr450MBăŭăRšçŽDăĚĚăYçɪʔžēŮʔă  
 ăçCădIJăɪɓçTĭăçCăyNăžčçăAiiijŊçĭNăžRăRĭĭIJăEeAăfŏăTžäyAçCžçCžiiž

čzSædIJaYřriižZèfZävylçL'LæIJñçŽDžzččăAæfŘèaŇæUúáRléIJAěeA7MBçŽDăEĚă■Ÿ-ăd'ğăd'ğèLĆçze

```

    efZāyÄēŁĈŽDæLǻIjřaijŽä;İetŰ ElementTree æłaaİUäy■çŽDäyd'äylæäyafČăŁšèĈ;āĂĆ
    çñnäyĀiiJNiterparse() æŰzæsTāĒAēōyārZXMLæŰĞæaçefZēaŊăcdéGRæŞ■ăIJāĂĆ
    ä;fçTlæŰüriijNă;ăeIJĀēęAæRRă;ŽæŰĞazūāŘ■ăŠŇăyĀäyłaŊĒăŘnăyNeİcâyĂçğ■æLŰûd'Žçğ■çşzādŊçŽDă
    start      , end, start-ns   āŠŇ     end-ns   āĂĆ   çTs   iterparse()
    āĹZāzzçŽDēf■ăzcăZlăijŽăzgçTšă;caēĆ   (event, elem)   çŽDăĒÇçŽDiiJŊ   āĔüäy■
    event æYřayŁēfřazŊăzūāLŰēaläy■çŽDăşRăyĀäyhijŊēĀŇ   elem æYřçŽyâzTçŽDXM-
    LăĒĈct'ăăĂĆă;ŊăēĆiiJŽ

```

start äẏNäẏuäIJläšRäẏläĖĈct'äçññäẏÄæñäècñäLZäẏzäẏüäYtēŸäšsæIJL'ècñäRŠšäĖäĖüäzÜæTṛæ■  
 èÄN end äẏNäẏuäIJläšRäẏläĖĈct'äaũšcZṚäoNäLṚæÜüècñäLZäẏzäÄĈ

är;çøæşæIJL'åIJlä;Nå■Räy■æijTçd'ziiN start-ns åŠN end-ns  
äzNäzûècñçTlæIæäd' DçRXMLæŮGæaçåS;åR■çl'zéŮt'çZDäçræYŌãĂĆ

èfZæIJnèLÇä;Nå■Räy■iiN start åŠN end äzNäzûècñçTlæIèçøaçRÊäĖČçt'ääŠNæăĢç■æăLăĂĆ  
æăLăzçèălăzEæŮGæaçècñèġçædRæŮŮçZDăŖCæñaçZŞædDiiN  
èfYèçñçTlæIæăLd'æŮ■æşRäylăĖČçt'ăæYřăRçăNzéĖ■ăijăçzZăĢ;æTř  
parse\_and\_remove() çZDèŮřă;DăĂĆ âçCædIJăNzéĖ■iiNăřsăLl'çTl yield  
èr■ăRēăRŠērČçTlèĂĖèfTăZdèfZăylăĖČçt'ăăĂĆ

åIJl yield äzNăRŌçZDăyNéIcèfZăylēr■ăRēæL■æYřă;Ĥă;ŮçlNăzRă■ăçTlædAăřSăĖĖă■YçZDElement

```
elem_stack[-2].remove(elem)
```

èfZăylēr■ăRēă;Ĥă;ŮçlNăL■çTs yield äzġçTşçZDăĖČçt'ăăzŌăđČçZDçLŮèLÇçCzăy■ăLăéZd'æŌLă  
ăAĢèç;ăŮşçZRæşæIJL'ăĖŮăđČçZDăIJræŮzăijTçTlèfZăylăĖČçt'ăăzEiiNéCçăzLèfZăylăĖČçt'ăăřsècñéTĂæ

ărzéLÇçCzçZDèf■ăzçăijRèġçædRăŠNăLăéZd'çZDăIJăçzLæTlædIJăřsæYřăyĂăylăIJlæŮGæaçăylénY  
æŮGæaçæăŞçZŞædDăzŌăġNèĢçzLæşæcñăđNæTt'çZDăLZăzžèfĢăĂĆăr;çøăæČæ■d'iiNèfYæYřèČ;éĂZ

èfZçġ■æŮzæăLçZDăyždèAçijzéZŮăřsæYřăđČçZDèfRèăNăĂġèČ;ăžEăĂĆ  
æLŠèĢăŮsætNèřTçZDçzŞædIJæYřiiNèřzăRŮæTt'ăylæŮGæaçăLřăĖĖă■Yăy■çZDçL'LæIJñçZDèfRèăNăĂş  
ă;EăYřăđČă■r'ă;ĤçTlăžEèŮĖèfĢăRŌèĂĖ60ăĂ■çZDăĖĖă■YăĂĆ  
ăZăæ■d'iiNăçCædIJă;ăæZt'ăĖşăĤčăĖĖă■Yă;ĤçTlèĢRçZDèřiiNéCçăzLăcđéĢRăijRçZDçL'LæIJăđNèČIJ

## 8.5 6.5 årĖă■ŮăĖYè;ñæ■căyžXML

### éŮóécY

ă;ăæČşă;ĤçTlăyĂăylPythonă■ŮăĖYă■YăĆlæTřæ■ōiiNăzŮăřĖăđČç;ñæ■cæLřXMLæăijăijRăĂĆ

### èġçăĖşæŮzæăL

ăr;çøă xml.etree.ElementTree âzŞéĂZăyçTlæIèăĂZèġçædRăŮèă;IJiiNăĖŮăđăđăČăzşăRřăžèăL  
ă;NăĖČiiNèĂČèZŞæCăyNèfZăylăĢ;æTřiiN

```
from xml.etree.ElementTree import Element

def dict_to_xml(tag, d):
    '''
    Turn a simple dict of key/value pairs into XML
    '''
    elem = Element(tag)
    for key, val in d.items():
        child = Element(key)
        child.text = str(val)
        elem.append(child)
    return elem
```

ăyNéIcæYřăyĂăylă;ĤçTlă;Nă■RiiZ



```
>>> # Proper XML creation
>>> e = dict_to_xml('item', d)
>>> tostring(e)
b'<item><name>&lt; spam&gt;</name></item>'
>>>
```

æşlæĐRāLřçlNāzRçŽĐāŘŎéícéĆčäyġāNā■Řäy■rijNā■Ůčņę āĀŸ<āĀŽ āŠŇ āĀŸ>āĀŽ  
èćnāŽŁæ■ćæĹŘäžĒ &lt; āŠŇ &gt;

äyNéIćäzĒäġZāRĆčĒĀčrijNāęĆädIJā;ăéIJĀëęAæLŇāLlāŎžè;ñæ■ćēŁZāžZā■ŮčņęrijN  
āRřäžēä;ŁçTl xml.sax.saxutils äy■čŽĐ escape() āŠŇ unescape()  
āĢ;æŦřāĀĆäġNāęĆrijŽ

```
>>> from xml.sax.saxutils import escape, unescape
>>> escape('<spam>')
'&lt; spam&gt;'
>>> unescape(_)
'<spam>'
>>>
```

éŽd'āžĒęĊ;āĹZāžzæ■ćçāŏčŽĐē;ŠāGžād'ŮrijNēŁŸæIJLāRęād'ŮäyĀäyġāŎšāZāæŎġē■Řä;āāĹZāžž  
Element āŏđäġNēĀNäy■æŸřā■ŮčņęäyşrijN éĆčāřsæŸřä;ŁçTlā■ŮčņęäyşçzĐāŘĹæđĐéĀäyĀäyġæŽt'ād'ğç  
ēĀN Element āŏđäġNāRřäžēäy■čTlēĀĆēŽSēğčæđRXMLæŮĢæIJŇčŽĐæĀĒāĒäyNēĀŽēŁĢād'Žçğ■æŮžā  
āžšāřsæŸřēřt'rijNā;āāRřäžēāIJāyĀäyġēŇŸčžgæŦřæ■ŏčzŠæđĐäyĹāŏNāĹRā;ææLĀæIJLčŽĐæŠ■āIJrijNāžŮ

## 8.6 6.6 èğčæđŘāŠŇNäŁŏæŦžXML

### éŮŏéćŸ

äġäæČşērźāRŮäyĀäyġXMLæŮĢæäčrijNāřźāŏĆæIJĀäyĀäžZāŁŏæŦžrijNçĐŮāRŎāřĒçzŠæđIJāĒZāždXM

### èğčāĒşæŮžæāĹ

äġŁçTl xml.etree.ElementTree æġāāĪŮāRřäžēäġĹāŏžæŸŞçŽĐād'ĐçŘĒēŁZāžZāžzāĹāĀĆ  
çŋŋäyĀæ■æŸřäžēēĀŽäyŸčŽĐæŮžāijRæĪēèğčæđŘēŁZäyġæŮĢæäčāĀĆäġNāęĆrijNāĀĢēŏġäġæIJL'äyĀäyġā  
pred.xml çŽĐæŮĢæäčrijNçşzāijijäyNéIćēŁZæāŮrijŽ

```
<?xml version="1.0"?>
<stop>
  <id>14791</id>
  <nm>Clark &amp; Balmoral</nm>
  <sri>
    <rt>22</rt>
    <d>North Bound</d>
    <dd>North Bound</dd>
  </sri>
  <cr>22</cr>
```

```

<pre>
  <pt>5 MIN</pt>
  <fd>Howard</fd>
  <v>1378</v>
  <rn>22</rn>
</pre>
<pre>
  <pt>15 MIN</pt>
  <fd>Howard</fd>
  <v>1867</v>
  <rn>22</rn>
</pre>
</stop>

```

äyÑéÍæÝřäyÄäyİäLİ'çTİ ElementTree æİëèrZâRŮëfŽäyİæŮĞæaçâzûârZâoČâAŽäyÄäzZæŁoæTžçŽ

```

>>> from xml.etree.ElementTree import parse, Element
>>> doc = parse('pred.xml')
>>> root = doc.getroot()
>>> root
<Element 'stop' at 0x100770cb0>

>>> # Remove a few elements
>>> root.remove(root.find('sri'))
>>> root.remove(root.find('cr'))
>>> # Insert a new element after <nm>...</nm>
>>> root.getchildren().index(root.find('nm'))
1
>>> e = Element('spam')
>>> e.text = 'This is a test'
>>> root.insert(2, e)

>>> # Write back to a file
>>> doc.write('newpred.xml', xml_declaration=True)
>>>

```

âd'DçŘEçzŞædİJæÝřäyÄäyİäČŘäyÑéÍèfŽæäüæŮřçŽĐXMLæŮĞäzûİijŽ

```

<?xml version='1.0' encoding='us-ascii'?>
<stop>
  <id>14791</id>
  <nm>Clark &amp; Balmoral</nm>
  <spam>This is a test</spam>
  <pre>
    <pt>5 MIN</pt>
    <fd>Howard</fd>
    <v>1378</v>
    <rn>22</rn>
  </pre>
  <pre>
    <pt>15 MIN</pt>

```



```
<fd>Howard</fd>
<v>1867</v>
<rn>22</rn>
</pre>
</stop>
```

## èóìèõž

ä£ôæŤžäyÄäyİXMLæŰĞæąççzŞæđĐæŸřăĹăőžæŸŞçŽĐiijŇăĬEæŸřăĬăă£ĚéązçĹ'cèõřçŽĐæŸřăĹ'ĂæĬ  
årĚăőČăĬJăyžžäyÄäyĹăĹŰëăĹæĬëăđ'ĐçŘĚăĂČăĬŇăęCiiJŇăęĆăđIJăĬăăĹăéŽđ'æŞŘăyĹăĚČçť'ăiijŇéĂŽè£ĜërČ  
remove() æŰžæşŤăžŎăőČçŽĐçŽť'æŎëçĹüèĹČçČžăy■ăĹăéŽđ'ăĂČ  
ăęĆăđIJăĬăæŘŠăĚëæĹŰăćđăĹăăŰřçŽĐăĚČçť'ăiijŇăĬăăŘŇăăüăĬ;£çŤĬçĹüèĹČçČžăĚČçť'ăçŽĐ  
insert()ăŠŇappend()æŰžæşŤăĂČè£ŸëČĬ;ăržăĚČçť'ăăĬ;£çŤĬçť'ćăijŤăŠŇăĹĜçĹĜăŞ■ăĬJiijŇăřŤăęĆ  
element[i]æĹŰelement[i:j]

ăęĆăđIJăĬăęĬJĂëęĂăĹŽăžžæŰřçŽĐăĚČçť'ăiijŇăŘřăžëăĬ;£çŤĬæĬJŇèĹĆăŰžæăĹăy■ăijŤçđ'žçŽĐ  
Element çşžăĂČăĹŚăžŇăĬJĬ6.5ărŘëĹĆăüşçžŘëřęçžĚëóìèõžè£ĜăžĚăĂČ

## 8.7 6.7 ăĹ'çŤĬăŚĬăŘ■çĹ'žéŰť'èğçæđŘXMLæŰĞæąç

### éŰöécŸ

ăĬăăČşèğçæđŘăşŘăyİXMLæŰĞæąçiijŇăŰĞæąçăy■ăĬ;£çŤĬăžĚXMLăŚĬăŘ■çĹ'žéŰť'ăĂČ

### èğçĂĚşæŰžæăĹ

ëĂČëŽŚăyŇéĬcè£ŽăyĹăĬ;£çŤĬăžĚăŚĬăŘ■çĹ'žéŰť'çŽĐæŰĞæąçiijŽ

```
<?xml version="1.0" encoding="utf-8"?>
<top>
  <author>David Beazley</author>
  <content>
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <title>Hello World</title>
      </head>
      <body>
        <h1>Hello World!</h1>
      </body>
    </html>
  </content>
</top>
```

ăęĆăđIJăĬăèğçæđŘë£ŽăyĹăŰĞæąçăžüăĹĜëăŇăŽöéĂŽçŽĐăşëëřçiiJŇăĬăăiijŽăŘŚçŎřë£ŽăyĹăžüăy■ăŸ

```

>>> # Some queries that work
>>> doc.findtext('author')
'David Beazley'
>>> doc.find('content')
<Element 'content' at 0x100776ec0>
>>> # A query involving a namespace (doesn't work)
>>> doc.find('content/html')
>>> # Works if fully qualified
>>> doc.find('content/{http://www.w3.org/1999/xhtml}html')
<Element '{http://www.w3.org/1999/xhtml}html' at 0x1007767e0>
>>> # Doesn't work
>>> doc.findtext('content/{http://www.w3.org/1999/xhtml}html/head/
↳title')
>>> # Fully qualified
>>> doc.findtext('content/{http://www.w3.org/1999/xhtml}html/'
... '{http://www.w3.org/1999/xhtml}head/{http://www.w3.org/1999/
↳xhtml}title')
'Hello World'
>>>

```

ä;äâRfrazéeĂŽèfĠärEâŚ;âR■çl'žéŮt'âd'DçŘEéĂžè;ŚâŇĚèċĚäyžäyĂäylâũëăĚũçszælēćóĂăŇŮëfZäyłè

```

class XMLNamespaces:
    def __init__(self, **kwargs):
        self.namespaces = {}
        for name, uri in kwargs.items():
            self.register(name, uri)
    def register(self, name, uri):
        self.namespaces[name] = '{'+uri+'}'
    def __call__(self, path):
        return path.format_map(self.namespaces)

```

éĂŽèfĠäyŇéIćçŽDæŮžâijRä;fçTlèfZäyłçsziiž

```

>>> ns = XMLNamespaces(html='http://www.w3.org/1999/xhtml')
>>> doc.find(ns('content/{html}html'))
<Element '{http://www.w3.org/1999/xhtml}html' at 0x1007767e0>
>>> doc.findtext(ns('content/{html}html/{html}head/{html}title'))
'Hello World'
>>>

```

## ëőlëőž

èğçædŘăRñæIJL'âŚ;âR■çl'žéŮt'çŽDXMLæŮĠæaçäijŽæfTè;ČçzAçŘRăĂĆ äyLéIćçŽD  
XMLNamespaces äzĚäžĚæYřăĚAèöyä;ää;fçTlçijl'çTěâR■äzçæŽŁăōŇæTt'çŽDURIârEăĚũăRŸă;Ůçí■ă;öç  
ă;Łäy■äzççŽDæYřijŇăIJlâşžæIJñçŽD ElementTree  
èğçædŘäy■æşæIJL'äzzä;TéĀTă;DèŮăRŮăŚ;âR■çl'žéŮt'çŽDăfæAřăĂĆ  
ä;EæYřijŇăçĈædIJä;ää;fçTl'iterparse() âĠ;æTřçŽDèlârśâRfrazéeŮũăRŮæŽt'âd'ŽăĚşăžŌăŚ;âR■çl'žé

```
>>> from xml.etree.ElementTree import iterparse
>>> for evt, elem in iterparse('ns2.xml', ('end', 'start-ns', 'end-
↳ns')):
...     print(evt, elem)
...
end <Element 'author' at 0x10110de10>
start-ns ('', 'http://www.w3.org/1999/xhtml')
end <Element '{http://www.w3.org/1999/xhtml}title' at 0x1011131b0>
end <Element '{http://www.w3.org/1999/xhtml}head' at 0x1011130a8>
end <Element '{http://www.w3.org/1999/xhtml}h1' at 0x101113310>
end <Element '{http://www.w3.org/1999/xhtml}body' at 0x101113260>
end <Element '{http://www.w3.org/1999/xhtml}html' at 0x10110df70>
end-ns None
end <Element 'content' at 0x10110de68>
end <Element 'top' at 0x10110dd60>
>>> elem # This is the topmost element
<Element 'top' at 0x10110dd60>
>>>
```

æIJĀāRŌäyĀçĆzījNāēĆæđIJä;àèAād'DçŘEçŽĐXMLæŮĜæIJñéŽd'āžEèeAä;£çŤlāLřāĚúāzŮénŸçžg  
 āžžèōōä;ăæIJĀāē;æŸřä;£çŤl lxml āĜ;æŤřāž\$æĪēāzčæŽ£ ElementTree āĀĆ  
 ä;NāēĆījNlxml āřžāLl'çŤlDTDēlNērAæŮĜæāčāĀAæZl'āē;çŽĐXPathæŤřæNĀāšNāyĀāžZāĚūāzŮénŸçžg  
 è£ŽāyĀārRēŁCāĚūāōđāRlæŸřæŤŽā;āāēĆä;Ťèol'XMLèğçæđRçl■ā;ōçōĀā■ŤäyĀçĆzāĀĆ

## 8.8 6.8 äyŌāĚŞçşzādNæŤřæ■ōāžŞçŽĐāžd'āžŠ

### éŮóécŸ

ä;ăæČşāIJlāĚşçşzādNæŤřæ■ōāžŞäy■æşèèrcāĀAācdāLāæLŮāLāéŽd'èōrā;ŤāĀĆ

### èğçāEşşæŮzæāL

Pythonäy■ēāłçd'žād'ŽèāNæŤřæ■ōçŽĐæāĜāĜEæŮžāijRæŸřäyĀäyłçŤsāĚĆçžĐæđĐæLŖçŽĐāžRāLŮāĀ

```
stocks = [
    ('GOOG', 100, 490.1),
    ('AAPL', 50, 545.75),
    ('FB', 150, 7.45),
    ('HPQ', 75, 33.2),
]
```

ä;Īæ■ōPEP249ījNēĀŽè£Ĝè£Žçğ■ā;ćāijRæŘŘä;ŽæŤřæ■ōījN  
 āŖřāžēā;ĹāōzæŸŞçŽĐä;£çŤlPythonæāĜāĜEæŤřæ■ōāžŞAPIāšNāĚşçşzādNæŤřæ■ōāžŞè£ŽèāNāžd'āžŠāĀĆ  
 æLĀæIJL'æŤřæ■ōāžŞäyŁçŽĐæŞ■ä;IJéČ;éĀŽè£ĜSQLæşèèrcēr■āRēæĪēāōNæLŖāĀĆæŖRāyĀèāNè;ŞāĚèè;

äyžāžEæijŤçd'žèŖ'æŸŌījNä;āāŖřāžēä;£çŤlPythonæāĜāĜEāžŞäy■çŽĐ sqlite3  
 æłāāĪŮāĀĆ æēĆæđIJä;ää;£çŤlçŽĐæŸřäyĀäyłäy■āŖNçŽĐæŤřæ■ōāžŞ(æŖŤæĆMySqlāĀPostgresqlæLŮèĀ

ēŁŸāĹŮāŁēĈĖŻŸāžŤĉŽĎĉňňäŸL'æŮžæĹāāĹŮæĹæŖŖäĹZæŤŖæŇAãĀĈ  
äŸ■ēŁĖĜĉŻŸāžŤĉŽĎĉĭŸŮĉĹŇæŮőăŖĈăĜăăžŮőĈĭæŸŖäŸĀæăŮĉŽĎĭĭŇēŽđ'ăžĖäŸĀĉĈžĉĈžĉzĖăĹőăŮőăĹŇăđ'Ůă  
ĉňňäŸĀæ■ēæŸŖēŁđăŮőăĹŖæŤŖæ■őăžŖăĀĈĖĀŽăŸŸăĭăēĖAæL'ĝēăŇ connect ()  
ăĜĭæŤŖĭĭŇĉžŽăőĈæŖŖäĹZăŸĀăžŽæŤŖæ■őăžŖăŖ■ăĀăŸzæĬžăĀĉŤĹæĹŮăŖ■ăĀăŖĖĉăĀăŖŇăĖŮăžŮăŁēŁ

```
>>> import sqlite3
>>> db = sqlite3.connect('database.db')
>>>
```

äŸžăžĖăđ'ĎĉŖĖæŤŖæ■őĭĭŇŇäŸŇäŸĀæ■ēăĭăēĬĀēĖAăĹZăžžăŸĀăŸĹæŸŸăăĜăĀĈ  
äŸĀæŮēăĭăăĬĹ'ăžĖăŸŸăăĜĭĭŇēĈăžĹăĭăŖŖăŖăžēæL'ĝēăŇSQLæŖēĖĉĖŖ■ăŖēăžĖăĀĈæŖŤæĈĭĭž

```
>>> c = db.cursor()
>>> c.execute('create table portfolio (symbol text, shares integer, _
↳ price real)')
<sqlite3.Cursor object at 0x10067a730>
>>> db.commit()
>>>
```

äŸžăžĖăŖŖăŖæŤŖæ■őăžŖăĹăŸ■æŖŖăĖēăđ'ŽæĹăēőŖăĭŤĭĭŇăĭĤĉŤĹĉŖăĭĭĭăŸŇēĬĖēŁZæăŮĉŽĎĖŖ■ăŖēĭĭž

```
>>> c.executemany('insert into portfolio values (?, ?, ?)', stocks)
<sqlite3.Cursor object at 0x10067a730>
>>> db.commit()
>>>
```

äŸžăžĖăĖL'ĝēăŇæŖŖăŸĹæŖēĖĉĭĭŇăĭĤĉŤĹăĈŖăŸŇēĬĖēŁZæăŮĉŽĎĖŖ■ăŖēĭĭž

```
>>> for row in db.execute('select * from portfolio'):
...     print(row)
...
('GOOG', 100, 490.1)
('AAPL', 50, 545.75)
('FB', 150, 7.45)
('HPQ', 75, 33.2)
>>>
```

ăĖĈăđĬăĭăăĈŖăŖăŖŮĉŤĹăĹŮēĹŖăĖăĭĬăŸžăŖĈæŤŖæĹæL'ĝēăŇæŖēĖĉæŖ■ăĭĬĭĭŇăŖēĖăžĉăőăĹăĭăă

```
>>> min_price = 100
>>> for row in db.execute('select * from portfolio where price >= ?
↳ ',
                           (min_price,)):
...     print(row)
...
('GOOG', 100, 490.1)
('AAPL', 50, 545.75)
>>>
```

## èõléõž

āIJārfTēĶČā;ŌčŽDčžgāLnāyLāŠNæTṛæ■ōāžŠāzd'āžŠæYřéIdāyȳčōĀā■TčŽDāĀĆ  
ā;āāRlēIJāæRŘā;ŽSQLēf■āRēāžūēřČčTlčŽyāžTčŽDāIāāIŪāřsāRřāžēæŽt'æŪræLŪæRŘāRŪæTṛæ■ōāžEāĀ  
ēŽjērt'æēCæ■d'īijNēfYæYřæIJL'āyĀāžZærTēĶČæcYæL'NčŽDčžEēLČēŪōécYēIJĀēēAā;āēĀRāyIāLŪāGžāČ

āyĀāyIēŽĶčCzæYřæTṛæ■ōāžŠāy■čŽDæTṛæ■ōāšNPythončšzādNčŽt'æŌēčŽDæYāārDāĀĆ  
āržāžŌæŪēæIJščšzādNīijNēĀŽāyāRřāžēā;ĲčTl datetime æIāāIŪāy■čŽD datetime  
āōdāĶNīijN æLŪēĀĒāRřēČjæYř time æIāāIŪāy■čŽDčšžčžšæŪēŪt'æLšāĀĆ  
āržāžŌæTṛā■ŪčšzādNīijNčL'žāLnæYřā;ĲčTlāLrāRræTṛčŽDēGšēd■æTṛæ■ōīijNāRřāžēčTl  
decimal æIāāIŪāy■čŽD Decimal āōdāĶNāIēēāIčd'žāĀĆ  
āy■āžyčŽDæYřīijNāržāžŌāy■āRŊčŽDæTṛæ■ōāžŠēĀNēIĀāĒŪā;ŠæYāārDēgDāLZæYřāy■āyĀæāūčŽDīijNā

āRēād'ŪāyĀāyIæŽt'āLāād'■æIČčŽDēŪōécYāřsæYřSQLēf■āRēā■ŪčņēāyščŽDædDēĀāāĀĆ  
ā;āā■ČāyGāy■ēēAā;ĲčTlPythonā■ŪčņēāyšæāijāijRāNŪæŠ■ā;IJčņē(āēČ%)æLŪēĀĒ  
.format() æŪžæšTæIēāLZāžžēfZæāūčŽDā■ŪčņēāyšāĀĆ  
āēČædIJāijāēĀščžZēfZāžZæāijāijRāNŪæŠ■ā;IJčņēčŽDāĀijæIēēGāžŌčTlāLŪčŽDēĶŠāĒēīijNēČčāžLā;āčŽ  
<http://xkcd.com/327> )āĀĆ æšēēřčēř■āRēāy■čŽDēĀŽēĒčņē ?  
æNĠčd'žāRŌāRræTṛæ■ōāžŠā;ĲčTlāōČēGāūščŽDā■ŪčņēāyšæZēæ■cæIJžāLŪīijNēfZæāūæŽt'āLāčŽDāōL'ā

āy■āžyčŽDæYřīijNāy■āRŊčŽDæTṛæ■ōāžŠāRŌāRrāržāžŌēĀŽēĒ■čņēčŽDā;ĲčTlæYřāy■āyĀæāūčŽDā  
? æLŪ %s īijN ēfYæIJL'āĒūāžŪāyĀāžZā;ĲčTlāžEāy■āRŊčŽDčņēāRūīijNærTāēČ:0æLŪ:1æIēæNĠčd'žāRČ  
āRŊæāūčŽDīijNā;āēfYæYřā;ŪāŌžāRČēĀČā;āā;ĲčTlčŽDæTṛæ■ōāžŠæIāāIŪčŽyāžTčŽDæŪGæāčāĀĆ  
āyĀāyIæTṛæ■ōāžŠæIāāIŪčŽD paramstyle āsdæĀgāNēāRnāžEāRČæTṛāijTčTlēcŌæāijčŽDāēqæĀřāĀĆ

āržāžŌčōĀā■TčŽDæTṛæ■ōāžŠæTṛæ■ōčŽDēřzāEŽēŪōécYīijNā;ĲčTlæTṛæ■ōāžŠAPIēĀŽāyēIdāyȳčōĀ  
āēČædIJā;āēēAād'DčRĒæŽt'āLāād'■æIČčŽDēŪōécYīijNāžžēōōā;āā;ĲčTlæŽt'āLāēnYčžgčŽDæŌēāRčīijNær  
čšžāijij SQLAlchemy ēfZæāūčŽDāžŠāĒēōyā;āā;ĲčTlPythončšzæIēēāIčd'žāyĀāyIæTṛæ■ōāžŠæāIīijN  
āžūāyTēČ;āIJlēZRēŪRāžTāsCSQLčŽDæČĒēIāyNāōdčŌrāRĎčg■æTṛæ■ōāžŠčŽDæŠ■ā;IJāĀĆ

## 8.9 6.9 çijŪčāAāŠNēgčçāAā■AāĒ■ēfZāLŪæTṛ

### éŪōécYř

ā;āæČšārEāyĀāyIā■AāĒ■ēfZāLŪā■ŪčņēāyšēgčçāAāēLŘāyĀāyIā■ŪēLČā■ŪčņēāyšæLŪēĀĒārEāyĀāyIā

### ēgčāEšæŪzæāI

āēČædIJā;āāRlæYřčōĀā■TčŽDēgčçāAāēLŪčijŪčāAāyĀāyIā■AāĒ■ēfZāLŪčŽDāŌšāgNā■ŪčņēāyšīijNā  
æIāāIŪāĀĆāĶNāēČīijŽ

```
>>> # Initial byte string
>>> s = b'hello'
>>> # Encode as hex
>>> import binascii
>>> h = binascii.b2a_hex(s)
>>> h
b'68656c6c66f'
```

```
>>> # Decode back to bytes
>>> binascii.a2b_hex(h)
b'hello'
>>>
```

čšzäijijčŽĎāŁšèČ;āŘŇæüāŘřäžēāIJĪ base64 æĹāāĪŮäy■æL;āĹřāĀĆä;ŇāēĆriiž

```
>>> import base64
>>> h = base64.b16encode(s)
>>> h
b'68656C6C6F'
>>> base64.b16decode(h)
b'hello'
>>>
```

## èőléőž

ād'gēČĹāĹEæČĚāEĵäyŇüijŇéĀŽēŁĜā;ŁçŤĹäyŁēŁřçŽĎāĜ;æŤřæĹēē;Ňæ■čā■AāĚ■ēŁZāĹūæŸřāĹŁçōĀā■Ť  
äyŁēĹčäyĎ'čĝ■æĹĀæIJřçŽĎäyžēēAäy■āŘŇāĪĹāžŌād'ĝārRāEŁZçŽĎād'ĎčŘEāĀĆ  
āĜ;æŤř base64.b16decode() āŠŇ base64.b16encode()  
āŘĹēČ;æŠ■ā;ĪĹād'ĝāEŁZā;čāijRçŽĎā■AāĚ■ēŁZāĹūā■Ůæř■üijŇ èĀŇ binascii  
æĹāāĪŮäy■čŽĎāĜ;æŤřād'ĝārRāEŁZēČ;èČ;ād'ĎčŘEāĀĆ

ēŁŸæĪĹL'äyĀçČzéĪĀēēAæšĹæĎŘçŽĎæŸřçijŮčāAāĜ;æŤřæĹ'ĀāžĝçŤšçŽĎē;ŠāĜžæĀžæŸřäyĀäyĹā■Ů  
āēČæĎĪJæČšāijžāĹūäžēUnicodeā;čāijRē;ŠāĜžriiŇā;æĪĀēēAāčĎāĹäyĀäyĹēčĹād'ŮçŽĎçŤŇēĹčæ■ēēĹĎ'āĀĆ

```
>>> h = base64.b16encode(s)
>>> print(h)
b'68656C6C6F'
>>> print(h.decode('ascii'))
68656C6C6F
>>>
```

āĪĹēĝčçāAā■AāĚ■ēŁZāĹūæŤřæŮüüijŇāĜ;æŤř b16decode()  
āŠŇ a2b\_hex() āŘřäžēæŌēāRŮā■ŮēŁĆæĹŮUnicodeā■ŮçŇēäyšāĀĆ  
ä;EæŸřriiŇUnicodeā■ŮçŇēäyšāēŁēēāžāžēĀžēĀŘĹāŇēāŘŇASCIIçijŮčāAçŽĎā■AāĚ■ēŁZāĹūæŤřāĀĆ

## 8.10 6.10 çijŮčāAēĝčçāAŁBase64æŤřæő

### éŮőéčŸ

ä;æĹĪĀēēAā;ŁçŤĪBase64æāijāijRēĝčçāAæĹŮçijŮčāAāžŇēŁZāĹūæŤřæ■őāĀĆ



```

from struct import Struct
def write_records(records, format, f):
    '''
    Write a sequence of tuples to a binary file of structures.
    '''
    record_struct = Struct(format)
    for r in records:
        f.write(record_struct.pack(*r))

# Example
if __name__ == '__main__':
    records = [ (1, 2.3, 4.5),
                 (6, 7.8, 9.0),
                 (12, 13.4, 56.7) ]
    with open('data.b', 'wb') as f:
        write_records(records, '<idd', f)

```

æIJL'â;Łâd'Žçġ■æŮzæŝŦæİëèrżâRŮëfŽäyġæŮĠäzûâzûëfŦâŽđäyÄäyġâĚČçzĐâĹŮëāġâĈ  
 éęŮâĚĹijŊăęĈăđIJă;ăæL'ŞçóŮăzēăİŮçŽĐă;ćăijŖăcđéĠŖèrżâRŮæŮĠäzûijŊă;ăăŖăzēëfŽæăûăAŽiijŽ

```

from struct import Struct

def read_records(format, f):
    record_struct = Struct(format)
    chunks = iter(lambda: f.read(record_struct.size), b'')
    return (record_struct.unpack(chunk) for chunk in chunks)

# Example
if __name__ == '__main__':
    with open('data.b', 'rb') as f:
        for rec in read_records('<idd', f):
            # Process rec
        ...

```

ăęĈăđIJă;ăæĈşârĖæŦŦ'âyġæŮĠäzûäyÄæŋăæĂğèrżâRŮăĹrăyÄäyġâ■ŮëĹĈă■Ůçŋâyşây■ijŊçĐŮăŖŎăĹ

```

from struct import Struct

def unpack_records(format, data):
    record_struct = Struct(format)
    return (record_struct.unpack_from(data, offset)
            for offset in range(0, len(data), record_struct.size))

# Example
if __name__ == '__main__':
    with open('data.b', 'rb') as f:
        data = f.read()
    for rec in unpack_records('<idd', data):
        # Process rec
    ...

```



äyd' çg■æČĚĀĒĭäyŇçŽĎçzŞæđIJéČ;æŸřäyÄäyĭāŔŕèĤĭāŽđçŤĭæĭēāĹZāzžèŕēæŮĠāzŭçŽĎāŎŞāgŇāĚČçz

## èõĭèõž

ārżāžŎēIJĀèĕAçijŮčāAāŠŇèġççāAāžŇèĤZāĹŭæŤŕæ■ōçŽĎçĭŇāžŔèĀŇēĭĀrijŇēĀŽāyŷaijŽā;ĤçŤĭ  
struct æĭāāĭŮāĀČ äyžāžĒāčŕæŸŎäyÄäyĭæŮŕçŽĎçzŞæđDä;ŞĭijŇāŔĭēIJĀèĕAāČŔèĤZæāŭāĹZāzžäyÄäyĭ  
Struct āōđäĭŇā■şāŔŕijŽ

```
# Little endian 32-bit integer, two double precision floats
record_struct = Struct('<idd')
```

çzŞæđDä;ŞēĀŽāyŷaijŽā;ĤçŤĭäyĀāžZçzŞæđDçāAāĀiji, d, fç■Ĺ [āŔČèĀČ  
PythonæŮĠæāç ĭāĀČ èĤZāžZāzççāAāĹĒāĹŇāžçēāĭæŞŔäyĭçĹ'žāōŽçŽĎāžŇèĤZāĹŭæŤŕæ■ōçşāđŇāēČ32ä;■  
çññäyÄäyĭā■Ůçņē < æŇĠāōŽāžĒā■ŮēĹČēāžāžŔāĀČāIJĭēĤZāyĭāĭŇā■Ŕäy■ijŇāōČēāĭçđ'žāĀĭā;Ŏä;■āIJĭāĹ■  
æŽt'æŤžèĤZāyĭā■Ůçņēäyž > èāĭçđ'žénŸä;■āIJĭāĹ■ijŇāĹŮèĀĒæŸŕ !  
èāĭçđ'žç;ŞçzIJā■ŮēĹČēāžāžŔāĀČ

āžġçŤŞçŽĎ Struct āōđäĭŇæIJĹāĭĹād'ŽāśđæĀġāŠŇæŮžæŞŤçŤĭæĭēæŞ■ā;IJçŽyāžŤçşāđŇçŽĎçzŞæđ  
size āśđæĀġāŇĒāŔŇāžĒçzŞæđDçŽĎā■ŮēĹČæŤŕijŇèĤZāIJĭ/OæŞ■ā;IJæŮŮēĭđäyŷæIJĹçŤĭāĀČ  
pack() āŠŇunpack() æŮžæŞŤçćŇçŤĭæĭēæĹŞāŇĒāŠŇèġçāŇĒæŤŕæ■ōāĀČæŕŤæČĭijŽ

```
>>> from struct import Struct
>>> record_struct = Struct('<idd')
>>> record_struct.size
20
>>> record_struct.pack(1, 2.0, 3.0)
b
↪ '\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x00\x00\x00\x00\x08@'
↪ '
>>> record_struct.unpack(_)
(1, 2.0, 3.0)
>>>
```

æIJĹæŮŮāĀŽā;æĤŸäijŽçIJŇāĹŕ pack() āŠŇ unpack()  
æŞ■ā;IJæžæĭāāĭŮçžġāĹŇāĠ;æŤŕēcŇērČçŤĭijŇçşāijjāyŇēĭçèĤZæāŭijŽ

```
>>> import struct
>>> struct.pack('<idd', 1, 2.0, 3.0)
b
↪ '\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00@\x00\x00\x00\x00\x00\x00\x00\x08@'
↪ '
>>> struct.unpack('<idd', _)
(1, 2.0, 3.0)
>>>
```

ēĤZæāŭāŔŕāžēāŭēā;IJijŇā;ĒæŸŕæĎŞēġĹæşāæIJĹāōđäĭŇæŮžæŞŤçČçāžĹaijŸēŽĒijŇçĹ'žāĹŇæŸŕāĹĭā  
ēĀŽèĤĠāĹZāzžäyÄäyĭ Struct āōđäĭŇrijŇæāijāijŔāzççāAāŔĭāijŽæŇĠāōŽāyĀæŇāāžŭäyŤæĹĀæIJĹçŽĎæ  
ēĤZæāŭäyĀæĭēāžççāAçzt'æĹđ'ārşāŔŸāĭŮæŽt'āĹāçōĀā■ŤāžĒ(āŽāyŷā;āāŔĭēIJĀèĕAæŤžāŔŸäyĀād'Ďāzççā

ērżāŔŮāžŇèĤZāĹŭçzŞæđDçŽĎāzççāAèĕAçŤĭāĹŕäyĀāžZēĭđäyŷæIJĹēŭçēĀŇāijŸç;ŎçŽĎçijŮçĭŇæĹĀ.

aIJlãGjæTřãÄread\_records äy■iijNiter() ècñçTlãlëãLZãzzäyÄäyÿlëTãZðãZzãõZãd' gârRæTřæ■õã  
 èfZäyÿlëf■äzçãZlãijZäy■æÜ■çZðërÇçTlãyÄäyÿlçTlãLüæRŘä;ZçZðãRrërÇçTlãrZèsa(æfTãæC  
 lambda: f.read(record\_struct.size)) iijN çZt'ãLřãõCèfTãZðäyÄäyÿlçL'zæõLçZðãÄij(æçCbã

```

>>> f = open('data.b', 'rb')
>>> chunks = iter(lambda: f.read(20), b'')
>>> chunks
<callable_iterator object at 0x10069e6d0>
>>> for chk in chunks:
...     print(chk)
...
b'\x01\x00\x00\x00\xff\xff\xff\xff\x02@\x00\x00\x00\x00\x00\x00\x12@'
b'\x06\x00\x00\x00\x03333333\x1f@\x00\x00\x00\x00\x00\x00"@'
b'\x0c\x00\x00\x00\x00\xcd\xcc\xcc\xcc\xcc\xcc*\x9a\x99\x99\x99\x99YL@'
>>>
    
```

æçCä;äæL'ÄègAijNãLZãzzäyÄäyÿlãRrër■äzçãrZèsaçZðäyÄäyÿlãÕšãZäæYřãõCèC;ãËÄèõyã;ççTlãyÄäy  
 æçCædIJä;äy■ä;ççTlëfZçg■æL'ÄæIJřijNéCçãZLãzççãAãRrërÇ;äijZãCŘäyNéIcèfZæüiijZ

```

def read_records(format, f):
    record_struct = Struct(format)
    while True:
        chk = f.read(record_struct.size)
        if chk == b'':
            break
        yield record_struct.unpack(chk)
    
```

aIJlãGjæTř unpack\_records() äy■ä;ççTlãZÈãRëãd'ÜäyÄçg■æÜzæşT  
 unpack\_from() ãÄÇ unpack\_from()ãrZãZÕãZÕäyÄäyÿlãd'gãdNãZNëfZãLüæTřçZðäy■æRŘãRÜãZNë  
 äZäyÿzãõCäy■äijZãzççTšãzzã;TçZðäyT'æÜüãrZèsaæLÜëÄËèfZëãNãÈËã■Yãd'■ãLüæ\$■ä;IJãÄÇ  
 ä;äãRlëIJÄèeAçZãõCäyÄäyÿlã■ÜëLÇã■Üçñæyş(æLÜæTřçZð)ãŠNäyÄäyÿlã■ÜëLÇãAŘçgZèGŘřijNãõCäijZã

æçCædIJä;äy■ä;ççTl unpack() ælëäzçæZf unpack\_from() iijN  
 ä;äëIJÄèeAäfõæTãzççãAælëædðéÄããd'gëGŘçZðãRçZðãLÇçL'GãZëãRlëfZëãNãAŘçgZèGŘçZðèõaçõÜ

```

def unpack_records(format, data):
    record_struct = Struct(format)
    return (record_struct.unpack(data[offset:offset + record_struct.
    ↪size])
            for offset in range(0, len(data), record_struct.size))
    
```

èfZçg■æÜzæqLéZd'ãZÈäzççãAçIJNäyLãÕzã;Lãd'■æIÇãd'ÜřijNëfYã;ÜãAŽã;Lãd'Žéciãd'ÜçZðãüëã;  
 äd'■ãLüæTřæ■õãZëãRlëdðéÄããrRçZðãLÇçL'GãrZèsaãÄÇ æçCædIJä;äãGÈãd'GãZÕëřãRÜãLřçZðäyÄäyÿlã  
 äijZëãlçÕřçZðæZt'ãGžèLšãÄÇ

aIJlëgçãNËçZðæÜüãÄZřijNcollections ælããlÜäy■çZðãS;ãŘ■ãËÇçZðãrZèsaæLÜëèõyæYřã;äæÇşè  
 ãõCãRfãZèèõl'ä;äçZZëfTãZðãËÇçZðèõç;õãşðæÄgãR■çgřãÄÇã;NæçCřijZ

```

from collections import namedtuple

Record = namedtuple('Record', ['kind', 'x', 'y'])
    
```

```

with open('data.p', 'rb') as f:
    records = (Record(*r) for r in read_records('<idd', f))

for r in records:
    print(r.kind, r.x, r.y)

```

æĈæđIJă;ăçŽĐċÍŇăžŔéIJĂèĕAăđ'ĐċŔĖăđ'gėĠŔçŽĐăžŇėĤZăĹŮăŤŕăĲőiiĴŇă;ăæIJĂăĕ;ă;ĤċŤÍ  
 numpy æĹăăĹŮăĂĈă;ŇăĕĈiiĴŇă;ăăŔŕăžėăŕĖăÿĂăÿĹăžŇėĤZăĹŮăŤŕăĲőĕŕzăŔŮăĹŕăÿĂăÿĹċžŞăđĐăŇŮăŤŕċ;

```

>>> import numpy as np
>>> f = open('data.b', 'rb')
>>> records = np.fromfile(f, dtype='<i,<d,<d')
>>> records
array([(1, 2.3, 4.5), (6, 7.8, 9.0), (12, 13.4, 56.7)],
      dtype=[('f0', '<i4'), ('f1', '<f8'), ('f2', '<f8')])
>>> records[0]
(1, 2.3, 4.5)
>>> records[1]
(6, 7.8, 9.0)
>>>

```

æIJăŔŔŎăŔŔăÿĂċŤiiĴŇăĕĈæđIJă;ăéIJĂèĕAăžŎăăşċŞĕçŽĐăŮĠăžŮăăiiĵăĴŕ(ăĕĈăŽĹċŤĠăăiiĵăĴŕiiĴŇă  
 ăĹŔăċĂăŞĕçIJŇċIJŇPythonăŸŕăÿăŸŕăăşċžŔăŔŔă;ŽăžĖċŎŕăŸċŽĐăĹăăĹŮăĂĈăZăăÿžăÿăăĹŕăÿĠăÿăă;

## 8.12 6.12 ĕŕzăŔŮăŤŇăĕŮăŞŇăŔŕăŔŸéŤĖăžŇėĤZăĹŮăŤŕăĲő

### éŮőĕċŸ

ă;ăéIJĂèĕAĕŕzăŔŮăŇĖăŔŇăŤŇăĕŮăĹŮăĂĖăŔŕăŔŸéŤĖăőŕă;ŤĖŤĖăŔŔĹċŽĐăđ'ăĖĬăžŇėĤZăĹŮăăiiĵăĴŕ

### ĕğċăĖşăŮăăăĹ

struct æĹăăĹŮăŔŕĕċŇċŤĹăĹĕċijŮċăA/ĕğċċăAăĠăăžŎăĹ'ĂăIJĹċşzăđŇċŽĐăžŇėĤZăĹŮăŤŕăĲőċž  
 æĹĕăĹċđ'žăÿĂăÿĹċžĐăĹŔăÿĂċşzăĹŮăđ'Žĕ;žăĹċžŽĐċŤċžĐĖŤĖăŔŔĹiiĴŹ

```

polys = [
    [ (1.0, 2.5), (3.5, 4.0), (2.5, 1.5) ],
    [ (7.0, 1.2), (5.1, 3.0), (0.5, 7.5), (0.8, 9.0) ],
    [ (3.4, 6.3), (1.2, 0.5), (4.6, 9.2) ],
]

```

ċŎŕăIJăĂĠĖőĹĕŤZăÿĹăŤŕăĲőĕċŇċijŮċăAăĹŕăÿĂăÿĹăžėăÿŇăĹŮăđ't'ėĈĹăijĂăġŇċžĐăžŇėĤZăĹŮăŮĠăž

Byte	Type	Description
0	int	æŮĠăžŮăăžċċăăiiĴŮx1234iiĴŇăŕŔċŇŕiiĴL'

4	double	x çŽĎæIJĀāřŘāĀijjijŁāřŘçńřijL'	
12	double	y çŽĎæIJĀāřŘāĀijjijŁāřŘçńřijL'	
20	double	x çŽĎæIJĀād'ğāĀijjijŁāřŘçńřijL'	
28	double	y çŽĎæIJĀād'ğāĀijjijŁāřŘçńřijL'	
36	int	äÿL'èğŠā;ćæŦřéĞŘijŁāřŘçńřijL'	

çťğèűşçİĀād't'ėĆİæŸřäÿĀçşżāŁŮçŽĎād'Žè;żā;ćèőřā;ŦřijŇçijŮčăAæăijăijŘăeĆăÿŇřijŽ

Byte	Type	Description	
0	int	èőřā;ŦéŦřāžęijĴNā■ŮèŁĆijL'	
→			
4-N	Points	(X,Y) āĴŘæăĜijŇăžēæŧōçĆzæŦřèăĴçd'ž	
→			

äÿžăŦēăĴžēŦžæăűçŽĎæŮĜăžűřijŇă;ăăŘřăžēă;ŧçŦĴăeĆăÿŇçŽĎPythonăžççăAřijŽ

```
import struct
import itertools

def write_polys(filename, polys):
    # Determine bounding box
    flattened = list(itertools.chain(*polys))
    min_x = min(x for x, y in flattened)
    max_x = max(x for x, y in flattened)
    min_y = min(y for x, y in flattened)
    max_y = max(y for x, y in flattened)
    with open(filename, 'wb') as f:
        f.write(struct.pack('<iddddi', 0x1234,
                               min_x, min_y,
                               max_x, max_y,
                               len(polys)))
        for poly in polys:
            size = len(poly) * struct.calcsize('<dd')
            f.write(struct.pack('<i', size + 4))
            for pt in poly:
                f.write(struct.pack('<dd', *pt))
```

ăřEæŦřă■őerzāŘŮāŽđæĴçŽĎæŮăăĀŽřijŇăŘřăžēăĴ'çŦĴăĜ;æŦř struct.unpack()  
řijŇăžççăAăĴĴçŽÿăijřijŇăşžæIJňăřsæŸřäÿĴéĴăeĴžæş■ă;IJçŽĎéĀēăžŘăĀĆăeĆăÿŇřijŽ

āŕ;çōæƒZāyłāzččāAāRfāzēāuēā;IJījNā;EæYřéGñÉícaūūæíCāzEā;Łād'ŽerzāRŪāĀAēgčāNĚæTřæ■ōçz  
 éĆcāeIJāĚ■āzšād'łczAæĪCāzEçČzāĀCāZāæ■d'ā;ŁæY;çDūāzTērēæIJLāRēāyĀçg■ēgčāEşæŪzæşTāRfāzēçō  
 āIJlæIJnārRēŁCæŌēāyNālēçŽDēČlāŁEījNāŁSāijZēĀRæ■ēaijTçd'žāyĀāylæZī'āŁāaijYçgĀçŽDēgčæ  
 çŽōæāĠæYřāRfāzēçzŽčíNāzRāSŸæRŘā;ZāyĀāylēnYčžgčŽDæŪĠāzūāi;āijRāNŪæŪzæşTīijNāzūçōĀāNŪ  
 æIJnārRēŁCæŌēāyNālēçŽDēČlāŁEāzččāAāzTērēæYřæTī'æIJnāzēāy■æIJĀād'■æĪCæIJĀēnYčžgčŽDā;Nā■  
 āyĀāōŽēæAāzTçzEçŽDēYĚērzaŁSāzñçŽDēōlēōzēČlāŁEījNāRēād'ŪāzşēæAāRČēĀČāyNāĒūāzŪçnāēŁCāE  
 ēēŪāĒŁījNā;ŞerzāRŪā■ŪēŁCæTřæ■ōçŽDæŪūāĀZīijNēĀŽāyāIJlæŪĠāzūāi;ĀāgNēČlāŁEāijZāNĚāR  
 āŕ;çōāstructēlāqāIŪāRfāzēēgčāNĚēfZāzZæTřæ■ōāŁřāyĀāylāĚČçzDāy■āŌzīijNāRēād'ŪāyĀçg■ēalçd'zēƒZçg  
 āřsāČRāyNēíCēƒZæāūījŽ

```

    æfZéGÑæLŠäznä;ŁçŦlāzEäyÄäyŁæRRèŁřāZlælēæłçd'žæřRäyŁçzŠædDā■ŮæōŧiijNæřRäyŁæRRèŁřāZlāN
    ā■ŸāCłāIJlĀĒĒēCłčZDāĒĒēā■ŸcijŠāĒšäv■āĀCāIJl __get__() æŮzæšTäy■iijNstruct.

```

`unpack_from()` aĜ;æTřecńċTłæİëäzŎċijŞâEşây■ēġcāNĚäyÄäyłāĀijīijNċIJAăŎzäzEęćİad'ŮċŽĎāŁEċL'Ĉ

Structure ċşzârşæYřäyÄäyłāşżçāĀċşzīijNæŎëâRŮă■ŮèŁĈæTřæ■ōăżŭă■YăĈİăIĬĬăEĚēĈĬċŽĎăEĚă  
StructField æRŘèĤřāZĬă;ĤċTłāĀĈ èĤŽèĜNă;ĤċTłāzE memoryview()  
īijNæŁSăznăijZăIĬĬăRŎēİċèřęċzEęðşēġċăŎĈæYřċTłæİëäzşāYŽċŽĎăĀĈ

ä;ĤċTłēĤZăyłāzċċăĀīijNă;ăċŎřăIĬĬăřşēĈ;ăðŽăzL'äyÄäyłēnYăşĈăñăċŽĎċzŞæĎĎăřzēsăæİēēăĬċĎ'žăyŁēĬ

```
class PolyHeader(Structure):  
    file_code = StructField('<i', 0)  
    min_x = StructField('<d', 4)  
    min_y = StructField('<d', 12)  
    max_x = StructField('<d', 20)  
    max_y = StructField('<d', 28)  
    num_polys = StructField('<i', 36)
```

äyNéİċċŽĎă;Nă■RăĬĬ'ċTłēĤZăyłċşzæİëèřzâRŮăzNăL'■æŁSăznăEŻăĒĚċŽĎăĎ'Žè;žă;ċæTřæ■ŏċŽĎăĎ't

```
>>> f = open('polys.bin', 'rb')  
>>> phead = PolyHeader(f.read(40))  
>>> phead.file_code == 0x1234  
True  
>>> phead.min_x  
0.5  
>>> phead.min_y  
0.5  
>>> phead.max_x  
7.0  
>>> phead.max_y  
9.2  
>>> phead.num_polys  
3  
>>>
```

èĤZăyłāĬŁæIĬĬ'ēŭċīijNăy■èĤĜèĤŽċġ■æŮzâijRèĤYăYřæIĬĬ'äyÄăżŽċĈęăżżċŽĎăIĬĬăŮzăĀĈéċŮăĒĬīij  
ă;EăYřèĤZăyłāzċċăĀēĤYăYřæIĬĬ'ċĈzèĜĈèĈĈīijNèĤYēIĬĬăēĀă;ĤċTłēĀĒæNĜăðZă;ŁăĎ'ŽăżTăşĈċŽĎċzE  
StructFieldīijNæNĜăðZăĀRċġzēĜRċ■L'ăĀĈăRēăĎ'ŮīijNèĤTăŽĎċŽĎċzŞæĎIĬĬşzăRĬNăăŭċăðăðăyĀă

ăżză;TăŮŭăĀZăRĬēċĀă;ăēĀĜăĬřăžEăĈRèĤZăăŭăEŮă;ŽċŽĎċşzăðŽăzL'īijNă;ăăżTèřēēĀĈèŽSăyNă;Ĥċ  
ăĒĈċşzæIĬĬ'äyÄäyłċL'zæĀġârşæYřăŏĈèĈ;ăĎ'şċecńċTłæİëăăñăĒĒēŏyăĎ'Žă;ŎăşĈċŽĎăðċŎřċzEęŁĈīijNăzŎ  
äyNéİċăĬSăİëäyċäyłă;Nă■RīijNă;ĤċTłāĒĈċşzċĬ■ă;ŏăTzéĀăäyNæŁSăznċŽĎ Structure  
ċşzīijŽ

```
class StructureMeta(type):  
    '''  
    Metaclass that automatically creates StructField descriptors  
    '''  
    def __init__(self, clsname, bases, clsdict):  
        fields = getattr(self, '_fields_', [])  
        byte_order = ''  
        offset = 0  
        for format, fieldname in fields:
```

```

        if format.startswith(('<', '>', '!', '@')):
            byte_order = format[0]
            format = format[1:]
            format = byte_order + format
            setattr(self, fieldname, StructField(format, offset))
            offset += struct.calcsize(format)
            setattr(self, 'struct_size', offset)

class Structure(metaclass=StructureMeta):
    def __init__(self, bytedata):
        self._buffer = bytedata

    @classmethod
    def from_file(cls, f):
        return cls(f.read(cls.struct_size))

```

ä;ŁçŤlæŮřčŽĎ Structure ċšziiŇä;ääŔřäzěäČŔäyŇéíçèŁZæüüăōŽázL'äyÄäyŁçzŠæđĎiijŽ

```

class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        ('d', 'min_x'),
        ('d', 'min_y'),
        ('d', 'max_x'),
        ('d', 'max_y'),
        ('i', 'num_polys')
    ]

```

æ■čæČä;äæL'ÄèġAĭijŇèŁZæüüăEŽăŕšçōĂă■Ťăđ'ŽázEăĂČæŁŠăznæüüăŁăčŽĎçšzæŮzæşŤ  
 from\_file() èŏl'æŁŠăznăIJläy■éIJĂèçAçşëéAşşăzä;ŤæŤŕæ■őçŽĎăđ'ġăŕŔăŠŇçzŠæđĎçŽĎæČĚăEĭäyŇă

```

>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.file_code == 0x1234
True
>>> phead.min_x
0.5
>>> phead.min_y
0.5
>>> phead.max_x
7.0
>>> phead.max_y
9.2
>>> phead.num_polys
3
>>>

```

äyĂæŮëä;ääijĂăġŇä;ŁçŤlăzEăĚČşziiŇä;ääŕšăŔřäzëèŏl'ăōČăŔŸă;ŮæŽŤ'ăŁăæŽzèČ;ăĂČă;ŇăçĈiijŇă  
 äyŇéíçæŸŕăŕzăL'■éíçăĚČşzçŽĎäyĂäyĽăŕŔçŽĎæŤzèŁZiiŇăŕŔă;ŽázEäyĂäyĽæŮřčŽĎè;ĚăŁl'æŔŔèŁŕăŽlă





```

class Point (Structure):
    _fields_ = [
        ('<d', 'x'),
        ('d', 'y')
    ]

class PolyHeader (Structure):
    _fields_ = [
        ('<i', 'file_code'),
        (Point, 'min'), # nested struct
        (Point, 'max'), # nested struct
        ('i', 'num_polys')
    ]

```

äzd' äžžæČŁëóůčŽDæŸřijŇăőČăžšëČ;æŇL'čĚğécĎæIJšçŽDæ■čăyŷăũëă;IJijŇæŁŚăžňăóđéŽĚæŞ■ă;IJ

```

>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.file_code == 0x1234
True
>>> phead.min # Nested structure
<__main__.Point object at 0x1006a48d0>
>>> phead.min.x
0.5
>>> phead.min.y
0.5
>>> phead.max.x
7.0
>>> phead.max.y
9.2
>>> phead.num_polys
3
>>>

```

ăĹrçŽóăĹ■ăyžæ■ćijŇăyĂăyĹăđ'ĐçŘĚăóŽéTĚëőřă;TçŽDæăĚăđăũšçzŔăĚŽăë;ăžĚăĂĆă;ĚăŸŕăĚĆăđĹ  
æŕTăĚĆijŇăđ'Žë;žă;ćăŮĞăžăăŇĚăŔňăŔŸéTĚçŽDéČĹăĹĚăĂĆ

ăyĂçğ■ăŮžăăĹăŸŕăĚŽăyĂăyĹçşzăĹëăĹčđ'žă■ŮëĹĆăŦŕăë■őijŇăŔŇăŮăăĚŽăyĂăyĹăũëăĚăăĜ;ăŦŕăë

```

class SizedRecord:
    def __init__(self, bytedata):
        self._buffer = memoryview(bytedata)

    @classmethod
    def from_file(cls, f, size_fmt, includes_size=True):
        sz_nbytes = struct.calcsize(size_fmt)
        sz_bytes = f.read(sz_nbytes)
        sz, = struct.unpack(size_fmt, sz_bytes)
        buf = f.read(sz - includes_size * sz_nbytes)
        return cls(buf)

```

```

def iter_as(self, code):
    if isinstance(code, str):
        s = struct.Struct(code)
        for off in range(0, len(self._buffer), s.size):
            yield s.unpack_from(self._buffer, off)
    elif isinstance(code, StructureMeta):
        size = code.struct_size
        for off in range(0, len(self._buffer), size):
            data = self._buffer[off:off+size]
            yield code(data)

```

çşzæŮzæşT SizedRecord.from\_file() æŸřäyÄäyſaũeäĖürijNçTſæſeazŌäyÄäyſæŮGäzũäy■erzä  
 èĤZäzşæŸřäĹLäd'ŽæŮGäzũæäijäijRäyycTſçŽDæŮzäijRãĀCäĴIäyžèĴŞăĖĕiijNăoČæŌeăRŮäyÄäyſaNĖăRnă  
 âRřeĀLçŽD includes\_size âRČæTřæNĜăoŽăžEă■ŮeŁCæTřæŸřăRęăNĖăRnăd't'ėČĹăd'ġăřRăĀC  
 äyNėſcæŸřäyÄäyſäĴNă■RăTŽăĴăæĀŌăăũăĴçTſlăzŌăd'ŽèĴzăĴcæŮGäzũäy■erzäRŮă■TçNņçŽDăd'ŽèĴzăĴcæ

```

>>> f = open('polys.bin', 'rb')
>>> phead = PolyHeader.from_file(f)
>>> phead.num_polys
3
>>> polydata = [ SizedRecord.from_file(f, '<i')
...               for n in range(phead.num_polys) ]
>>> polydata
[<__main__.SizedRecord object at 0x1006a4d50>,
<__main__.SizedRecord object at 0x1006a4f50>,
<__main__.SizedRecord object at 0x10070da90>]
>>>

```

âRřäzèçIJNăĜzĴijN SizedRecord âoďăĴNçŽDăĖĖăožèĤŸæşæIJL'ėcñèġcæďRăĜzæſeăĀC  
 âRřäzëăĴçTſ iter\_as() æŮzæşTæſeèĴĴăĴřçŽoçŽDĴijNèĤZäyſæŮzæşTæŌeăRŮäyÄäyſçzŞæďDæäijäijRă  
 Structure çşzäĴIäyžèĴŞăĖĖăĀC èĤZæăũă■RăRřäzëăĴĴAĵæt'zcŽDăŌžèġcæďRăTřæ■őĴijNăĴNăçCĴijŽ

```

>>> for n, poly in enumerate(polydata):
...     print('Polygon', n)
...     for p in poly.iter_as('<dd'):
...         print(p)
...
Polygon 0
(1.0, 2.5)
(3.5, 4.0)
(2.5, 1.5)
Polygon 1
(7.0, 1.2)
(5.1, 3.0)
(0.5, 7.5)
(0.8, 9.0)
Polygon 2
(3.4, 6.3)
(1.2, 0.5)
(4.6, 9.2)

```

```

>>>

>>> for n, poly in enumerate(polydata):
...     print('Polygon', n)
...     for p in poly.iter_as(Point):
...         print(p.x, p.y)
...
Polygon 0
1.0 2.5
3.5 4.0
2.5 1.5
Polygon 1
7.0 1.2
5.1 3.0
0.5 7.5
0.8 9.0
Polygon 2
3.4 6.3
1.2 0.5
4.6 9.2
>>>

```

āŕĖæL'ĀæIJL'æfZāzZçzŠāŖĹetūælēijŃāyŃeīcæYŕāyĀäyġ  
 āĠ;æŦçŽDāŖeād'ŪāyĀäyġæfōæ■ççL'ĹijŽ
 read\_polys()

```

class Point(Structure):
    _fields_ = [
        ('<d', 'x'),
        ('d', 'y')
    ]

class PolyHeader(Structure):
    _fields_ = [
        ('<i', 'file_code'),
        (Point, 'min'),
        (Point, 'max'),
        ('i', 'num_polys')
    ]

def read_polys(filename):
    polys = []
    with open(filename, 'rb') as f:
        phead = PolyHeader.from_file(f)
        for n in range(phead.num_polys):
            rec = SizedRecord.from_file(f, '<i')
            poly = [ (p.x, p.y) for p in rec.iter_as(Point) ]
            polys.append(poly)
    return polys

```

## eòléōž

èfZäyÄeLCaŘSä;ääsTçd'žazEèöyad'ŽénYçžgçŽDçijÚçlNæLÄæIJfrijNāNĖæNñæRRèfřaŽlrijNāzūèfšçDūeĀNrijNāoČaznéČ;äyžazEāRNāyÄäyłçL'žaoŽçŽDçZōæāGæIJ■āLāāĀČ

äyLéIççŽDāođçŎřçŽDäyÄäyłäyžèeAçL'žā;AæYřaōČæYřašžazŎæGŠègçāNĖçŽDæĀiæČšāĀČā;ŠäyĀā Structure āođä;NēcnaĹŽāžžæŮūrijN \_\_init\_\_ () äžĒāžĒāRĹæYřāĹŽāžžäyÄäyłā■ŮeLCæTřæ■ōçŽDā çL'žāĹnçŽDrijNēfZæŮūāĀŽāžžæšæIJL'āžžā;TçŽDègçāNĖæLŮeĀĒāĒūāžŮäyŎçžŠædDçZyāĒšçŽDæŠ■ā; èfZæāūāĀŽçŽDäyÄäyłāĹlæIJžæYřā;āāRřèČ;äžĒāžĒāRĹāřžäyÄäyłā■ŮeLCèōřā;TçŽDæšRäyĀāRŘéČĹāĹEæĹ

äyžazEāođçŎřæGŠègçāNĖāŠNæL'SāNĖrijNēIJĀèeAā;fçTl StructField æRŘèfřaŽlçšžāĀČ çTlæĹūāIJl \_fields\_ äy■āĹŮāGžæIèçŽDæfRäyłāsđæĀgéc;äijZècñē;ñāNŮæĹRäyÄäy StructField æRŘèfřaŽlrijN āoČāfççZyāĒšçžšædDæāijāijRçāĀāŠNāAŘçgžāĀijāĹā■YāĹrā■YāČlçijŠā■Yäy■āĀČāĒČçš StructureMeta āIJlād'ŽäyłçžšædDçšžècnaōŽāžL'æŮūeĠlāĹlāĹŽāžžazEèfZāžžæRŘèfřaŽlāĀČ æĹSāžñā;fçTlāĒČçšççŽDäyÄäyłäyžèeAāŎšāžžæYřaōČā;fā;ŮçTlæĹūeĹdāyŷæŮžā;fçŽDæŽēfGäyÄäyĹæ

StructureMeta çŽDäyÄäyłā;Ĺā;ōāççŽDāIJræŮžāřsæYřaōČāijŽāžžāōŽā■ŮeLCæTřæ■ōeāžžāRāĀ äžšāřsæYřèft'rijNāeČædIJāžžæDŘçŽDāsđæĀgæNĠāōŽāžEäyÄäyłā■ŮeLCéāžžāR(<eāłçd'žā;Ŏā;■āijYāĒĹ æĹŮeĀĒ>eāłçd'žénYā;■āijYāĒĹ)rijN éČčāRŎéIcæL'ĀæIJL'ā■ŮæōtçŽDēāžžāRēČ;äžèèfZäyłeāžžāRäyžāGE æTlæČrijNā;āāRřèČ;æIJL'äyĀāžZæfTè;Čād'■æIççŽDçžšædDrijNāřsāČRäyNēIcèfZæāūrijŽ

```
class ShapeFile(Structure):
    _fields_ = [ ('>i', 'file_code'), # Big endian
                 ('20s', 'unused'),
                 ('i', 'file_length'),
                 ('<i', 'version'), # Little endian
                 ('i', 'shape_type'),
                 ('d', 'min_x'),
                 ('d', 'min_y'),
                 ('d', 'max_x'),
                 ('d', 'max_y'),
                 ('d', 'min_z'),
                 ('d', 'max_z'),
                 ('d', 'min_m'),
                 ('d', 'max_m') ]
```

äžNāL'■æĹSāžñæRŘāĹřèfGrijNmemoryview() çŽDā;fçTlāRřāžèäyōāĹl'æĹSāžñæĀfāĒ■āĒĒā■YçŽĹ ā;ŠçžšædDā■YāIJlātNāeŮçŽDæŮūāĀŽrijNmemoryviews āRřāžèāRāāĹāāRNāyĀāĒĒā■YāNžāššäyĹāōžā èfZäyłçL'žæĀgæfTè;Čā;ōāççrijNā;EæYřaōČāĒšæšłçŽDæYřāĒĒā■YègEāž;äyŎæŽōéĀžā■ŮeLCæTřçžDçž āeČædIJā;āāIJlāyÄäyłā■ŮeLCā■ŮçñäyšæĹŮā■ŮeLCæTřçžDäyĹæL'gēāNāĹGçL'GæŠ■ā;IJrijNā;æĀŽäyŷā èĀNāĒĒā■YègEāž;āĹGçL'Gäy■æYřèfZæāūççŽDrijNāoČāžĒāžĒæYřāIJlāūsā■YāIJçŽDāĒĒā■YäyLéIcāRāā

èfYæIJL'ā;Ĺād'ŽççZyāĒšççŽDçnæeLCaŘfřāžèäyōāĹl'æĹSāžñæL'āšTèfZéGÑèóléōžçŽDæŮžæāĹāĀČ āRČeĀČ8.13ārRèLCā;fçTlæRŘèfřaŽlāēdDāžžäyÄäyłçšžādNçšççžšāĀČ 8.10ārRèLCæIJL'æZř'ād'ŽāĒšāžŎāžūèfšèōāçŮŮāšđæĀgāĀijçŽDèóléōžrijNāžūäyTēu\$NestedStructæRŘèfřa 9.19ārRèLCæIJL'äyÄäyłā;fçTlāĒČçšžæĹāĹlāgNāNŮçšžæĹRāšYçŽDā;Nā■RrijNāŠN StructureMeta çšžēĹdāyŷçZyāijijāĀČ PythonçŽD ctypes æžRçāĀāRNæāūāžšā;ĹæIJL'ēūçrijNāoČæRŘā;ŽāžEāřžāōžāžL'æTřæ■ōçžšædDāĀAæTřæ■ōçžšædDātNāeŮ

## 8.13 6.13 æṬṛæ■óçŽĎçṭ'ráŁăăŸŌçžšèőqæ\$■äĳĲ

éŬóécŸ

äĳäéĲÄèçAäđ'ĎçŘĚäŸÄäŸĳäŁäđ'ğçŽĎæṬṛæ■óéŽĚäžúéĲÄèçAèőqçőŬæṬṛæ■óæĂzăŠŇæĹŬăĚŸäžŬçž

èğčăĒşæŰzæąĹ

ăržăžŌăžžă;ṬæŭĹ'ăŔĹăĹŔçžšèőqăĂĀæŬŭéŬṭ'ăžŔăĹŬăžčăŔĹăĚŸäžŬçŽŸăĚşæĹĂæĲççŽĎæṬṛæ■óăĹĒ  
Pandasăž\$ āĂĈ

ăŸžăžĒèŎŦ'ăĳăăĚĹă;ŞéŇăŸŇĳĳăŸŇéĲæŸŕăŸĂăŸĳă;ğçŦĲPandasæĲăăĹĒæđŔèĹăĹăăŞèă\$ŌăŸĈçŽĎ  
èĂĒéĳăăŠŇăŦŏéĳçşžăĹĲĹ'æṬṛæ■óăž\$ çŽĎăĴŇă■ŔăĂĈăĲăĹĹŖăĒĒĒçĒçŕĠæŰĠçăçŽĎæŬŭăĂŽĳĳŇèĒ

```
>>> import pandas

>>> # Read a CSV file, skipping last line
>>> rats = pandas.read_csv('rats.csv', skip_footer=1)
>>> rats
<class 'pandas.core.frame.DataFrame'>
Int64Index: 74055 entries, 0 to 74054
Data columns:
Creation Date 74055 non-null values
Status 74055 non-null values
Completion Date 72154 non-null values
Service Request Number 74055 non-null values
Type of Service Request 74055 non-null values
Number of Premises Baited 65804 non-null values
Number of Premises with Garbage 65600 non-null values
Number of Premises with Rats 65752 non-null values
Current Activity 66041 non-null values
Most Recent Action 66023 non-null values
Street Address 74055 non-null values
ZIP Code 73584 non-null values
X Coordinate 74043 non-null values
Y Coordinate 74043 non-null values
Ward 74044 non-null values
Police District 74044 non-null values
Community Area 74044 non-null values
Latitude 74043 non-null values
Longitude 74043 non-null values
Location 74043 non-null values
dtypes: float64(11), object(9)

>>> # Investigate range of values for a certain field
>>> rats['Current Activity'].unique()
array([nan, Dispatch Crew, Request Sanitation Inspector], _
      ↪dtype=object)
>>> # Filter the data
```

```

>>> crew_dispatched = rats[rats['Current Activity'] == 'Dispatch_
↳Crew']
>>> len(crew_dispatched)
65676
>>>

>>> # Find 10 most rat-infested ZIP codes in Chicago
>>> crew_dispatched['ZIP Code'].value_counts()[:10]
60647 3837
60618 3530
60614 3284
60629 3251
60636 2801
60657 2465
60641 2238
60609 2206
60651 2152
60632 2071
>>>

>>> # Group by completion date
>>> dates = crew_dispatched.groupby('Completion Date')
<pandas.core.groupby.DataFrameGroupBy object at 0x10d0a2a10>
>>> len(dates)
472
>>>

>>> # Determine counts on each day
>>> date_counts = dates.size()
>>> date_counts[0:10]
Completion Date
01/03/2011 4
01/03/2012 125
01/04/2011 54
01/04/2012 38
01/05/2011 78
01/05/2012 100
01/06/2011 100
01/06/2012 58
01/07/2011 1
01/09/2012 12
>>>

>>> # Sort the counts
>>> date_counts.sort()
>>> date_counts[-10:]
Completion Date
10/12/2012 313
10/21/2011 314
09/20/2011 316

```







## èġċàEşæŪzæąŁ

årEaijzålŪaĖşéTõa■ŪaŖĆæTŗæTĹ;ålŖæşŖäy!\*aŖĆæTŗæLŪëĂĖa■Täy!\*aŖŖŌéİcârşèĈ;èĹĹ;ålŖëfZçğ■æT

```
def recv(maxsize, *, block):  
    'Receives a message'  
    pass  
  
recv(1024, True) # TypeError  
recv(1024, block=True) # Ok
```

ålŖ'çTĹëfZçğ■æLĂæIJřijNæŁSäzñëfYëĈ;åIJæŖŖŌëåŖŪäzzæDŖåd'ŽäyĹä;■ç;õaŖĆæTŗçŽDăĜ;æTŗäy■æT

```
def minimum(*values, clip=None):  
    m = min(values)  
    if clip is not None:  
        m = clip if clip > m else m  
    return m  
  
minimum(1, 5, 2, -5, 10) # Returns -5  
minimum(1, 5, 2, -5, 10, clip=0) # Returns 0
```

## èŏİëŏž

åĹŁad'ŽæĈĖăEĹäyNřijNä;fçTĹaijzålŪaĖşéTõa■ŪaŖĆæTŗäijZæŖTä;fçTĹä;■ç;õaŖĆæTŗëaĹæDŖæŽt'ålă  
ăĹNăëĈřijNëĂĈëZŚäyNăëCăyNăyĂäyĹăĜ;æTŗëŖĈçTĹřijŽ

```
msg = recv(1024, False)
```

ăëĆæđIJëŖĈçTĹëĂĖårzrecvăĜ;æTŗăžŭäy■æYŖăĹĹçEşæĈLřijNëĈcăžŪëĈŖaŏŽäy■æYŖŖçŽ;éĈCăyĹFalseă  
ăĹEæYřijNăëĈæđIJăžççăAăŖYæŁŖăyNéİcèfZæăŭa■ŖçŽDëŖăŖşæyĖæëŽad'ŽăžEřijŽ

```
msg = recv(1024, block=False)
```

årĖăd'ŪřijNä;fçTĹaijzålŪaĖşéTõa■ŪaŖĆæTŗăzşäijZæŖTä;fçTĹ\*\*kwargsaŖĆæTŗæŽt'ăë;řijNăŽăyžålIJ

```
>>> help(recv)  
Help on function recv in module __main__:  
recv(maxsize, *, block)  
    Receives a message
```

ăijzålŪaĖşéTõa■ŪaŖĆæTŗăIJĹäyĂăžZæŽt'énYçžğăIJzăŖĹăŖNăăŭăžşăĹLăIJĹçTĹăĂĈ  
ăĹNăëĈřijNăŏCăžnăŖŖăžëèççTĹăİăIJĹä;fçTĹ\*argsăŖN\*\*kwargsaŖĆæTŗă;IJăyžëĹŞăĖëçŽDăĜ;æTŗăy■æŖŚ



## èġċàEşæŮzæąŁ

äyžäzEèĈ;èŁTāZđāđ'ŽäyłāĀijīijŃāĠ;æTŕçŽt' æŬëreturnäyĀäyłāĒĈçzĎārsèāŃāžEāĀĈă;ŃāęĈīijŽ

```
>>> def myfun():
...     return 1, 2, 3
...
>>> a, b, c = myfun()
>>> a
1
>>> b
2
>>> c
3
```

## èőléőž

ār;çőāmyfun()çIJŃäyŁāŬžèŁTāZđāžEāđ'ŽäyłāĀijīijŃāőđéŽĒäyŁæŸrāĒŁāŁZāžzāžEäyĀäyłāĒĈçzĎĈĎ  
èŁŽäyłēr■æşTçIJŃäyŁāŬžæŕTè;ĈāęĠæĀīijŃāőđéŽĒäyŁæŁSāžñā;ŁçTłçŽĎæŸréĀŬāRūæİęĈTşæŁRäyĀäy

```
>>> a = (1, 2) # With parentheses
>>> a
(1, 2)
>>> b = 1, 2 # Without parentheses
>>> b
(1, 2)
>>>
```

ā;ŞæŁSāžñērĈçTłèŁTāZđäyĀäyłāĒĈçzĎĈŽĎāĠ;æTŕçŽĎæŬūāĀŽ  
īijŃēĀŽāyŷæŁSāžñāijŽārEçzŞæđIJètŃāĀijçzŽāđ'ŽäyłāRŸéĠRīijŃārśāĈRäyŁéİççŽĎéĈçæūāĀĈ  
āĒūāőđéŁŽārśæŸŕ1.1ārRèŁĈäy■æŁSāžñāL'Āèŕ'çŽĎāĒĈçzĎĒğĈāŃĒāĀĈèŁTāZđçzŞæđIJāžşārŕāžèè;ŃāĀij  
èŁŽæŬūāĀŽèŁŽäyłāRŸéĠRāĀijārśæŸrāĠ;æTŕèŁTāZđçŽĎéĈçäyłāĒĈçzĎæIJñèžñāžEīijŽ

```
>>> x = myfun()
>>> x
(1, 2, 3)
>>>
```

## 9.5 7.5 āőŽāžŁ'æIJŁ'éžŸëőđ'āŔĆæTŕçŽĎāĠ;æTŕ

### éŬőéćŸ

ä;ăæĈşāőŽāžŁ'äyĀäyłāĠ;æTŕæŁŬëĀĒæŮzæşTīijŃāőĈçŽĎäyĀäyłāĒŬāđ'ŽäyłāŔĆæTŕæŸŕāŔréĀŁçŽ

## èġċàEşæŮzæąŁ

åőŽázŁ'äyÄäylæIJL'ârřéĀL'ârĈCæŤřčŽĎăĜ;æŤřæŸřéİdäyÿçőĀ■ŤçŽĎiijŇčŽt'æŎěāIJĀĜ;æŤřăőŽázŁ

```
def spam(a, b=42):  
    print(a, b)  
  
spam(1) # Ok. a=1, b=42  
spam(1, 2) # Ok. a=1, b=2
```

åęĆæđIJézŸëöd'ârĈCæŤřæŸřäyÄäylăŤřăřăőæŤžčŽĎăőžăŽĭærŤăęCâyÄäylăĹŮëąĭăĀęŻĒăŖĹæĹŮëĂĔ

```
# Using a list as a default value  
def spam(a, b=None):  
    if b is None:  
        b = []  
    ...
```

åęĆæđIJă;ăăžŮäy■æĈşæŖŖă;ZăyÄäylézŸëöd'ăĀijiiijŇëĀŇæŸřæĈşăžĒăžĒæŤŇerŤăyŇæşŖăylézŸëöd'

```
_no_value = object()  
  
def spam(a, b=_no_value):  
    if b is _no_value:  
        print('No b value supplied')  
    ...
```

æĹSăžŋæŤŇerŤăyŇëĤŽăylăĜ;æŤřiiijŽ

```
>>> spam(1)  
No b value supplied  
>>> spam(1, 2) # b = 2  
>>> spam(1, None) # b = None  
>>>
```

ăžŤčžEęĈCărşăŖŖăžăŖŖşĈŎŕăĹŕăijăęĀşăyÄäylNoneăĀijăŖŇăy■ăijăăĀijăyđ'çğ■æĈĔăĒŤæŸřæIJL'ăũőăĹ

## ëőĹëőž

åőŽázŁ'äyęézŸëöd'ăĀijăŖĈCæŤřčŽĎăĜ;æŤřæŸřăĹĹčőĀ■ŤçŽĎiijŇă;Eçžĭäy■ăžĒăžĒăŖĭæŸřëĤŽăylriijŇ  
éęŮăĒĹiijŇézŸëöd'ârĈCæŤřčŽĎăĀijăžĒăžĒăIJĀĜ;æŤřăőŽázŁ'čŽĎăŮăăĀŽëŤŇăĀijăyĀæŋăăĀĈërŤçĹ

```
>>> x = 42  
>>> def spam(a, b=x):  
...     print(a, b)  
...  
>>> spam(1)  
1 42  
>>> x = 23 # Has no effect
```

```
>>> spam(1)
1 42
>>>
```

æşlæĐRăĹră;ŞæĹSăznæŤzăRŸxçŽDăĀijçŽDæŮŭăĂZăfzézŸèod'ăRCæŤrăĀijăzŭæşæIJĹ'ă;śăŞ■ijNè  
ăĔŭăñaiijNézŸèod'ăRCæŤrçŽDăĀijăzŤèrèæŸrăy■ăRrăRŸçŽDărfzèşaiijNærŤăçCNoneăĂATrueăĂAFal  
çĹ'zăĹnçŽDriijNă■ČăyGăy■èçAăCRăyNéİcèfZæăŭăEŽăzčçăĀijŽ

```
def spam(a, b=[]): # NO!
    ...
```

æçCædIJă;æèfZăzĹăAŽăzEijNă;ŞézŸèod'ăĀijăIJăĔŭăzŮăIJræŮzècnaŕŌæŤzăRŌă;ăârEăijŽéAĞăĹrăR

```
>>> def spam(a, b=[]):
...     print(b)
...     return b
...
>>> x = spam(1)
>>> x
[]
>>> x.append(99)
>>> x.append('Yow!')
>>> x
[99, 'Yow!']
>>> spam(1) # Modified list gets returned!
[99, 'Yow!']
>>>
```

èfŽçg■çzŞæđIJăzŤèrèäy■æŸră;ăæČşèçAçŽDăĂCăyžăzEéAŕăăĔŭă■èfŽçg■æČĔăĔçŽDăRŚçŤşriijNæIJĂă  
çĹDŭăRŌăIJăĠă;æŤrèGŤNéİcæçĂæşèăŕŌçriijNăĹ■éİcçŽDă;Nă■RăŕşæŸrèfZæăŭăAŽçŽDăĂC

ăIJăŤNërŤNoneăĀijæŮŭă;ŕçŤĹ is æŞ■ă;IJçnæŸră;ĹéG■èçAçŽDriijNăzşæŸrèfZçg■æŮzæăĹçŽDăĔş  
æIJĹæŮŭăĂZăđ'ğăŕŭăijŽçĹrăyNăyNéİcèfZæăŭçŽDéŤŽèŕriijŽ

```
def spam(a, b=None):
    if not b: # NO! Use 'b is None' instead
        b = []
    ...
```

èfZăzĹăEŽçŽDèŮŕécŸăIJăzŌăŕ;çŕăNoneăĀijçăŕăŕăđæŸrècna;ŞæĹRFalseiijN  
ă;EæŸrèfŸæIJĹăĔŭăzŮçŽDărfzèşă(ærŤăçCŤŕăžæyžŌçŽDă■ŮçnæyşăĂĂăĹŮèăĹăĂĂăĔCçzDăĂĂă■ŮăĔy  
ăZăæ■d'riijNăyĹéİcçŽDăzčçăĀăijŽèŕŕărEăyĂăzŽăĔŭăzŮè;ŞăĔĔăzşă;ŞæĹRæŸræşæIJĹ'è;ŞăĔĔăĂCærŤăçC

```
>>> spam(1) # OK
>>> x = []
>>> spam(1, x) # Silent error. x value overwritten by default
>>> spam(1, 0) # Silent error. 0 ignored
>>> spam(1, '') # Silent error. '' ignored
>>>
```

æIJĀāŔŌäyÄäyléŮóécŸæŕTē;Ĉā;ôæŻiijNéCčāŕsæŸŕäyÄäylāĜ;æTŕéIJĀèèAætNērTæšŔäylāŔŕéĀL'āŔ  
èĚŽæŮūāĀŽéIJĀèèAārŔāĤĈĈŽĎæŸŕā;äāy■èĈ;ĉTlæšŔäyléžŸèóđ'āĀijæŕTāēĈNoneāĀA  
0æĹŮèĀĒFalseāĀijæĪæætNērTĉTlæĹuæŔŔā;ZĉŽĎāĀij(āZāyžèĚŽāžZāĀijéĈ;æŸŕāŔĹæšTĉŽĎāĀijīijNæŸ  
āZāæ■d'īijNā;æéIJĀèèAāĒūāzŮĉŽĎèĝčāEşæŮzæāĹāžĒāĈ

äyžāžEèĝčāEşèĚŽäyléŮóécŸīijNā;āāŔŕäzēāĹZāžžäyÄäylĉNñäyĀæŮāāžNĉŽĎĉĝAæIJL'āržèśāāóđā;Nīij  
āIJĪāĜ;æTŕéĜNéĪĉīijNā;āāŔŕäzēēĀŽèĚĜæĉĀæšèèĉnāijæĀŠāŔĈæTŕāĀijēūšèĚŽäylāóđā;NæŸŕāŔēäyĀæāū  
èĚŽéĜNĉŽĎæĀĪēūŕæŸŕĉTlæĹuäy■āŔŕēĈ;āŌžāijæĀŠèĚŽäyl\_no\_valueāóđā;Nā;IJäyžè;ŠāĒēāĈ  
āZāæ■d'īijNèĚŽéĜNéĀŽèĚĜæĉĀæšèèĚŽäylāĀijāršèĈ;ĉāóāōŽæšŔäylāŔĈæTŕæŸŕāŔēèĉnāijæĀŠèĚŽæĪēāž

èĚŽéĜNārž object() ĉŽĎā;ĤĉTlĉIJNäyĹāŌzæIJL'ĉĈžäy■ād'ĪäyŷèĝAāĈObject  
æŸŕpythonäy■æĹ'ĀæIJL'ĉšžĉŽĎāšžĉšžāĈ ä;āāŔŕäzēāĹZāžž object  
ĉšžĉŽĎāóđā;NīijNā;EæŸŕèĚŽāžZāóđā;NæšāžĀāžĹāóđéŽĒĈTlād'DīijNāZāyžāōĈāzūæšāæIJL'āžžā;TæIJL  
āžšæšāæIJL'āžžā;Tāóđā;NæTŕæ■ō(āZāyžāōĈæšāæIJL'āžžā;TĉŽĎāóđā;Nā■ŮāĒŸīijNā;āĉTŽèĜšéĈ;äy■èĈ;  
ā;āāŦŕäyĀèĈ;āĀZĉŽĎārŕæŸŕæ;NērTāŔNäyĀæĀĝāĈĈèĚŽäylāĹZāē;ĉņēāŔĹæĹŚĉŽĎèèAæšĈīijNāZāyžæĹ

## 9.6 7.6 āōŽāžĹ'āNĚāŔ■æĹŮāĒĒèĀŦāĜ;æTŕ

### éŮóécŸ

ä;āæĈšäyž sort() æŞ■ä;IJāĹZāžžäyÄäylā;Ĺĉş■ĉŽĎāZđērĈāĜ;æTŕīijNā;ĒāŔĹäy■æĈşĈTl  
def āŌžāĒZäyÄäylā■TēāNāĜ;æTŕīijN ēĀNæŸŕäyNæIJŽéĀŽèĚĜæšŔäylāĤŕāæ■ūæŮāijŔäzēāĒĒèĀŦæŮžāij

### èĝčāEşæŮzæāĹ

ā;ŠäyĀāžZāĜ;æTŕā;ĹĉōĀā■TīijNāžĒāžĒāŔĹæŸŕèōāĉōŮäyÄäylēāĹè;āijŔĉŽĎāĀijĉŽĎæŮūāĀŽīijNārš

```
>>> add = lambda x, y: x + y
>>> add(2, 3)
5
>>> add('hello', 'world')
'helloworld'
>>>
```

èĚŽéĜNā;ĤĉTlĉŽĎλæāĹè;āijŔēūšäyNéĪĉĉŽĎæTŕĹæđIJæŸŕäyĀæāūĉŽĎīijŽ

```
>>> def add(x, y):
...     return x + y
...
>>> add(2, 3)
5
>>>
```

λæāĹè;āijŔāĒŸāđNĉŽĎā;ĤĉTlāIJæŽŕæŸŕæŌšāžŔæĹŮæTŕæ■ōreduceĉ■ĹīijŽ

```
>>> names = ['David Beazley', 'Brian Jones',
...          'Raymond Hettinger', 'Ned Batchelder']
>>> sorted(names, key=lambda name: name.split()[-1].lower())
```

```
['Ned Batchelder', 'David Beazley', 'Raymond Hettinger', 'Brian_
↪Jones']
>>>
```

## èõléõž

år;çõλλbdaèałè;āijRāĖĖøÿä;āāōŽāzL'çōĀ■TāĜ;æTřijNā;EæYřāōČžDā;£çTłæYřæIJL'ėŽRāLŪç  
ä;āāRłèČ;æNĜāōŽā■Tāyłèałè;āijRijNāōČžDāĀijāřsæYřæIJAāRŌčŽDè£TāZđāĀijāĀCāzšāřsæYřèřt'äy■  
āNĖæNñād'Žāyłèr■āRēāĀAæIāzūèałè;āijRāĀAè£■āzčāzēāRŁāijCāyŷad'DçRĖç■L'ç■L'āĀĆ

ä;āāRřāzēäy■ä;£çTłλbdaèałè;āijRāřsèČ;çijŪāĖŽād'gēČlāLEpythonāzččāAāĀĆ  
ä;EæYřijNā;ŠæIJL'āžžçijŪāĖŽād'gēGRèøaçōŪèałè;āijRāĀijčŽDç\$■āřRāĜ;æTřæLŪèĀĖéIJAèeAçTłæLūa  
ä;āāřsāijŽçIJNāLřλbdaèałè;āijRçŽDèžnā;śāžĖāĀĆ

## 9.7 7.7 āNĖāR■āĜ;æTřæ■TèŌuāRŸéĜRāĀij

### éŬóécŸ

ä;āçTłλbdaāōŽāzL'āžĖāyĀāyłāNĖāR■āĜ;æTřijNāzūæČšāIJlāōŽāzL'æŪūæ■TèŌuāLřæšRāžZāRŸéĜ

### èğčāĖşæŪzæał

āĖŁçIJNāyNāyNéłcāzččāAçŽDæTłæđIJijŽ

```
>>> x = 10
>>> a = lambda y: x + y
>>> x = 20
>>> b = lambda y: x + y
>>>
```

çŌřāIJlāŁŚéŬōä;āijNa(10)āŠN(10)è£TāZđçŽDçzŠæđIJæYřāzĀāzLijšāçCæđIJä;æøđ'äyžçzŠæđIJæY

```
>>> a(10)
30
>>> b(10)
30
>>>
```

è£ŽāĖŪāy■çŽDāčēāçŽāIJlāžŌλbdaèałè;āijRāy■çŽDxæYřāyĀāyłèĜłçTśāRŸéĜRijN  
āIJłèRēāNæŪūçzŠāōŽāĀijijNèĀNāy■æYřāōŽāzL'æŪūāřçzŠāōŽijNè£ŽèušāĜ;æTřçŽDèzŸèød'āĀijāRČa  
āZāæ■đ'rijNāIJłèřČçTłè£Žāyłλbdaèałè;āijRçŽDæŪūāŽijNççŽDāĀijæYřæL'gēāNæŪūçŽDāĀijāĀCä;N

```
>>> x = 15
>>> a(10)
25
>>> x = 3
```

```
>>> a(10)
13
>>>
```

æ̥CædIJä;äæČšèol' æšŘäyIaŃfäŘ■aĜ;æTřaIJláoŽžäZl' æUúäršæ■TěOúáLřaĀijuijŃaŘřazěärĚēČčäyIaŘC

```
>>> x = 10
>>> a = lambda y, x=x: x + y
>>> x = 20
>>> b = lambda y, x=x: x + y
>>> a(10)
20
>>> b(10)
30
>>>
```

èóìèőž

ãIJlè£ZéGÑãLŮãGžæIëçŽDěŮóécŸæŸræŮřæL'Ñã;ŁãőžæŸŞçŁřçŽDěŤZěrríijÑæIJL'ăžZæŮřæL'ÑãRřæ  
 ærŤæçŮijÑéAžè£GãIJlăyĂăyIã;łçŮřæLŮãLŮëãŁăřijăy■ãŁžăžăyĂăyIλbdaëãlë;ł;ăijRãLŮëãlŮijÑăžŮă

```
>>> funcs = [lambda x: x+n for n in range(5)]
>>> for f in funcs:
...     print(f(0))
...
4
4
4
4
4
>>>
```

ä;EæYřaóđéŽĚæTŁæđIĲæYřèřĚŘæŇæYřŋčŽĎăĀijäyžèř■äzččŽĎæIĴăĀřŮăyĀăyĴăĀijăĀĈčŮăĴĴăĴŤăĴŤă

```
>>> funcs = [lambda x, n=n: x+n for n in range(5)]
>>> for f in funcs:
...     print(f(0))
...
0
1
2
3
4
>>>
```

éĀžēfĠgā;ƒçĬlāĠg;æTřézĚēod'āĀijāRĆæTřā;ćāijRġijŊlambdāāĠg;æTřāIĬlāōZāzL'æUūārsēČ;çzSāōZāLřā



æIJñēŁĆēęAęğçâEşçŽĐēŮőécÿæYřeól'ăŎșæIJñäy■ăĖijăôzčŽĐazččăAărRāzēäyĂęțūăûëă;IJăĂCâyNéİ  
çñňäyÄăyłă;Nă■RăYřrijNăAĞēōļă;ăæIJL'äyĂăyłčČzčŽĐālŮēalēlēēalčd'ž(x,y)ălĪrăăGăĚČčzĐăĂĆ  
ă;ăârRāzēä;ľcTlăyNéİlcčŽĐăĠ;æTrăīēēōaçŮăyđ'ćCzázNéŮt'čŽĐēũłęcziiž



```
from socketserver import StreamRequestHandler, TCPServer

class EchoHandler(StreamRequestHandler):
    def handle(self):
        for line in self.rfile:
            self.wfile.write(b'GOT:' + line)

serv = TCPServer(('', 15000), EchoHandler)
serv.serve_forever()
```

äy■ēfĜiijŇŅAĜēō;ä;äæČšçzŽEchoHandlerāćđāŁääyÄäyłāŔřāzēæŎēāŔŮāĚŮāzŮēĚ■ç;őéĀŁ'ēāzčŽD  
\_\_init\_\_ æŮzæşŦāĀĆærŦæĆiijŽ

```
class EchoHandler(StreamRequestHandler):
    # ack is added keyword-only argument. *args, **kwargs are
    # any normal parameters supplied (which are passed on)
    def __init__(self, *args, ack, **kwargs):
        self.ack = ack
        super().__init__(*args, **kwargs)

    def handle(self):
        for line in self.rfile:
            self.wfile.write(self.ack + line)
```

ēfŽāzŁāfōæŦzāŔŎiijŇŅĹSāznāršäy■ēIJĀēēAæŸ;āijŔāIJŔāIJŦCPServerčšzäy■æūzāŁāāL■çijĀāzĒāĀ  
ä;EæŸřā;āāE■æñæēŦŔēāŇçĹŇāžŔāŔŎäiijŽæĹčšzäiijäyŇēĹčçŽDēŦŽēřriijŽ

```
Exception happened during processing of request from ('127.0.0.1',
→59834)
Traceback (most recent call last):
...
TypeError: __init__() missing 1 required keyword-only argument: 'ack
→'
```

āĹĹiçIJŇēŦuālēāē;āČŔā;ĹēŽ;āfōæ■čēfŽäyłēŦŽēřriijŇēŽd'āzĒāfōæŦz  
socketserver æĹāĹŮæžŔāzččāAæĹŮēĀĒä;ŦçŦĹæšŔāžZāēĜæĀčŽDæŮzæşŦāžŇād'ŮāĀĆ  
ä;EæŸřriijŇæČæđIJä;ŦçŦĹ partial() āřšēČ;ā;Ĺē;žæĹ;çŽDēğčāEşāĀŦāĀŦçzŽāōČäiijæēĀš  
ack āŔĆæŦŦçŽDāĀijælēāĹĪğŇāŇŮā■şāŔřiijŇæČäyŇriijŽ

```
from functools import partial
serv = TCPServer(('', 15000), partial(EchoHandler, ack=b'RECEIVED:
→'))
serv.serve_forever()
```

āIJĹēfŽäyłä;Ňā■Ŕäy■iijŇ\_\_init\_\_() æŮzæşŦäy■çŽDack-  
āŔĆæŦŦäçŕæŸŎæŮžāijŔçIJŇäyŁāŎžā;ĹæIJĹēüçriijŇāĒŮāōđārşæŸŕäçŕæŸŎackäyžäyĀäyłāijžāĹŮāĒşēŦŎā■  
āĒşāžŎāijžāĹŮāĒşēŦŎā■ŮāŔĆæŦŦēŮōēçŸæĹSāznāIJŦ.2ārŔēĹĆæĹSāznāūşçzŔēōlēōžēfĜāžĒiijŇēržeĀĒĀŔ  
ā;ĹĹād'ŽæŮūāĀŽ partial() ēČ;āōđçŎŦçŽDæŦĹæđIJiijŇlambdaēālē;āijŔāžşēČ;āōđçŎŦāĀĆærŦæÇ

```

points.sort(key=lambda p: distance(pt, p))
p.apply_async(add, (3, 4), callback=lambda result: output_
    ↪ result(result, log))
serv = TCPServer(('', 15000),
    lambda *args, **kwargs: EchoHandler(*args, ack=b'RECEIVED:',
    ↪ **kwargs))

```

æŁæāũāEŻāzšèĈ;ăôđĉŎřăŔŇăăũĉŽĐæŦŁæđIĲijŇăy■ēŁĜĉŽŷæŕŦēĂŇăũšăijŽæŸ;ă;ŮæŕŦē;ĈēĜĈēĈ  
 ēŁæŮũăĂŽă;ŁĉŦĲpartial() âŔŕăžæŽŦ'ăŁăĉŽŦ'ēĝĈĉŽĐēăĹē;ă;ăĉŽĐæĐŔăŽ;Ĺ(ĉžŽæšŔăžŽăŔĈæŦŕécĐ

## 9.9 7.9 âŔĖă■ŦæŮzæşŦĉŽĐĉşzè;ŋæ■ćăŷžăĜ;æŦŕ

### éŮóécŸ

ăĵăæIJĻ'ăŷĂăŷĹēŽđ' \_\_init\_\_() æŮzæşŦăđ'ŮăŔăăŏŽăžĻ'ăžĖăŷĂăŷĹæŮzæşŦĉŽĐĉşzăĂĈăŷžăžĖĉŏĂ

### ēĝĉăĖşæŮzæăĹ

âđ'ĝăđ'ŽæŦŕæĈĖăĖŦăŷŇĲijŇăŔŕăžæă;ŁĉŦĲēŮ■ăŇĖăĹēăŕĖă■ŦăŷĹæŮzæşŦĉŽĐĉşzè;ŋæ■ćăĹŔăĜ;æŦŕăĂ  
 äŷ;ăŷĹă;Ňă■ŔĲijŇăŷŇĲēĈđ'žă;Ňăŷ■ĉŽĐĉşzăĂĖăŏŷă;ŁĉŦĲēĂĖăăžæ■ŏæşŔăŷĹăĹăăĹæŮzæăĹăĹēēŎăŔŮă

```

from urllib.request import urlopen

class UrlTemplate:
    def __init__(self, template):
        self.template = template

    def open(self, **kwargs):
        return urlopen(self.template.format_map(kwargs))

# Example use. Download stock data from yahoo
yahoo = UrlTemplate('http://finance.yahoo.com/d/quotes.csv?s={names}
    ↪ &f={fields}')
for line in yahoo.open(names='IBM,AAPL,FB', fields='sl1clv'):
    print(line.decode('utf-8'))

```

ēŁŽăŷĹĉşzăŔŕăžæžèĉŋăŷĂăŷĹæŽŦ'ĉŏĂă■ŦĉŽĐăĜ;æŦŕăĹēăžăæŽĲĲijŽ

```

def urltemplate(template):
    def opener(**kwargs):
        return urlopen(template.format_map(kwargs))
    return opener

# Example use
yahoo = urltemplate('http://finance.yahoo.com/d/quotes.csv?s={names}
    ↪ &f={fields}')
for line in yahoo(names='IBM,AAPL,FB', fields='sl1clv'):
    print(line.decode('utf-8'))

```

## ěóíěőž

ād' gēČlāĽĚæČĚāĚtāyŇiijŇā;ăæŇēæIJĽ'ăyĀăyĽā■TæŮzæşTçşzçŽDăŮşăZăæYréIJĀēēAā■YăČlāşŘăžZ  
æŕTăēČiijŇăőZăZĽ'UrlTemplateçşzçŽDăTŕăyĀçŽŏçŽDăŕsæYŕăĚĽăIJĽæşŘăyĽăIJŕæŮzā■YăČlāēāēĽăĀiijŇ

ă;ĽçTĽăyĀăyĽăĚĚēČlāĜ;æTŕăĽŮēĀĚēŮ■ăŇĚçŽDăŮzæāĽēĀŽăyŷăijŽæZt'ăijYéZĚăyĀăžZăĀČŏĀă■  
ăŕĽăy■ēĽĜăIJĽăĜ;æTŕăĚĚēČlāyēăyĽăžĚăyĀăyĽēčĽăd'ŮçŽDăŕYéĜŔçŎŕăčČăĀČēŮ■ăŇĚăĚşēTŏçĽ'žçČžŕs:  
ăŽăæ■d'ijŇăIJĽăĽsăžŇçŽDēğčăĚşæŮzæāĽăy■ijŇopener()ăĜ;æTŕēŏŕă;ŔăžĚ  
templateăŔČæTŕçŽDăĀiijŇăžŭăIJĽăŎăyŇăĽēçŽDēŕČçTĽăy■ă;ĽçTĽăŏČăĀČ

ăžžă;TæŮŭăĀŽăŕĽēēAă;ăçčŕăĽŕēIJĀēēAçžZăşŘăyĽăĜ;æTŕăčđăĽăēčĽăd'ŮçŽDçĽŭăĀăĽăæAŕçŽDēŮ  
çŽyæŕTăŕĚă;ăçŽDăĜ;æTŕē;Ňă■čăĽŔăyĀăyĽçşzēĀŇēĽĀiijŇēŮ■ăŇĚēĀŽăyŷăYŕăyĀçğ■æZt'ăĽăçŏĀæŕ'ĀăŞ

## 9.10 7.10 äyēéčĽăd'ŮçĽŭăæĀăĽăæAŕçŽDăŽdēŕČăĜ;æTŕ

### éŮŏéčY

ă;ăçŽDăžččăĀăy■ēIJĀēēAă;ĽēŮăĽŕăŽdēŕČăĜ;æTŕçŽDă;ĽçTĽ(æŕTăēČăžŇăžŭăd'ĐçŔĚăŽĽăĀăç■Ľ'ă;Ě  
ăžŭăyTă;ăēĽYēIJĀēēAēŏĽ'ăŽdēŕČăĜ;æTŕăŇēæIJĽ'ēčĽăd'ŮçŽDçĽŭăĀăĀiijŇăžēă;ĽăIJĽăŏČçŽDăĚĚēČĽă

### ēğčăĚşæŮzæāĽ

ēĽŽăyĀăŕŔēĽČăyžēēAēŏíěőžçŽDăYréČčăžZăĜžçŎŕăIJĽă;Ľăd'ŽăĜ;æTŕăžŞăŇăŇăæĚăđŭăy■çŽDăŽdēŕ  
ăyžăžĚăijTçd'žăyŎăŕŇŕTijŇăĽsăžŇăĚĽăŏZăZĽ'ăçČăyŇăyĀăyĽēIJĀēēAēŕČçTĽăŽdēŕČăĜ;æTŕçŽDăĜ;æT

```
def apply_async(func, args, *, callback):  
    # Compute the result  
    result = func(*args)  
  
    # Invoke the callback with the result  
    callback(result)
```

ăŏđēŽĚăyĽiijŇēĽŽăŕăžččăĀăŔŕăžēăĀŽăžžă;TæZt'ēŇYçžğçŽDăd'ĐçŔĚiijŇăŇĚæŇŇçĽçĽŇăĀăēĽç  
æĽsăžŇăžĚăžĚăŕĽēIJĀēēAăĚşæşĽăŽdēŕČăĜ;æTŕçŽDēŕČçTĽăĀČăyŇēĽăæYŕăyĀăyĽăijTçd'žăĀŎăăŭă;ĽçTĽă

```
>>> def print_result(result):  
...     print('Got:', result)  
...  
>>> def add(x, y):  
...     return x + y  
...  
>>> apply_async(add, (2, 3), callback=print_result)  
Got: 5  
>>> apply_async(add, ('hello', 'world'), callback=print_result)  
Got: helloworld  
>>>
```

```
def print_result():
    """Print the result of the asynchronous operation"""
    result = await loop.run_in_executor(None, func)
    print(f'Result: {result}')

# Create an event loop and run the asynchronous operation
loop = asyncio.get_event_loop()
loop.run_until_complete(print_result())
```

```
class ResultHandler:

    def __init__(self):
        self.sequence = 0

    def handler(self, result):
        self.sequence += 1
        print(f'[{self.sequence}] Got: {result}')
```

```
def handler():
    """Print the result of the asynchronous operation"""
    result = await loop.run_in_executor(None, func)
    print(f'Result: {result}')
```

```
>>> r = ResultHandler()
>>> apply_async(add, (2, 3), callback=r.handler)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=r.handler)
[2] Got: helloworld
>>>
```

Now we can create a handler function that can be used to handle the result of the asynchronous operation.

```
def make_handler():
    sequence = 0
    def handler(result):
        nonlocal sequence
        sequence += 1
        print(f'[{sequence}] Got: {result}')
    return handler
```

Now we can create a handler function that can be used to handle the result of the asynchronous operation.

```
>>> handler = make_handler()
>>> apply_async(add, (2, 3), callback=handler)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=handler)
[2] Got: helloworld
>>>
```

Now we can create a handler function that can be used to handle the result of the asynchronous operation.

```
def make_handler():
    sequence = 0
    while True:
```

```

result = yield
sequence += 1
print('[] Got: {}'.format(sequence, result))

```

ărzăžŌă■RçlNriiNă;ăeIJĂēēAă;fçTlăŌČçŽD send() æŰzæşTă;IJăyžăŽđērČăĜ;æTřriiNăēCăyNăL'Ăç

```

>>> handler = make_handler()
>>> next(handler) # Advance to the yield
>>> apply_async(add, (2, 3), callback=handler.send)
[1] Got: 5
>>> apply_async(add, ('hello', 'world'), callback=handler.send)
[2] Got: helloworld
>>>

```

## ěőlěőž

ăşžăžŌăŽđērČăĜ;æTřçŽĐē;řăžŭēĂžăyŷēČ;æIJL'ăRřēČ;ăRŶă;ŰēIdăyŷăđ'■æIČăĂCăyĂēČlăLēăŌşăŽăZăă■d'riiNăērŭăēŖCăL'gëăNăŖNăd'DçRĚçžŖæđIJăžNēŰt'çŽĐăL'gëăNçŖŌřăčČăŏđēŽĚăyLăŭşçžRăyčăđ'săžĚēČčă;ăăřŝăfĚēăžăŌžēğčăĚşăēČă;TăfIă■ŶăŖNăĂăđ'■çŽŷăĚşçŽĐçLŭăĂăfăæAřăžĚăĂČ

ēĜşăřŖŖæIJL'ăyđ'çğ■ăyžēēAæŰžăijRălēæ■TēŌŭăŖNăfIă■ŶçLŭăĂăfăæAřriiNă;ăăRřăžēăIJăyĂăyIărăyđ'çğ■æŰžăijRçŽŷăřTřijNēŰ■ăNĚăLŰēőyăŶřăŽt'ăLăē;žēĜRçžğăŖNēĜlçDŭăyĂçČžriiNăŽăyžăŏČăžŭăăŏČăžŭēfŶēČ;ēĜlăLă■TēŌŭăL'ĂæIJL'ēčŭă;fçTlăLřçŽĐăRŶēĜRăĂČăZăă■d'riiNă;ăăŰăēIJăăŌžăNĚăf

ăēČăđIJă;fçTlăŰ■ăNĚriiNă;ăēIJĂēēAæşlăēĐRăřžēČčăžŽăRřăfŏăTžăRŶēĜRçŽĐăŖ■ă;IJăĂČăIJăyLēnonlocalăčřăŶŌēr■ăRēçTlălēăNĜçđ'žăŌăyNălēçŽĐăRŶēĜRăijŽăIJăŽđērČăĜ;æTřăy■ēčŭăfŏăTžă

ēĂNă;fçTlăyĂăyIă■RçlNălēă;IJăyžăyĂăyIăŽđērČăĜ;æTřăřŖŖæŽt'æIJL'ēŭčăžĚriiNăŏČēŭşēŰ■ăNĚăŰžăæŖçğ■ăĐRăžL'ăyLălēēŏŝriiNăŏČăŶ;ă;ŰăŽt'ăLăçŏĂăř' AřriiNăŽăyžăĂăžăĚŝăřŝăyĂăyIăĜ;æTřēĂNăŭŝăĂăžŭăyTřijNă;ăăRřăžēă;LēĜlçTŖçŽĐăfŏăTžăRŶēĜRēĂNăŰăēIJăăŌžă;fçTl nonlocalăčřăŶŌăĂČ ēfĚçğ■æŰžăijRăTřăyĂçijžçČăřŖŖæŶřçŽŷăřžăžŌăĚŭăžŰPythonăLăæIJřēĂNēIĂăLŰēőyăřTē;ăRēăđ'ŰēfŶæIJL'ăyĂăžŽăřTē;ČēŽ;æĜČçŽĐēČlăLēriiNăřTăēČă;fçTlăžNăL'■ēIJĂēēAēřČçTl next() řriiNăŏđēŽĚă;fçTlăŰŭēfŽăyIă■ēēld'ă;LăŏžăŶşēčŭăfŶēŏřăĂČăř;çŏăăēČă■d'riiNă■RçlNēfŶæIJL'ăĚŭăžŰçTlăđ'DriiNăřTăēČă;IJăyžăyĂăyIăĚēĂTăŽđērČăĜ;æTřçŽĐăŏž

ăēČăđIJă;ăăžĚăžĚăRlēIJĂēēAçžŽăŽđērČăĜ;æTřăijăēĂŖēčlăđ'ŰçŽĐăĂăijçŽĐērriiNēfŶæIJL'ăyĂçğ■ăpartial() çŽĐăŰžăijRăžşă;LăIJL'çTlăĂČăIJăşşăæIJL'ă;fçTl partial() çŽĐăŰŭăĂžriiNă;ăăRřēČ;çžRăyŷçIJNăLřăyNēlčēfĚçğ■ă;fçTlămbdăēălē;ăijRçŽĐăđ'■æIČăžççăAřijŽ

```

>>> apply_async(add, (2, 3), callback=lambda r: handler(r, seq))
[1] Got: 5
>>>

```

ăRřăžēăRČēĂČ7.8ăRřēLČçŽĐăĜăăyIčđ'žă;NriiNăTžă;ăăēČă;Tă;fçTl partial()ălēăŽt'ăTžăRČăTřç■ăR■ălēçŏĂăNŰăyLēfřăžççăĂăĂČ

## 9.11 7.11 áĚĚàĀĤāZđèŕĈāĜĭæŦŕ

### éŬóécŸ

ā;Šā;āçijŪāĚŽā;ĤçŦĭāZđèŕĈāĜĭæŦŕçŽĐāžççāAçŽĐæŬūāĀŽiijŊæŊĚāĤĈā;Ĺād'ŽārŔāĜĭæŦŕçŽĐæĹŦā  
ā;āāyŊæIJŽæĹ;āĹŕæšŔāyĭæŪzæŦŦæĭēēōŦ'āžççāAçIJŊāyĹāŌzæŽŦ'āĈŔæŸŕāyĀāyĭæŽōēĀŽçŽĐæĹġēāŊāžĭ

### èġĉāĒşæŪzæāĹ

éĀŽēĤĜā;ĤçŦĭçŦŦşæĹŔāŽĭāŠŊā■ŔçĹŊāŔŕāzēā;Ĥā;ŬāZđèŕĈāĜĭæŦŦŕāĒĚāĤĀĹĭæšŔāyĭāĜĭæŦŦŕāy■āĈ  
āyžāžĒæijŦçd'žèŦ'æŸŌiijŊāĀĜēō;ā;āæIJĹāçĈāyŊæĹĀçd'žçŽĐāyĀāyĭæĹġēāŊæšŔçġ■ēōāçōŬāzzāĹāçĎŬ

```
def apply_async(func, args, *, callback):
    # Compute the result
    result = func(*args)

    # Invoke the callback with the result
    callback(result)
```

æŌēāyŊæĭēēōŦ'æĹŠāžŋçIJŊāyĀāyŊāyŊéĭççŽĐāžççāĀiijŊāōĈāŊĚāŔŋāžĒāyĀāyĭ  
Async çšzāŠŊāyĀāyĭ inlined\_async èĈĚēēŕāŽĭiijŽ

```
from queue import Queue
from functools import wraps

class Async:
    def __init__(self, func, args):
        self.func = func
        self.args = args

def inlined_async(func):
    @wraps(func)
    def wrapper(*args):
        f = func(*args)
        result_queue = Queue()
        result_queue.put(None)
        while True:
            result = result_queue.get()
            try:
                a = f.send(result)
                apply_async(a.func, a.args, callback=result_queue.
→put)
            except StopIteration:
                break
        return wrapper
```

ēĤŽāyđ'āyĭāžççāAçĹĜæōŦāĒĀēōyā;āā;ĤçŦĭyieldēŦ■āŔēāĒĚāĤāZđèŕĈā■ēĭd'āĈĀŕŦāçĈiijŽ



æCædIä;æřČTÍ test () iijŇä;äiijŽă;UăLřčzäiijjæCäyNčŽDě;ŠăGžiiŽ

[illegible]

æIɲn̥ər̥ɛ̀lC̥aj̥ž̥ǎođ̥ǎođ̥ǎIɲl̥ǎIɲč̥ž̥D̥æɲ̥N̥ər̥ɲ̥ä; ä̌ä̌Ěšä̌ž̌Ō̌ǎž̌d̊ər̥č̥ǎǦ; æɲ̥ɲ̥r̥ǎǍç̣ɲ̥Ťš̌æ̌L̊r̊ǎž̊l̊ǎš̊N̊æ̌Ō̌g̊ǎL̊ǔæɲ̥Ǎç̣;  
 é̌Ǔä̌Ē̌l̊ij̊N̊ǎIɲl̊ěIɲǍěə̌Ǎä; ɛ̌ç̣ɲ̥l̊ǎl̊r̊ǎž̊d̊ər̥č̣ẓ̌Ḍä̌ẓ̌ç̣ç̣ä̌Ǎä̌y̌■ǐiǰN̊ä̌ĚšěŤǒç̣C̣ẓǎIɲl̊ä̌ž̌Ō̌ǎ; š̌ǎL̊■è̌ǒǎç̣ǒǓä̌ǔë̌ä; Iɲä̌ǐ  
 ä̌; š̌è̌ǒǎç̣ǒǓé̌G̊■ǎR̊r̊æ̌ǓǚǐiǰN̊ä̌ž̊d̊ər̥č̣ǎǦ; æɲ̥ɲ̥r̊ěc̊n̊ər̊č̣ɲ̥l̊æ̌l̊ěç̣ẓ̌g̣ç̣■ä̌d̊' D̊ç̣R̊Ěç̣ẓ̌ṣ̌æ̌d̊Iɲä̌ǍC̣ǎp̣p̣ḷy̌\_̣ǎṣy̌ṇc̣()̌  
 ä̌Ǧ; æɲ̥ɲ̥ɲ̥r̊ä̌iǰT̊ç̣d̊' ž̌ä̌ž̌Ěæ̌L̊g̊è̌ǎN̊ä̌ž̊d̊ər̥č̣ẓ̌D̊ä̌ǒd̊é̌ž̌Ěé̌Ǎž̌ěç̣; š̌ǐiǰN̊ä̌r̊ç̣; ç̣ǒǎä̌ǒd̊é̌ž̌Ěæ̌C̊Ē̊ä̌Ěɲ̊ä̌y̌■ä̌ǒč̣ǎR̊r̊ěč̣; ä̌iǰž̊æ̌ž̊t̊ä̌L̊ä̌  
 è̌ǒǎç̣ǒǓç̣ẓ̌D̊æ̌ẓ̌č̣ǎǍIɲä̌y̌Ō̌é̌G̊■ǎR̊r̊æ̌Ǎǐěǔr̊ěǔṣ̌ɲ̥Ťš̌æ̌L̊r̊ǎž̊l̊ǎǦ; æɲ̥ɲ̥ɲ̥ç̣ẓ̌D̊æ̌L̊g̊è̌ǎN̊ä̌l̊ǎä̌d̊N̊ä̌y̌■è̌r̊N̊ěǍN̊ä̌R̊L̊ä̌Ǎ  
 ä̌Ěǚä̌; š̌æ̌l̊è̌è̌ǒš̌ǐiǰN̊y̌ǐěl̊d̊æ̌ṣ̌■ä̌; Iɲä̌iǰž̊ä̌; ä̌ä̌y̌ä̌ä̌y̌ṭɲ̥Ťš̌æ̌L̊r̊ǎž̊l̊ǎǦ; æɲ̥ɲ̥ɲ̥ä̌ž̊g̣ç̣Ťš̌ä̌y̌ä̌ä̌y̌l̊ǎä̌iǰä̌ž̊ǚæ̌ẓ̌č̣ǎǍIɲä̌ǍC̣  
 æ̌Ō̌è̌ä̌y̌N̊ä̌ǐè̌ər̊č̣ɲ̥l̊ç̣Ťš̌æ̌L̊r̊ǎž̊l̊ç̣ẓ̌D̊\_\_\_\_\_next\_\_()̌æ̌L̊Ū\_\_\_\_\_send\_\_()̌  
 æ̌Ū̌ž̌æ̌ṣ̌ɲ̥ɲ̥r̊L̊ä̌iǰž̌ěǒl̊' ä̌ǒč̣ä̌ž̌Ō̌æ̌ẓ̌č̣ǎǍIɲä̌d̊' D̊ç̣ẓ̌g̣ç̣■æ̌L̊g̊è̌ǎN̊ä̌ǍC̣

```
æāzæ■ðēfZāyIæĀIeūriiŃēfZāyĀārRēLĈċZDæāyāfĈārsāIĬ      inline_async()
èĉĒēērāZlāĜ;æTrāy■āzEāĀĈ āĒšēTōċĈzāršæYriiŃēĈĒēērāZlāiŃZēĀŖæ■ēĀ■āŌĒçTšæĹŖāZlāĜ;æTŕçZDæ
yield ēr■āRēriŃNærRāyĀānqāyĀāyIāĀĈ āyžāzĒēfZēāuāAŽriiŃāĹZāiŃĀāĜŃçZDæŪūāĀZāĹZāzžāzĒāyĀā
result      ēYšāĹŪāzūāŖSēĜŃēIcæT;āĒēāyĀāyI      None      āĀiŃāĀĈ
```

çDúâRÖâijAâgNâyÄäylâ;İçÖræŞ■ä;IiijNâzÖeYşâLÜäy■âRÜâGzçzŞædIJâÄijâzüâRŞéÄAçzZçTşæLŘâZİi  
yield èr■âRërijN âIJİèZÉGÑâyÄäy Async çZDâõdä;NècñæÖëâRÜâLřāĀĆçDúâRÖâ;İçÖrâijAâgNæçÄæ  
apply\_async() ãĀĆ çDüèĀNiiijNèfZäylèõaçoÜæIJL'äylæIJĀëraâijCéCİāLĒæYřāõČāzüæşæIJL'ä;İçTİā  
put() æŰzæşTæİēāZdërČāĀĆ

èfZæUûâĀZiiijNæYřæUûâĀZèrççzEèğçéGŁäyNāLřāZTāRŚçTşāzEāzĀāzLāzEāĀĆäyza;İçÖrçñNā■şèf  
get() æŞ■ä;IJāĀĆ æÇædIJæTřæ■óā■YāIJiijNāõČäyĀāõZæYř put() āZdërČā■YæT;çZDçzŞædIJāĀĆæÇædIJæşææIJL'æTřæ■õiiijNéCčāzLāĒLæZČāAIJæŞ■ä;IJāzüç■L'ā;ĒçzŞæ  
èfZäylāEûā;ŞæĀŌæāuāõdçÖræYřçTş apply\_async() āG;æTřæİēāEşāõZçZDāĀĆ  
æÇædIJā;āäy■çZyāfaâijZæIJL'èfZāzLçèdæGçZDāzNæČErijNā;āāRřāzēā;İçTİ  
multiprocessing āzŞæİèèrTāyĀäyNiiijN âIJĪā■TçNñçZDèfZçİNāy■æLgēāNāijCæ■èèõaçoÜæŞ■ä;IiijN

```
if __name__ == '__main__':  
    import multiprocessing  
    pool = multiprocessing.Pool()  
    apply_async = pool.apply_async  
  
    # Run the test function  
    test()
```

āõdÉZĒäyLā;āaijZāRŚçÖrèfZäylçIJşçZDārşæYřèfZæāũçZDiiijNā;EæYřèçAèğçéGŁæyĒæēZāEûā;ŞçZİ  
ārEād'■æİCçZDæŌgāLūætAéZReŪRāLřçTşæLŘāZİāG;æTřèČNāRŌçZDä;Nā■ŘāIJæāGāGEāzŞāSñç  
ærTāèČiiijNāIJİ contextlib äy■çZD @contextmanager  
èçÉèērāZİā;İçTİāzEäyÄäylāzd'āzžè' zèğççZDæLĀāũgiiijN éĀZèfGäyÄäy yield  
èr■âRëârEèfZāEëāSñçzâijÄäyLäyNæŰGçõaçoRĒāZİçşYāRĪLāIJĪāyĀetūāĀĆ  
ārEād'ŰÉİdāyætAēāNçZD Twisted āNĒäy■āzşāNĒāRnāzEÉİdāyçşzâijijçZDāEĒèĀTāZdërČāĀĆ

## 9.12 7.12 èõŁéUõéU■āNĒäy■āõZāzL'çZDāRŸéGR

### éUõéçY

ā;āæČşèçAæL'l'āsTāG;æTřäy■çZDæşRäyléU■āNĒiiijNāEæðyāõČèČ;èõŁéUõāSñāŁæTzāG;æTřçZDā

### èğçāEşæŰzæāL

éĀZāyÿæİèèõşiiijNéU■āNĒçZDāEĒéCİāRŸéGRārzažŌād'ŰçTNæİèèõşæYřāõNāĒÉéZReŪRçZDāĀĆ  
ā;EæYřiiijNā;āāRřāzēēĀZèfGçijŰāEZeõŁéUõāG;æTřāzüārEāĒūā;IJāyzaG;æTřāsdæĀğçzSāõZāLřéU■āNĒäy

```
def sample():  
    n = 0  
    # Closure function  
    def func():  
        print('n=', n)  
  
    # Accessor methods for n  
    def get_n():  
        return n
```

```
def set_n(value):
    nonlocal n
    n = value

# Attach as function attributes
func.get_n = get_n
func.set_n = set_n
return func
```

äyÑéÍæÝřä;ŁçŤÍçŽDä;Ňă■Ř:

```
>>> f = sample()
>>> f()
n= 0
>>> f.set_n(10)
>>> f()
n= 10
>>> f.get_n()
10
>>>
```

## èóìèőž

äyžäZÈrt' æÝŎæyĚæěŽăóČăÇCă;Ťăüěä;ĬçŽDñijŇæĬJL'äyd'çĆzéĬJĚèĚAèğcéĜLäyĂäyŇăĂĆéĚŮăĚĹij  
 ăčřæÝŎăŔřäzèèŏl' æĹSăznçijŮăĚŽăG;æŤřæĬèăŁăŤzăĚĚčĹăŔŸéĜŔçŽDăĬijaĂĆ  
 ăĚŮăñajijŇăĜ;æŤřăśđæĂğăĚAèőyæĹSăznçŤĹăyĂçğ■ă;ĹçőĂă■ŤçŽDæŮzăijŔăŕĚèőĚéŮŏæŮzăşŤçzŚăőŽăĹ  
 èŁŸăŔřäzèèŁŽăyĂæ■èçŽDæĹl'ăśŤijŇèŏl' éŮ■ăŇĚăĹæŇşçşzçŽDăŏđă;ŇăĂĆă;ăèĚAăĂŽçŽDăzĚăzĚă

```
import sys
class ClosureInstance:
    def __init__(self, locals=None):
        if locals is None:
            locals = sys._getframe(1).f_locals

        # Update instance dictionary with callables
        self.__dict__.update((key,value) for key, value in locals.
→items()

                                if callable(value) )

        # Redirect special methods
    def __len__(self):
        return self.__dict__['__len__']()

# Example use
def Stack():
    items = []
    def push(item):
        items.append(item)
```

```

def pop():
    return items.pop()

def __len__():
    return len(items)

return ClosureInstance()

```

äYÑéÍæYřäYÄäyläžd' äžŠäijRäijŽerÍæIëæijTçd'žãŃæYřæĈä;Tåũëä;IJçŽDriijŽ

```

>>> s = Stack()
>>> s
<__main__.ClosureInstance object at 0x10069ed10>
>>> s.push(10)
>>> s.push(20)
>>> s.push('Hello')
>>> len(s)
3
>>> s.pop()
'Hello'
>>> s.pop()
20
>>> s.pop()
10
>>>

```

æIJL'ëũčçŽDæYřriijÑeĤŽäyläzčçäAeĤŘeaÑetũæIëæijŽæřTäyÄäylæŽöéÄŽçŽDçšãŃæYřæL'èeAãĤná;ÍLäd'

```

class Stack2:
    def __init__(self):
        self.items = []

    def push(self, item):
        self.items.append(item)

    def pop(self):
        return self.items.pop()

    def __len__(self):
        return len(self.items)

```

æĈædIJeĤŽæũũAŽriijNä;ääijŽä;UåŁřçšzäijjæĈäyNçŽDçzŠædIJriijŽ

```

>>> from timeit import timeit
>>> # Test involving closures
>>> s = Stack()
>>> timeit('s.push(1);s.pop()', 'from __main__ import s')
0.9874754269840196
>>> # Test involving a class
>>> s = Stack2()

```

```
>>> timeit('s.push(1);s.pop()', 'from __main__ import s')
1.0707052160287276
>>>
```

çzŞæđIæÿçd' ziiŋNéŮ■āNĖÇŽDæŮzæāLèĤRèqNèŭæIèèĕAāĤnād' gæĕC8%iiŋNād' gēĆlāLEāŌšāZāæÿ  
éŮ■āNĖæŽt' āĤnæÿrāZāäyžäy■aijŽæŭL' āRĹāĹrēclād' ŮÇŽDselfāRŸéĠRāĀĆ

Raymond HettingerārřzāžŌèĤZäyĹéŮŏécÿèŏçāāGžāžEæŽt' āĹæéŽçäžĕçRĖèġççŽDæŤžèĤZæŮzæāLāĀ  
èĀNäyŤāŏĈāRĹæÿřçIJšāŏđçšçŽDäyĀäyĹæĖGæĀĤçŽDæŽĤæ■çèĀNāŭŝiiŋNäçNāĕĈiiŋNçšççŽDäyžèĕAçL' zæA  
āzŭāyŤajāĕĕAāZäyĀāzZāĖŭāzŮÇŽDāŭĕā;IJæL'■ĕĈçèŏl' äyĀāžŽçL' zæŏLæŮzæšŤçŤšæŤĹ(æŕŤæĈäyĹéĬ  
ClosureInstance äy■ĕG■āĖŽĕĤĠçŽD \_\_len\_\_() āŏđçŌřāĀĆ)

æIJĀāRŌiiŋNä;āāRĤĕĈçĕĤÿaijŽèŏl' āĖŭāzŮĕÿĖĕrřzajāāžĕçāAçŽDāžžæĎšāĹřçŮšæĈŝiiŋNäyžāzĀāzĹāŏ  
(āçŞçĎŭiiŋNāzŮāznāzšæĈşçšĕĕAşäyžāzĀāzĹāŏĈĕĖĤRèqNèŭæIèäijŽæŽt' āĤn)āĀĈārççŏāĕĈæ■d' iiŋNèĤZārřzā

æĀžāçŞäyĹèŏŝiiŋNāIJĹéĖ■çççŽDæŮŭāĀZçzŽĕŮ■āNĖæŭzāĹāæŮzæšŤaijŽæIJL' æŽt' āđ' ŽçŽDāŏđçŤĹāĹ  
æŕŤæĈĈäçāĕĖIJĀĕĕAĕĠççç;ŏāĖĖĕĈĬçĹŭæĀĀāĀĀāĹŭæŮřçijŞāĖšāNžāĀĀæyĖĖéŽd' çijŞā■ÿæĹŮāĖŭāzŮÇŽDāR

## 10 çññāĖñçñäiiŋŽççszäyŌāržèśą

æIJñçñääyžèĕAāĖšæšĬçççŽDæÿrāŖNçşzāŏŽāzĹ' æIJL' āĖšççŽDäyŷĖġAçijŮçĬNæĹāđNāĀĈāNĖæNñèŏl'  
çşzārĀĕĈĖæĹĀæIJrāĀAçzğæL' ĤāĀĀāĖĖā■ÿçŏāçRĖäžĕāRĹæIJL' çŤĬççŽDèŏçèŏāæĹāaijRāĀĆ

Contents:

### 10.1 8.1 æŤžāRŸāržèśąççŽDā■Ůçñĕäyşæÿççd'ž

#### éŮŏécÿ

äjāæĈşæŤžāRŸāržèśąāŏđäçĹççŽDæL' şā■ræĹŮæÿççd' žĕçşāGžiiŋNèŏl' āŏĈāznæŽt' āĖŭāRĤĕrřzæĀğāĀĆ

#### èġĈāĖşæŮzæāĹ

èĕAæŤžāRŸäyĀäyĹāŏđäçĹççŽDā■ŮçñĕäyşæāĬçd' ziiŋNāRĤĕĠæŮŕāŏŽāzĹ' āŏĈççŽD  
\_\_str\_\_() āŖN \_\_repr\_\_() æŮzæšŤāĀĈäçNāĕĈiiŋŽ

```
class Pair:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return 'Pair({0.x!r}, {0.y!r})'.format(self)

    def __str__(self):
        return '({0.x!s}, {0.y!s})'.format(self)
```

```
>>> p = Pair(3, 4)
>>> p
Pair(3, 4) # __repr__() output
>>> print(p)
(3, 4) # __str__() output
>>>
```

```
>>> p = Pair(3, 4)
>>> print('p is {0!r}'.format(p))
p is Pair(3, 4)
>>> print('p is {0}'.format(p))
p is (3, 4)
>>>
```

```
>>> f = open('file.dat')
>>> f
<_io.TextIOWrapper name='file.dat' mode='r' encoding='UTF-8'>
>>>
```

```
def __repr__(self):
    return 'Pair({0.x!r}, {0.y!r})'.format(self)
```

ä;IjäyžèŁŻç■āōđčŔřŽĎäyÄäytæZfäzčüjNä;ääššāRräžēä;ŁçTÍ %  
æS■ä;IJčņēüijNārśāČŘäyNéÍcèŁZæāüüijŽ

```
def __repr__(self):
    return 'Pair(%r, %r)' % (self.x, self.y)
```

## 10.2 8.2 èĜłăŏŽăzL'ă■ŮčņęäÿŝçŽĎæăĭăĭjŘăŇŮ

### éŮŏécŸ

ăĭăæČŝéĂŽèĤĜ format() äĜĭæŤřăŠŇă■ŮčņęäÿŝæŮzæŝŤăĤăĤ; ŮăÿĂăÿłăřzèsăèČĭæŤřăŇŮăĜłăŏŽăzL'

### èĝčăĚŝæŮzæăĹ

ăÿžăŽĚèĜłăŏŽăzL'ă■ŮčņęäÿŝçŽĎæăĭăĭjŘăŇŮĭĭŇăĹŝăžŇéĬĂèĕĂăĬĬŝŝăÿĹéĬăŏŽăzL'  
\_\_format\_\_() æŮzæŝŤăĂčăĹŇăĕĈĭĭŽ

```
_formats = {
    'ymd' : '{d.year}-{d.month}-{d.day}',
    'mdy' : '{d.month}/{d.day}/{d.year}',
    'dmy' : '{d.day}/{d.month}/{d.year}'
}

class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    def __format__(self, code):
        if code == '':
            code = 'ymd'
        fmt = _formats[code]
        return fmt.format(d=self)
```

çŮřăĬĬ Date çŝžçŽĎăŏđăĤŇăŘřăzèæŤřăŇŮăæăĭăĭjŘăŇŮæŝ■ăĭĬăžĚĭĭŇăĕČăŘŇăÿŇéĬčèĤŽăăĭĭjŽ

```
>>> d = Date(2012, 12, 21)
>>> format(d)
'2012-12-21'
>>> format(d, 'mdy')
'12/21/2012'
>>> 'The date is {:ymd}'.format(d)
'The date is 2012-12-21'
>>> 'The date is {:mdy}'.format(d)
'The date is 12/21/2012'
>>>
```

## èõìèõž

`__format__()` æŰzæşŦçzŽPythonçŽĐā■ŰçñęäÿşæäijäijRāŃŰāŁşèČ;æŦŦä;ŽāžEäÿÄäÿłéŠł'ā■RāÄ  
èłŽéĜŇéIJĀèēAçĬĀéĜ■āijžèŦČçŽĐæŸŦæäijäijRāŃŰāžčçāAçŽĐèğčæđŦāũēä;IJāōŃāĒłçŦšçşžèĜłāũśāEşşāōž  
ä;ŇāēČiijŇāŦČèÄČäÿŇéłçæłèèĜł `datetime` æłāāłŰäÿ■çŽĐāžčçāAäijŽ

```
>>> from datetime import date
>>> d = date(2012, 12, 21)
>>> format(d)
'2012-12-21'
>>> format(d, '%A, %B %d, %Y')
'Friday, December 21, 2012'
>>> 'The end is {: %d %b %Y}. Goodbye'.format(d)
'The end is 21 Dec 2012. Goodbye'
>>>
```

āržāžŌāEĚç;őçşşāđŇçŽĐæäijäijRāŃŰæIJLäÿÄāžZæāĜāĜEçŽĐçžæāōŽāÄČ  
āŦŦāžēāŦČèÄČ `string`æłāāłŰæŰĜæaç èŦ' æŸŌāÄČ

## 10.3 8.3 èõł'āržèşşæŦŦŦæŇÄäÿŁäÿŇæŰĜçóaçŦEā■Ŧèõõ

### éŰóécŸ

ä;äæČşèõł'ä;äçŽĐāržèşşæŦŦŦæŇÄäÿŁäÿŇæŰĜçóaçŦEā■Ŧèõõ(with èŦ'āŦŦē)āÄČ

### èğčāEşşæŰzæāł

äÿžāžEèõł'äÿÄäÿłāržèşşāĒijāōž with èŦ'āŦŦēijŇä;äéIJĀèēAāōđçŌŦ `__enter__()`  
āšŇ `__exit__()` æŰzæşŦāÄČ ä;ŇāēČiijŇèÄČèŽşæČäÿŇçŽĐäÿÄäÿłçşşijŇāōČèČ;äÿžæŁşāžŇāŁZāžžäÿ

```
from socket import socket, AF_INET, SOCK_STREAM

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = family
        self.type = type
        self.sock = None

    def __enter__(self):
        if self.sock is not None:
            raise RuntimeError('Already connected')
        self.sock = socket(self.family, self.type)
        self.sock.connect(self.address)
        return self.sock

    def __exit__(self, exc_ty, exc_val, tb):
```



```
self.sock.close()
self.sock = None
```

èŁŻäÿłçşzçŽĐăĖşéŤōçŁżçĆzăĬJlăžŌăōĈeăłçđ'žăžĖăÿĂăÿłç;ŚçzĬJèŁđăŌëĭĭŃăĭĖăŸrăĬlăgŃăŃŮçŽĐă  
èŁđăŌëçŽĐăžžçŃŃăŖŃăĖşéŮăăŸrăĭçŤĬ with èŕăăŖëèĠăĬlăŏŃăĬŖçŽĐĭĭŃăĭŃăçĈĭĭŽ

```
from functools import partial

conn = LazyConnection(('www.python.org', 80))
# Connection closed
with conn as s:
    # conn.__enter__() executes: connection open
    s.send(b'GET /index.html HTTP/1.0\r\n')
    s.send(b'Host: www.python.org\r\n')
    s.send(b'\r\n')
    resp = b''.join(iter(partial(s.recv, 8192), b''))
    # conn.__exit__() executes: connection closed
```

## èóíèőž

çĭĭŮăĖŽăÿĬăÿŃăŮĠçŏăçŖĖăŽĬçŽĐăÿzèĖĂăŌşçŖĖăŸrăĭçŽĐăžççăĂăĭĭŽăŤĭăĬŕ  
with èŕăăŖëăĬŮăÿăăĬĖăŃăĂĈăĭŖăĠçŏŕ with èŕăăŖëçŽĐăŮăăĂŽĭĭŃăŕžèşăçŽĐ  
\_\_enter\_\_() æŮžæşŤèçŃèĝĖăŖŖĭĭŃăŏŐĈēŤăŽđçŽĐăĂĭĭ(ăĖĈăđĬăĬĬçŽĐēŕĬ)ăĭĭŽèçŃèŤŃăĂĭĭçžŽ  
as âçŕăŸŌçŽĐăŖŸéĠŖăĂĈçĐăăŖŌĭĭŃwith èŕăăŖëăĬŮëĠççŽĐăžççăĂăĭĭĂăĝŃăĬĖăŃăĂĈ  
ăĬĂăŖŌĭĭŃ\_\_exit\_\_() æŮžæşŤèçŃèĝĖăŖŖŖçēŤŽăŃăÿĖçŖĖăăĭĭĂăĂĈ

ăÿŃçŏă with äžççăĂăĬŮăÿăăŖŖŖŤşăžĂăžĬĭĭŃăÿĬëĬççŽĐăŌĝăĬŮăŤĂëĈĭăĭĭŽăĬĖăŃăŏŃĭĭŃăŕşçŏŮă  
ăžŃăŏđăÿĬĭĭŃ\_\_exit\_\_() æŮžæşŤçŽĐçŃăÿĬăŷăŖĈăŤŕăŃĖăŖŃăžĖăĭĭĈăÿÿçşăđŃăĂăăĭĭĈăÿÿăĂĭăŖ  
\_\_exit\_\_() æŮžæşŤèçĭèĠăŮşăĖşăŏžăĂŌăăăăĬŖçŤĬèŁŻăÿłăĭĭĈăÿÿăŤăăĂŖĭĭŃăĬŮëĂĖăŤçŤèăŏĈăžŮă  
ăĖĈăđĬ\_\_exit\_\_() èŁŤăŽđ True ĭĭŃëçĈăžĬăĭĭĈăÿÿăĭĭŽèçŃăÿĖçŖ'žĭĭŃăŕşăĖăĈŖăžĂăžĬëĈĭăşăăŖŖŖŤ  
with èŕăăŖëăŖŌëĬççŽĐçĬŃăžŖççççăĬĬăăăăÿăĬĖăŃăĂĈ

èŁŸăĬĬăÿĂăÿłçžĖĖĬçĖŮëçŸăŕşăŸŕ LazyConnection  
çşşăŸŕăŖăĖĖăĖŏÿăđ'Žăÿł with èŕăăŖëăĬăŤŃăăŮăĭçŤĬèŁđăŌëăĂĈ  
ăĬĬăŸçĐăĭĭŃăÿĬëĬççŽĐăŏžăžĬăÿăăŸăăŃăăŖĖçĭăĖăĖŏÿăÿĂăÿłsocketèŁđăŌëĭĭŃăĖĈăđĬăăăăĬĬăĭçç  
with èŕăăŖëĭĭŃăŕşăĭĭŽăžĝçŤşăÿĂăÿłăĭĭĈăÿÿăžĖăĂĈăÿăĖĠăăŖŕăžăăŖăÿŃëĬççŽăăăăŤăžăŃăÿĬë

```
from socket import socket, AF_INET, SOCK_STREAM

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = family
        self.type = type
        self.connections = []

    def __enter__(self):
        sock = socket(self.family, self.type)
        sock.connect(self.address)
```

```

self.connections.append(sock)
return sock

def __exit__(self, exc_ty, exc_val, tb):
    self.connections.pop().close()

# Example use
from functools import partial

conn = LazyConnection(('www.python.org', 80))
with conn as s1:
    pass
    with conn as s2:
        pass
    # s1 and s2 are independent sockets

```

aIÍlçññāzNāyłçL'LæIJñāy■iijNLazyConnectionçşzāRřāzēēcñçIJNāAžæYřæšRāyłēfđæŌēāuēāŌĆā  
 æRřæñā\_\_enter\_\_() æŰžæşTæL'gēāNçŽDæŰūāĀŽiijNāōČād'■āLūāLZāzžāyĀāyłæŰřçŽDēfđæŌēāzūā  
 \_\_exit\_\_() æŰžæşTçōĀā■TçŽDāzŌæāLāy■āiijāGžæIJĀāRŌāyĀāyłēfđæŌēāzūāĒşēŰ■āōČāĀĆ  
 èfŽēGŇčÍ■āļōæIJL'çČzéŽçRĒēğçiiijNāy■ēfGāōČēČ;āĒĀēōyāŦNāēŰā;ŁçTÍ with  
 èr■āRēāLZāzžād'ŽāyłēfđæŌēiijNāřsāēCāyLēlČæiijTçd'žçŽDēČcæāuāĀĆ

aIÍléIJĀēēAçōaçRĒāyĀāžŽēŦDæžRærTāēČæŰGāzūāĀAç;ŠçzIJēfđæŌēāSŇēTĀçŽDçijŰčlNçŌřācČāy■  
 èfŽāžŽēŦDæžRçŽDāyĀāyłāyžēēAçL'žā;AæYřāōČāzñāfĒēāzēcñæL'NāLlçŽDāĒşēŰ■āLŰēGLæTç;ælēçāōāf  
 ā;NāēČiijNāēČādIJā;āērūāēšCāžEāyĀāyłēTĀiijNēČčāzLā;āāfĒēāzçāōāfĪāzNāRŌēGLæTç;āžEāōČiijNāRēāL  
 éĀŽēfGāōđçŌř \_\_enter\_\_() āSŇ \_\_exit\_\_() æŰžæşTāzūā;ŁçTÍ with  
 èr■āRēāRřāzēēā;LāōžæYŠçŽDēAŁāĒēfŽāžŽēŰōēčYiijN āŽāāyž \_\_exit\_\_()  
 æŰžæşTĀRřāzēēōl'ā;āēŰāēIJĀæNĒāfČēfŽāžŽāžEāĀĆ

aIÍlcontextmanager ælāālŰāy■æIJL'āyĀāyłæāGāGĒçŽDāyLāyNæŰGçōaçRĒæŰžæāLælāæĪēiijNā  
 āRŇæŰūāIJl12.6ārRēLČāy■ēfYæIJL'āyĀāyłāřzæIJñēLČçd'žā;NçlNāžRçŽDçžŁçlNāōL'āĒlçŽDāfōæTžçL'L

## 10.4 8.4 āLZāzžād'gēGRāržēsāæŰūēLČçlJAāĒĒā■YæŰžæşT

### éŰōēčY

ā;āçŽDçlNāžRēēAāLZāzžād'gēGR(āRřēČ;āyŁçŽç;āyĠ)çŽDāržēsāiijNāřijēGr'ā■āçTlā;Lād'gçŽDāĒēĒā■

### ēğçāĒşæŰžæāL

āřzāžŌāyžēēAæYřçTlāēā;šæL'RçōĀā■TçŽDæTřæ■ōçzšæđDçŽDçşzēĀŇēlĀiijNā;āāRřāzēēĀŽēfGçž  
 \_\_slots\_\_ āşđæĀğælēæđĀād'gçŽDāGRārŠāōđā;NæL'Āā■āçŽDāĒēĒā■YāĀĆærTāēČiijZ

```

class Date:
    __slots__ = ['year', 'month', 'day']
    def __init__(self, year, month, day):
        self.year = year

```

```
self.month = month
self.day = day
```

ā;Šā;āāōŽāzL' \_\_slots\_\_ āRŌiijNPythonāršaijŽāyžāōđā;Nā;ŁçTlāyĀçg■æŽt' āŁāçt' gāGŚçŽDāEĒēČ  
āōđā;NēĀŽēŁGāyĀāyĪā;ŁārRçŽDāZžāōŽād' gārRçŽDæTřçzDæĪēæđDāzziiNēĀNāy■æYřāyžæfRāyĪāōđā;N  
āIJĪ \_\_slots\_\_ äy■āLŪāGžçŽDāsđæĀgāR■āIJĪāEĒēČĪēčnāYāārDāŁřēŁŽāyĪæTřçzDçŽDæNĠāōŽārRæāČ  
ā;ŁçTlslotsāyĀāyĪāy■āē;çŽDāĪJřæŪzāršæYřæŁSāznāy■ēČ;āE■çzŽāōđā;NæūzāŁāæŪřçŽDāsđæĀgāžEiijNā  
\_\_slots\_\_ äy■āōŽāzL'çŽDēČčāžŽāsđæĀgāR■āĀČ

## ēōĪēōŽ

ā;ŁçTlslotsāRŌēŁČçIJĀçŽDāEĒā■YāijŽēūšā■YāČĪāsđæĀgçŽDæTřēGRāŠNçszādNāIJL'āEšāĀČ  
äy■ēŁGiijNāyĀēŁNāēĪēēōšiiNā;ŁçTlāŁřçŽDāEĒā■YæĀzéGRāŠNārEæTřæ■ōā■YāČĪāIJāyĀāyĪāēČçzDāy■  
āyžāžEçzŽā;āāyĀāyĪçŽt' ēgČēōđ' ēfEiijNāĀGēō;ā;āāy■ā;ŁçTlslotsçŽt' æŌēā■YāČĪāyĀāyĪDateāōđā;NiiN  
āIJĪ64ā;■çŽDPythonāyŁēĪēēĀā■āçTl428ā■ŪēŁČiijNēĀNāēČæđIJā;ŁçTlāžEslotsiiNāEĒā■Yā■āçTlāyNēŽ■  
āēČæđIJçĪNāžRāy■ēIJĀēēĀāRŊæŪūāŁZāžžād' gēGRçŽDæŪēāIJšāōđā;NiiNēČčāžŁēŁŽāyĪāršēČ;æđĀād' g

ār;çōāslotsçIJNāyŁāŌžæYřāyĀāyĪā;ŁæIJL'çTlçŽDçL'žæĀgiijNā;Łād' ŽæŪūāĀŽā;āēŁYæYřā;ŪāGRārš  
PythonçŽDā;Łād' ŽçL'žæĀgēČ;ā;ĪēŁŪāžŌæŽōēĀŽçŽDāšžāžŌā■ŪāEyçŽDāōđçŌřāĀČ  
ārēād' ŪriijNāōŽāzL'āžEslotsāRŌçŽDçszāy■āE■æTřæNāāyĀāžŽæŽōēĀŽçszçL'žæĀgāžEiijNārTāēČād' Žçz  
ād' gād' ŽæTřæČēĀEĪāyNiiNā;āāžTērēāRĪāIJĪēČčāžŽçzRāyŷēčnā;ŁçTlāŁřçŽDçTlā;IJæTřæ■ōçzŠæđDçŽDçs  
(ærTāēČāIJĪçĪNāžRāy■ēIJĀēēĀāŁZāžžæšRāyĪçszçŽDāGāçŽ;āyGāyĪāōđā;Nāržēsā)āĀČ

āEšāžŌ \_\_slots\_\_ çŽDāyĀāyĪāyŷēgĀēřrāNžæYřāōČāRřāzēā;IJāyžāyĀāyĪārĀēēĒāūēāĒūāĪēēYšæ■  
ār;çōāā;ŁçTlslotsāRřāzēē;ā;āŁřēŁŽæāūçŽDçŽōçŽDiiNā;EæYřēŁŽāyĪāzūāy■æYřāōČçŽDāĪēāūāĀČ  
\_\_slots\_\_ æŽt' ād' ŽçŽDæYřçTlāēĪā;IJāyžāyĀāyĪāEĒā■YāijYāNŪāūēāĒūāĀČ

## 10.5 8.5 āIJĪçszāy■ārĀēčĒāsđæĀgāR■

### ēŪōēčY

ā;āæČšārĀēčĒçszçŽDāōđā;NāyŁēĪççŽDāĀIJçgĀæIJL'āĀĪæTřæ■ōiijNā;EæYřPythonēr■ēĪĀāžūæšāæIJL

### ēgčāEšæŪžæāŁ

PythonçĪNāžRāŠYāy■āŌžā;ĪēŁŪēr■ēĪĀçL'žæĀgāŌžārĀēčĒæTřæ■ōiijNēĀNāYřēĀŽēŁGēĀĪā;ĪāyĀāōŽ  
çñnāyĀāyĪçžēāōŽæYřāžžā;Tāžēā■TāyNāŁšçžŁ\_āijĀād' t'çŽDāR■ā■ŪēČ;āžTērēæYřāEĒēČĪāōđçŌřāĀČærTā

```
class A:
    def __init__(self):
        self._internal = 0 # An internal attribute
        self.public = 1 # A public attribute

    def public_method(self):
        '''
        A public method
```

```
'''
pass

def __internal_method(self):
    pass
```

Pythonázúäy■äijŽçIJšçŽĐēYzæ■cālŇāžžēōēUōāEēČlāŘ■çğrāĀĆä;EæYřæĆæđIJä;æēŽāzĹāAŽēĆr  
āŘŇæUūēēYēēAæšlæĐRāĹŕiijŇä;ēçTīāyŇāĹŠçžēāijĀād't'çŽĐçžēāōŽāŘŇæūēēĀĆçTīāzŌāēlāāiUāŘ■āŠŇæ  
ä;ŇāēČiijŇāēĆæđIJä;āçIJŇāĹræšRāylælāāiUāŘ■āzēā■TāyŇāĹŠçžēāijĀād't'(æŕTāēĆ\_socket)riijŇēČčāōČār  
çszāijijçŽĐriijŇāēlāāiUçžgāĹnāĠ;æTŕæŕTāēĆ sys.\_\_getframe()  
āIJlā;ēçTīçŽĐæUūāĀZārśā;UāĹāāĀ■ārRāēČāzEāĀĆ

ä;æēYāRrēČ;äijŽēAĠāĹrāIJlçszāōŽāzĹāy■ä;ēçTīāyđ'āylāyŇāĹŠçžē(\_\_\_\_)āijĀād't'çŽĐāŠ;āŘ■āĀĆæŕTāē

```
class B:
    def __init__(self):
        self.__private = 0

    def __private_method(self):
        pass

    def public_method(self):
        pass
        self.__private_method()
```

ä;ēçTīāŘŇāyŇāĹŠçžēāijĀāgŇāijŽārījēĠŕ'ēōēēUōāŘ■çğrāRŸæĹRāĒūāzŪā;çāijRāĀĆ  
æŕTāēČiijŇāIJlāĹ■ēlççŽĐçszBāy■riijŇçgAæIJL'āsđæĀgāijŽēcŇāĹēāĹŇēĠ■āŠ;āŘ■āyž  
\_B\_\_private āŠŇ \_B\_\_private\_method āĀĆ ēēZæUūāĀZā;āāRrēČ;äijŽēUōēēZæūēēĠ■āŠ;āŘ■çŽĐ

```
class C(B):
    def __init__(self):
        super().__init__()
        self.__private = 1 # Does not override B.__private

    # Does not override B.__private_method()
    def __private_method(self):
        pass
```

ēēZēĠŇriijŇçgAæIJL'āŘ■çğŕ \_\_private āŠŇ \_\_private\_method  
ēcŇēĠ■āŠ;āŘ■āyž \_C\_\_private āŠŇ \_C\_\_private\_method  
riijŇēēZāyĽēușçĹūçszBāy■çŽĐāŘ■çğŕæYřāōŇāĒlāy■āŘŇçŽĐāĀĆ

## ēōlēōž

āyĹēlāēRŘāĹræIJL'āyđ'çg■āy■āŘŇçŽĐçijŪçāAçžēāōŽ(ā■TāyŇāĹŠçžēāŠŇāŘŇāyŇāĹŠçžē)ælēāŠ;āŘ  
ād'gād'ZæTŕēĀŇēlĀriijŇä;āāzTērēēōl'ä;āçŽĐēlāēĒāēSāŘ■çğŕāzēā■TāyŇāĹŠçžēāijĀād't'āĀĆä;EæYřriijŇāē  
āzūāyTāIJL'āžZāēēČlāśđæĀgāžTērēāIJlā■Rçszāy■ēŽRēŪRēŭælēriijŇēČčāzĹæĹ■ēĀČēZŠä;ēçTīāŘŇāyŇā

ēēYæIJL'āyĀçĆžēēAæšlæĐRçŽĐæYřriijŇæIJL'æUūāĀZā;āāōŽāzĹ'çŽĐāyĀāylāRŸēĠRāŠŇæšRāylāēlç

```
lambda_ = 2.0 # Trailing _ to avoid clash with lambda keyword
```

è£ŽéĜÑæĹŚāznāzūāy■ā;£çŦlā■ŦāyŦāĹŚçž£āĹ■çijĀçŽĎāŌšāZāæŸřāŏČéA£āĚ■érègčāŏČçŽĎā;£çŦlā  
(ā£Čā;£çŦlā■ŦāyŦāĹŚçž£āĹ■çijĀçŽĎçŽŏçŽĎæŸřāyžāžEéŸšæ■čāŚ;āŔ■āEšçĹAèĀŦāy■æŸřæŦĜæŸŌè£Ž  
éĀŽè£Ĝā;£çŦlā■ŦāyŦāĹŚçž£āŔŌçijĀāŔřāžèègčāEšè£ŽāyĹéŮŏécŸāĀĆ

## 10.6 8.6 āĹZāžžāŔřčŏaçŔĚçŽĎāśđæĀğ

éŮŏécŸ

ā;āæČšçžZæšŔāyĹāŏđā;ŦattributeāčđāĹæ£Ž'èŏ£éŮŏāyŌā£ŏæŦžāžŦād'ŮçŽĎāĚūāžŮād'ĐçŔĚéĀžè;Ś

ègčāEšæŮžæāĹ

èĜĹāŏŽāžĹæšŔāyĹāśđæĀğçŽĎāyĀçg■čŏĀā■ŦæŮžæšŦæŸřāŔĚāŏČāŏŽāžĹāyžāyĀāyĹpropertyāĀĆ  
ā;ŦāēČiijŦāyŦāĹēĹčçŽĎāžčçāĀāŏŽāžĹāžĚāyĀāyĹpropertyiijŦāčđāĹāāŔžāyĀāyĹāśđæĀğçŏĀā■ŦçŽĎçšžādŦāē

```
class Person:
    def __init__(self, first_name):
        self.first_name = first_name

    # Getter function
    @property
    def first_name(self):
        return self._first_name

    # Setter function
    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

    # Deleter function (optional)
    @first_name.deleter
    def first_name(self):
        raise AttributeError("Can't delete attribute")
```

āyĹè£řāžčçāĀāy■æIJĹ'āyĹ'āyĹçŽyāĚšèĀŦçŽĎæŮžæšŦiijŦè£ŽāyĹ'āyĹæŮžæšŦçŽĎāŔ■ā■ŮéČ;ā£Ěéāžāy  
çŦŦāyĀāyĹæŮžæšŦæŸřāyĀāyĹ getter āĜ;æŦŕiijŦāŏČā;£ā;Ů first\_name  
æĹŔāyžāyĀāyĹāśđæĀğāĀĆ āĚūāžŮāyđ'āyĹæŮžæšŦçžŽ first\_name āśđæĀğæŮžāĹāāžĚ  
setter āŦŦ deleter āĜ;æŦŕāĀĆ éIJĀèēĀāijžèŔČçŽĎæŸřāŔĹæIJĹāIJĹ first\_name  
āśđæĀğèčŦāĹZāžžāŔŌiijŦ āŔŌéĹčçŽĎāyđ'āyĹèçĚēēŔāŽĹ @first\_name.setter āŦŦ  
@first\_name.deleter æĹ■èČ;èčŦāŏŽāžĹāĀĆ

propertyçŽĎāyĀāyĹāĚšéŦŏçĹ'žā;AæŸřāŏČçIJŦāyĹāŌžèùšæŽŏéĀŽçŽĎattributeāšāžĀāžĹāyđ'æāūiijŦ  
ā;ĚæŸŔèŏ£éŮŏāŏČçŽĎæŮūāĀŽāijŽèĜĹāĹéğçāŔŚ getter āĀĀsetter āŦŦ deleter  
æŮžæšŦĀĀĆā;ŦāēČiijŽ

[illegible]

## ěőłěőž

äyÄäyłpropertyåsdæĀġăĔüăôđārsæYřäyĀçşzâĹŮçŽyăĔşçzŚăôŽæŮzæşŤçŽĐéŽĚăŘĹăĀĆăĕCăđIJăăă  
årśäijŽăŘŚçŎřpropertyæIJñěžñçŽĐfgetăĀĀfsetăŠŇfdelăsdæĀġăřsæYřçşzéĠŇéíççŽĐæŽóéĀŽæŮzæşŤăĀĆ

```
>>> Person.first_name.fget
<function Person.first_name at 0x1006a60e0>
>>> Person.first_name.fset
<function Person.first_name at 0x1006a6170>
>>> Person.first_name.fdel
<function Person.first_name at 0x1006a62e0>
>>>
```

éĀŽăyŷæĬěëőšüijŇăjăäy■äijŽçŽŤ æŎěăŘŮěřČçŤĬfgetăĹŮěĀĔfsetüijŇăôČăžňäijŽăIJĬěőĕĕŮőpropertyçŽ  
ăŘĹæIJĹă;ŞăjăçăőăôđéIJĀđĕĀăřzattributeæĹġĕăŇăĔüăžŮéćĬăđ' ŮçŽĐæŞ■ăjIJçŽĐæŮüăĀŽæĹ■ăžŤĕřĕ  
æIJĹæŮüăĀŽăyĀăžŽăžŎăĔüăžŮçijŮçĬŇĕř■ĕĬĀ(æřŤăĕĆJava)ĕĕĠæĬĕçŽĐçĬŇăžŘăŚŸæĀžĕôđ' äyžæĹĀæIJĹ  
æĹĀăžĕăžŮăžñĕôđ' äyžăžççăĀăžŤĕřĕăĈŘăyŇéĬĕĕĹZæăüăĔĚzrijŽ

```
class Person:
    def __init__(self, first_name):
        self.first_name = first_name

    @property
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        self._first_name = value
```

äy■ĕĕĀăĔŽĕĹŽçġ■æşææIJĹăĀŽăžzăjŤăĔüăžŮéćĬăđ' ŮæŞ■ăjIJçŽĐpropertyăĀĆ  
ĕĕŮăĔĬüijŇăôČăijŽĕôĬ'ăjăçŽĐăžççăĀăŘŸăjŮăĹĕĠĈĕĈĕĬüijŇăžüăyŤĕĹŸăijŽĕĕŮăĈŚĕŸĔĕřzĕĀĔăĀĆ  
ăĔüăñăüijŇăôČĕĹŸăijŽĕôĬ'ăjăçŽĐçĬŇăžŘĕĹĔĕăŇĕĕŮăĬĕăŘŸæĔĕăĹăđ'ŽăĀĆ  
æIJĀăŘŎüijŇĕĹŽăăüçŽĐĕőĕĕőăăžüæşææIJĹăyĕæĬĕăžzăjŤçŽĐăĕjăđ'ĐăĀĆ  
çĹzăĹŇæŸřăjŞăjăăžĕăŘŎăĈşçžŽæŽóéĀŽattributeĕĕĕĕŮőæüžăĹăéćĬăđ' ŮçŽĐăđ'ĐçŘĔĕĀžĕ;ŚçŽĐæŮüăĀŽ  
ăjăăăŘăžĕăřĔăôČăĀŸæĹĹăyĀäyłpropertyĕĀŇæŮăĕIJĀæŤžăŘŸăŎşæĬĕçŽĐăžççăĀăĀĆ  
ăŽăäyžĕĕĕŮăattributeçŽĐăžççăĀĕĹŸæŸřăĬăĤăĀăŎşæăüăĀĆ

PropertiesĕĹŸæŸřăyĀçġ■ăôŽăžĹăĹĹăĀĔĕőăçőŮattributeçŽĐæŮzæşŤăĀĆ  
ĕĹŽçġ■çşzăđŇçŽĐattributeşăžüăy■äijŽĕĕŇăôđéŽĔçŽĐă■ŸăĈüijŇĕĀŇæŸřăIJĬéIJĀĕĕĀçŽĐæŮüăĀŽĕőăçőŮ

```
import math
class Circle:
    def __init__(self, radius):
        self.radius = radius

    @property
    def area(self):
        return math.pi * self.radius ** 2
```



```

@property
def diameter(self):
    return self.radius * 2

@property
def perimeter(self):
    return 2 * math.pi * self.radius

```

The `Circle` class has two properties, `diameter` and `perimeter`, which are calculated based on the `radius` attribute. The `diameter` property is simply twice the radius, and the `perimeter` property is calculated using the formula  $2 \times \pi \times \text{radius}$ .

```

>>> c = Circle(4.0)
>>> c.radius
4.0
>>> c.area # Notice lack of ()
50.26548245743669
>>> c.perimeter # Notice lack of ()
25.132741228718345
>>>

```

The `Person` class has two properties, `first_name` and `last_name`, which are used to store the first and last names of a person. The `first_name` property is a simple attribute, and the `last_name` property is a property that is calculated based on the `first_name` attribute.

```

>>> p = Person('Guido')
>>> p.get_first_name()
'Guido'
>>> p.set_first_name('Larry')
>>>

```

The `Person` class has two properties, `first_name` and `last_name`, which are used to store the first and last names of a person. The `first_name` property is a simple attribute, and the `last_name` property is a property that is calculated based on the `first_name` attribute.

```

class Person:
    def __init__(self, first_name, last_name):
        self.first_name = first_name
        self.last_name = last_name

    @property
    def first_name(self):
        return self._first_name

    @first_name.setter
    def first_name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._first_name = value

```



```
# Repeated property code, but for a different name (bad!)
@property
def last_name(self):
    return self._last_name

@last_name.setter
def last_name(self, value):
    if not isinstance(value, str):
        raise TypeError('Expected a string')
    self._last_name = value
```

éĜ■āđ■āzčăĀăijŽārijeĠt'èĠĈèĈĤăĀĀæŸŞăĠžēŢŽăŠŇăyŚéŽŇċŽĎċÍŇăžŔăĀĈăē;æŭĹæĀŕæŸŕijŇéĀ  
 āŖŕăžēāŖĈèĀĈ8.9ăŠŇ9.21ārŔèĹĈċŽĎĀĒăőzāĀĈ

## 10.7 8.7 ěŤĈŤĹĹŹşzæŰzæşŢ

éŰőéĈŸ

ăĵăæĈşăĪĴă■Ŗċşzäy■ěŤĈŤĹĹŹşzċŽĎæşŖăyĴăŭşċzŔèċŇèċĒċŽŰċŽĎæŰzæşŢăĀĈ

èġĉăĒşşæŰzæăĴĹ

äyžăžĒěŖĈċŤĹĹŹşz(èŭĒĈşz)ċŽĎăyĀăyĴăŰzæşŢŕijŇăŖŕăžēă;ĤċŤĪ super()  
 āĠĵæŢŕijŇăŕŤăċĪijŽ

```
class A:
    def spam(self):
        print('A.spam')

class B(A):
    def spam(self):
        print('B.spam')
        super().spam()  # Call parent spam()
```

super() \_\_init\_\_()  
 æŰzæşŢăy■ċăőăĤĹĹŹşzèċŇæ■ċăőċŽĎăĴĴăġŇăŇŰăžĒŕijŽ

```
class A:
    def __init__(self):
        self.x = 0

class B(A):
    def __init__(self):
        super().__init__()
        self.y = 1
```

super() ċŽĎăŔēād'ŰăyĀăyĴăyŷèġĀċŤĴæşŢăĠžċŖăĪĴèċĒċŽŰPythonċĴ'žăőĴæŰzæşŢċŽĎăžċăĀăy

```

class Proxy:
    def __init__(self, obj):
        self._obj = obj

    # Delegate attribute lookup to internal obj
    def __getattr__(self, name):
        return getattr(self._obj, name)

    # Delegate attribute assignment
    def __setattr__(self, name, value):
        if name.startswith('_'):
            super().__setattr__(name, value) # Call original __
↪setattr__
        else:
            setattr(self._obj, name, value)

```

aIJlāyŁÉÍcāzččāAāy■ījŇ\_\_setattr\_\_() çŽDāōđçŎřāŇĚāŔnāyĀāyłāŔ■ā■ŮæčĀæšēāĀĆ  
 æĖĆædIJæšŔāyłāsdæĀğāŔ■āzēāyŇāĹŠçžŁ( )āijĀād't'īijŇārsēĀŽēŁĠ super()  
 èŕČçŦlāŎšāğŇçŽD \_\_setattr\_\_() īijŇ āŔēāĹŽçŽDēŕlārsāğŦæt'ĭçžŽāĒĚēČĭçŽDāzčçŔĒāŕzēšā  
 self.\_obj āŎžād'ĎçŔĒāĀĆ èŁŽçIJŇāyŁāŎžæIJL'çČzæĎŔæĀīījŇāŽāyžāŕšçŏŮæšæIJL'æŸĭāijŔçŽDæ  
 super() āz■çĐūāŔfāzæIJL'æŦĹçŽDāūēāIJāĀĆ

## èõlèõž

āōđéŽĚāyŁīijŇād'ğāōūāŕzāžŎāIJĪPythonāy■āēČā;Ŧæ■čçāŏā;ŁçŦĪ super()  
 āĠ;æŦŕæŽŏēA■çšēāzŇçŦŽāŕŠāĀĆ ä;āæIJL'æŮūāĀŽāijŽçIJŇāĹŕāČŔāyŇēĭcèŁŽæāūçŽŦ'æŎēŕČçŦĭçŁūçšçç

```

class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        Base.__init__(self)
        print('A.__init__')

```

āŕ;çŏāŕzāžŎād'ğēČĭāĹĒāzččāAēĀŇēĭĀēŁZāžĹāAŽæšāžĀāžĹēŮŏécŸīijŇā;ĒæŸŕāIJlæŽŦ'ād'■æĪČçŽĹ  
 æŕŦāēČīijŇēĀĆèŽŠāēČāyŇçŽDæČĚāĒīijŽ

```

class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        Base.__init__(self)
        print('A.__init__')

class B(Base):

```

```

def __init__(self):
    Base.__init__(self)
    print('B.__init__')

class C(A,B):
    def __init__(self):
        A.__init__(self)
        B.__init__(self)
        print('C.__init__')

```

æĒĈæđĬJăjæĒĤRëaÑeĤŽæōġăzĉĉăAârŝăijŽăRŚĉŎř Base.\_\_init\_\_()  
 ěĈnërĈĉŤĭăyď' æñqĭjÑæĈăyÑæL' Āĉď' žĭjŽ

```

>>> c = C()
Base.__init__
A.__init__
Base.__init__
B.__init__
C.__init__
>>>

```

årĤrëĈjăyď' æñqërĈĉŤĭ Base.\_\_init\_\_() æŝăăzĂăžĹăĬRăď' ĎřĭjÑăjEæĬJL' æŮŭăĂŽăĤ' äy■æŸřăĂĆ  
 årĤëăŸĂæŮžěĭĭjÑăĀĜëōĵăjăăĬJăzĉĉăAăy■æ■cæĹRăjĤĉŤĭ super()  
 ĭjÑĉzŚæđĬJăřŝăĵĹăŎÑĉĵĹŌăžEřĭjŽ

```

class Base:
    def __init__(self):
        print('Base.__init__')

class A(Base):
    def __init__(self):
        super().__init__()
        print('A.__init__')

class B(Base):
    def __init__(self):
        super().__init__()
        print('B.__init__')

class C(A,B):
    def __init__(self):
        super().__init__() # Only one call to super() here
        print('C.__init__')

```

ĕĤRëaÑeĤŽăyĹæŮřĉL'ĹæĬJăŋăRŎřĭjÑăjăăĭjŽăRŚĉŎřæřRăyĤ \_\_init\_\_()  
 æŮžæŝŤăRĭăĭjŽěĈnërĈĉŤĭăyĂæñqăžEřĭjŽ

```

>>> c = C()
Base.__init__
B.__init__

```

```
A.__init__
C.__init__
>>>
```

äyžāẸāijĐäyĖāōČčŽĐāŎšçŘĚijNæĹŚāžñéIJĀēēAēĹśçČzæŮűéŮťèğćéĠăyŊPythonæŸřăĈă;Ţăōđ  
årzāžŎă;ăăōŽāzĹčŽĐæfRăyĀăyĹçşziiŊPythonăijŽēōaçōŮăĠžăyĀăyĹæĹĀērŞçŽĐæŰzæşŢēğćæđŘēāžāžŔ(Ĺ  
ēfŽăyĹMROăĹŮēăĹăřsæŸřăyĀăyĹçōĀăŢçŽĐæĹĀæIJĹăşžçşçŽĐçžĤæĀğēāžāžŘēăĹăĈă;ŊăēĆiijŽ

```
>>> C.__mro__
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>,
<class '__main__.Base'>, <class 'object'>)
>>>
```

äyžāẸāōđçŎřçžğæĹĤiijŊPythonăijŽăIJĹMROăĹŮēăĹăyĹăžŎăűēăĹăřăŖşăijĀăğŊæşēæĹăşžçşziiŊŊçŽť  
ēĀŊēfŽăyĹMROăĹŮēăĹçŽĐæđĐēĀăæŸřēĀžēfĠăyĀăyĹC3çžĤæĀğăŊŰçōŮæşŢæĹēăōđçŎřçŽĐăĈ  
æĹŚāžñăy■ăŎzæűçĹűēfŽăyĹçōŮæşŢçŽĐæŢřă■ēăŎšçŘĚijNăōĈăōđēŽĖăyĹăřsæŸřăĹĹăžűæĹĀæIJĹçĹŮç

- āŖçşziiŊŽăĹĹăžŎçĹŮçşžèćnăĈĀæşē
- āđŽăyĹçĹŮçşziiŊŽăăzæ■ăōĈăžñăĹIJăĹŮēăĹăy■çŽĐēāžāžŘēćnăĈĀæşē
- āēĈæđIJăřzăyŊăyĀăyĹçşză■ŸăIJăyđ'ăyĹăĹĹæşŢçŽĐēĀĹæŊĹ'ijŊēĀĹæŊĹ'ĉñăyĀăyĹçĹŮçşž

ēĀĀăōđēřťiijŊă;ăæĹĀēēAçşēēAşçŽĐăřsæŸřMROăĹŮēăĹăy■çŽĐçşžēāžāžŘăijŽēōĹă;ăăōŽăzĹçŽĐăž  
ă;şă;ăă;ĤçŢĹsuper()ăĠ;æŢřăŮűiijŊPythonăijŽăIJĹMROăĹŮēăĹăyĹçžğçz■æŘIJĉťăăyŊăyĀăyĹçşzăĀ  
ăŖĹēēAăřRăyĹēĠăăōŽăzĹçŽĐæŰzæşŢçžşăyĀă;ĤçŢĹsuper()ăžűăŖĹēŖĈçŢĹăōĈăyĀăñăijŊéĈăžĹæŎğăĹŮăĤAæIJăçžĹăijŽēA■ăŎēăōŊăŢťăyĹM-  
ROăĹŮēăĹiijŊăřRăyĹæŰzæşŢăžşăŖĹăijŽēćnēřĈçŢĹăyĀăñăăĈ  
ēfŽăžşæŸřăyžăžĀăžĹăIJĹĉñăžŊăyĹă;Ŋă■Ŗăy■ă;ăăy■ăijŽēřĈçŢĹăyđ'ăñăBase.  
\_\_init\_\_()çŽĐăŎşăŽăăĈ

super()æIJĹăyĹăžđ'ăžžăŖĈæĈĹçŽĐăIJăřŰzæŸřăōĈăžűăy■ăyĀăōŽăŎzæşēæĹăşŖăyĹçşzăIJĹMRO  
ă;ăçŢŽēĠşăŖăřăēăIJăyĀăyĹæşăæIJĹçŽťăŎēçĹŮçşçžŽĐçşžăy■ă;ĤçŢĹăōĈăĈă;ŊăēĆiijŊēĈēŽŚăēĈăyŊē

```
class A:
    def spam(self):
        print('A.spam')
        super().spam()
```

ăēĈæđIJă;ăērŢçĹĀçŽťæŎēă;ĤçŢĹēfŽăyĹçşzăřsăijŽăĠžēŢŽiijŽ

```
>>> a = A()
>>> a.spam()
A.spam
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 4, in spam
AttributeError: 'super' object has no attribute 'spam'
>>>
```

ă;ĖæŸřiijŊăēĈæđIJă;ăă;ĤçŢĹăđ'ŽçžğæĹĤçŽĐēřĹçIJŊçIJŊăijŽăŖŚçŢşăžĀăžĹiijŽ

```
>>> class B:
...     def spam(self):
...         print('B.spam')
...
>>> class C(A, B):
...     pass
...
>>> c = C()
>>> c.spam()
A.spam
B.spam
>>>
```

ä;ääRräzëçIJNälRäIJlçszAäy■ä;çTl  
 åóðéZËäyLërÇçTlçZDæYrëu§çszAærnæUääË§çszçZDçszBäy■çZD spam() æÚzæsTäÄÇ  
 èfZäyIçTlçszCçZDMROälUèaIärsäRräzëäöNäÈlègçéGLæyÈæëZäzEijZ

```
>>> C.__mro__
(<class '__main__.C'>, <class '__main__.A'>, <class '__main__.B'>,
<class 'object'>)
>>>
```

åIJlääZäZLæuüäËçszçZDæUüäÄZëfZæuüä;çTl  
 æYrä;LæZóéA■çZDäÄÇäRräzëäRÇèÄÇ8.13åŠN8.18ärRèLCäÄÇ

çDüèÄNrijNçTsäzÖ super() äRrèÇ;äijZërÇçTlây■æYrä;äæÇsèçAçZDæÚzæsTijNä;ääzTërëéAç;läy  
 éçÚäÈLrijNçqäöäfläIJlçzgaeL'æ;§çszäy■æL'ÄæIJL'çZyâRñâR■â■UçZDæÚzæsTæNëæIJL'âRfäEijäöçZDâR  
 èfZæuüäRräzëçqäöäfl super() èrÇçTlâyÄäyIèlçZt'æÖèçLüçszæÚzæsTæUüäy■äijZäGžéTZäÄÇ  
 äÈüæñärijNæIJÄäçqäöäflæIJÄéaüäsÇçZDçszæRRä;ZäzEèfZäyIæÚzæsTçZDåóçÖrijNèfZæuüçZDèrläIJl

åIJlPythonçd'äNžäy■ärzäzÖ super() çZDä;ççTlæIJLæUüäÄZäijZäijTæIäyÄäzZäZL'èöäÄÇ  
 är;çqäæÇæ■d'rijNäçÇædIJäyÄäL GéažälL'çZDèrlrijNä;ääzTërëäIJlä;äæIJÄæÚräzççäAäy■ä;ççTlääÇäÄÇ  
 Raymond Hettingeräyžæ■d'äEžZäžEäyÄçrGéIdäyÿäë;çZDæÚGçnä äÄIJPythonâÄŽs super()  
 Considered Super!âÄI rijN éÄZëfGäð'gèGRçZDä;Nä■RäRŠæLSäznègçéGLäžEäyžäzÄäZL  
 super() æYräðAäë;çZDäÄÇ

## 10.8 8.8 ä■Rçszäy■æL'fäsTproperty

éUóécY

åIJlâ■Rçszäy■rijNä;äæÇsèçAæL'fäsTäóZäZL'åIJlçLüçszäy■çZDpropertyçZDäLšèÇ;äÄÇ

ègçäEşæÚzæaL

èÄÇèZŠäçCäyNçZDäzççäArijNäöÇCäóZäZL'äžEäyÄäyIpropertyijZ

```
class Person:
    def __init__(self, name):
```

```

        self.name = name

    # Getter function
    @property
    def name(self):
        return self._name

    # Setter function
    @name.setter
    def name(self, value):
        if not isinstance(value, str):
            raise TypeError('Expected a string')
        self._name = value

    # Deleter function
    @name.deleter
    def name(self):
        raise AttributeError("Can't delete attribute")

```

äÿÑéÍæŸřäÿÄäÿłçď'žă;ŇčšziiĴŇăőČčžğæŁĤèĜłPersonâžúæŁł'âšŤăžĚ name  
 âšđæĂğçŽĐăŁšèČ;iiĴŽ

```

class SubPerson(Person):
    @property
    def name(self):
        print('Getting name')
        return super().name

    @name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)

    @name.deleter
    def name(self):
        print('Deleting name')
        super(SubPerson, SubPerson).name.__delete__(self)

```

æŎëäÿŇæİëă;ĤçŤİëĤŽäÿłæŮřčšziiĴŽ

```

>>> s = SubPerson('Guido')
Setting name to Guido
>>> s.name
Getting name
'Guido'
>>> s.name = 'Larry'
Setting name to Larry
>>> s.name = 42
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example.py", line 16, in name

```

```
raise TypeError('Expected a string')
TypeError: Expected a string
>>>
```

æĈædIJä;äazĖäzĖäRlæČşæL'f'ásTpropertyçŽDæşŘäyÄäylæŮzæşTijNéCčázLāRřäzěäČRäyNéIcéfZæ

```
class SubPerson(Person):
    @Person.name.getter
    def name(self):
        print('Getting name')
        return super().name
```

æLŮèĀĖijNä;ääRlæČşæfōæTzsetteræŮzæşTijNārsèfZázLāEZijŽ

```
class SubPerson(Person):
    @Person.name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)
```

## èóIèőž

āIJlāRčşzäy■æL'f'ásTäyÄäylpropertyāRrèČ;äijŽäijTètūā;Lād'Žäy■æYşārşègŁçŽDēŮóécYrijN  
āZäyžäyÄäylpropertyāĖūāóðæYř getterāĀAsetter āšN  
deleter æŮzæşTçŽDēZEāRLiijNèĀNäy■æYřā■TäylæŮzæşTaĀC  
āZäæ■d'rijNā;šä;æL'f'ásTäyÄäylpropertyçŽDæŮūāĀŽiijNā;æéIJĀèeAāĖŁçāóāóŽä;āæYřāRčèeAéG■æŮřāó

āIJlčnnäyÄäylā;Nā■Räy■iijNæL'ĀæIJL'çŽDpropertyæŮzæşTéČ;ècnéG■æŮřāóŽázL'āĀC  
āIJlærRäyÄäylæŮzæşTäy■iijNā;ŁçTlāžE super() æIèèrČçTlçLúçşzçŽDāóðçŌřāĀC  
āIJl setter āG;æTřäy■ā;ŁçTl super(SubPerson, SubPerson).  
name.\_\_set\_\_(self, value) çŽDèr■āRèæYřæşqæIJL'éTŽçŽDāĀC  
äyžāžEāgTæL'YçžŽázNāL'■āóŽázL'çŽDsetteræŮzæşTijNéIJĀèeAārEæŌgāLūæIČäijæĀšçžŽázNāL'■āóŽáz  
\_\_set\_\_() æŮzæşTaĀC äy■èfGrijNèŌūāRŮèfZäylæŮzæşTçŽDāTřäyĀéĀTā;DæYřā;ŁçTlçşzāRŸéGRèĀ  
èfZázşæYřäyžāžĀāžLæLšāžnèeAā;ŁçTl super(SubPerson, SubPerson)  
çŽDāŌşāZāāĀC

æĈædIJä;ääRlæČşéG■āóŽázL'āĖūäy■äyÄäylæŮzæşTijNéCčāRlā;ŁçTl @property  
æIJnèžnæYřäy■ād'şçŽDāĀCærTāeČiijNäyNéIćçŽDžčçāAārşæŮāæşTāuèä;IJijŽ

```
class SubPerson(Person):
    @property # Doesn't work
    def name(self):
        print('Getting name')
        return super().name
```

æĈædIJä;æèrTçIĀèfŘèaNäijŽāRŚçŌrsetterāG;æTřæT'äylæŮLād'sāžEijijŽ

```
>>> s = SubPerson('Guido')
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in <module>
File "example.py", line 5, in __init__
    self.name = name
AttributeError: can't set attribute
>>>
```

äjäãžŤerëãĈRázNãL■èrt'èfĜçŽĐéCĉæũäăŃóæŤžázĉăAïijŽ

```
class SubPerson(Person):
    @Person.name.getter
    def name(self):
        print('Getting name')
        return super().name
```

èĚŽázĹãĚŽãŔŌïijŊpropertyázNãL■ũšçžŔãŏŽázĹ'èĚĜçŽĐæŮzæşŤäijŽèćnăđ'■ăĹŭèĚĜæĬëijŊèĂŊget

```
>>> s = SubPerson('Guido')
>>> s.name
Getting name
'Guido'
>>> s.name = 'Larry'
>>> s.name
Getting name
'Larry'
>>> s.name = 42
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example.py", line 16, in name
    raise TypeError('Expected a string')
TypeError: Expected a string
>>>
```

ăIJĭèĚŽäyĭçĹ'žăĹŋçŽĐèğĉăĖşæŮzæăĹăy■ïijŊæĹŤăznæşăăĹđæşŤă;ĤçŤĭæŽŤ'ăĹăéĂŽçŤĭçŽĐæŮzăijŔăĈ  
Person çşzăŔ■ăĂĈ æĈăđIJă;ăăy■çşĉéAşăĹŕăžŤæŸŕăşĭăyĭăşşçşzăŏŽázĹ'ăžĖpropertyïijŊ  
éĈĉă;ăăŔĭèĈ;éĂŽèĚĜéĜ■æŮŕăŏŽázĹ'æĹ'ĂæIJĹpropertyăžŭă;ĤçŤĭ super()  
æĬëăŕĖæŎğăĹŭăĬăijăéĂşçžŽăĹ'■éĬçŽĐăŏđĈŎŕăĂĈ

ăĂijçŽĐæşĭăĎŔçŽĐæŸŕăyĹéĬăëijŤçđ'žçŽĐçŋăyĂçğ■æĹĂæIJŕèĚŸăŔŕázèèćŋçŤĭăĬëăĹĹ'ăşŤăyĂăyĭă

```
# A descriptor
class String:
    def __init__(self, name):
        self.name = name

    def __get__(self, instance, cls):
        if instance is None:
            return self
        return instance.__dict__[self.name]

    def __set__(self, instance, value):
        if not isinstance(value, str):
```



```

        raise TypeError('Expected a string')
    instance.__dict__[self.name] = value

# A class with a descriptor
class Person:
    name = String('name')

    def __init__(self, name):
        self.name = name

# Extending a descriptor with a property
class SubPerson(Person):
    @property
    def name(self):
        print('Getting name')
        return super().name

    @name.setter
    def name(self, value):
        print('Setting name to', value)
        super(SubPerson, SubPerson).name.__set__(self, value)

    @name.deleter
    def name(self):
        print('Deleting name')
        super(SubPerson, SubPerson).name.__delete__(self)

```

æIJĀāRŌāĀijçŽDæşlæĐRçŽDæŸriijNèrZāLrèŁŻéGŃæŮūriijNā;āāzTèrēāijŽāRŚçŌrā■RçśZāŃŮ  
 setter āŠŃ deleter æŮzæşŤāĒūāōdæŸrā;ŁçōĀā■ŤçŽDāĀĆ  
 èŁŻéGŃæijŤçd'žçŽDèğcāEşæŮzæāLāRŃæāūéĀĆçŤriijNā;EæŸrāIJĪ PythonçŽDissueéāŤéĬ  
 æŁčāSŁçŽDāyĀāyĭbugriijNæLŮēōyāijŽā;Łā;ŮārEæĭççŽDPythonçL'LæIJñāy■āGžçŌrāyĀāyĭæŽt'āŁăçōĀæt'

## 10.9 8.9 āLŽāzzæŮrçŽDçşzæLŮāōdā;ŃāsdæĀğ

ēŮōécŸ

ä;äæČşāLŽāzzāyĀāyĭæŮrçŽDæŃæIJL'āyĀāzŽéĭād'ŮāŁşèČ;çŽDāōdā;ŃāsdæĀğçşzādŃriijNærŤæČç

èğcāEşæŮzæāL

æçCædIJā;äæČşāLŽāzzāyĀāyĭāĒĭæŮrçŽDāōdā;ŃāsdæĀğriijNārřāzēēĀŽēŁGāyĀāyĭæRŘèřrāŽĭçşçŽD.

```

# Descriptor attribute for an integer type-checked attribute
class Integer:
    def __init__(self, name):
        self.name = name

```

```

def __get__(self, instance, cls):
    if instance is None:
        return self
    else:
        return instance.__dict__[self.name]

def __set__(self, instance, value):
    if not isinstance(value, int):
        raise TypeError('Expected an int')
    instance.__dict__[self.name] = value

def __delete__(self, instance):
    del instance.__dict__[self.name]

```

äyÄäylæRRèfräZlârsæYräyÄäylæoðçÖräzEäyL'äylæäyæfÇçZDäsdæÄgèøféUöæS■ä;IJ(get,  
 set, delete)çZDçszijN äLEäLnäyZ \_\_get\_\_() äÄA\_\_set\_\_() äŠN  
 æfZäyL'äylçL'zæøLçZDæÜzæSṽTäÄC  
 èfZäzZæÜzæSṽTæÖèäRÜäyÄäylæoðä;Nä;IJäyžè;ŠäEërijNäzNäRÖçZyāzTçZDæS■ä;IJäoðä;NäzTāsCçZDä■  
 äyžäzEä;fçTlāyÄäylæRRèfräZlīijNéIJÄärEèfZäylæRRèfräZlçZDäoðä;Nä;IJäyžçszāsdæÄgæT;äLräyÄ

```

class Point:
    x = Integer('x')
    y = Integer('y')

    def __init__(self, x, y):
        self.x = x
        self.y = y

```

ä;Šä;æèfZæuüäAZäRÖrijNæL'ÄæIJL'ärzæRRèfräZlâsdæÄg(æfTæCæLÜy)çZDèøféUöäijZècñ  
 \_\_get\_\_() äÄA\_\_set\_\_() äŠN \_\_delete\_\_() æÜzæSṽTæ■TèÖuäLrāÄCä;NäeCijZ

```

>>> p = Point(2, 3)
>>> p.x # Calls Point.x.__get__(p, Point)
2
>>> p.y = 5 # Calls Point.y.__set__(p, 5)
>>> p.x = 2.3 # Calls Point.x.__set__(p, 2.3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "descrip.py", line 12, in __set__
    raise TypeError('Expected an int')
TypeError: Expected an int
>>>

```

ä;IJäyžè;ŠäEërijNæRRèfräZlçZDæfRäyÄäylæÜzæSṽTäijZæÖèäRÜäyÄäylæS■ä;IJäoðä;NäÄC  
 äyžäzEäoðçÖrèrûæSÇæS■ä;IJijNäijZçZyāzTçZDæS■ä;IJäoðä;NäzTāsCçZDä■ÜäEÿ(\_\_dict\_\_äsdæÄg)äÄC  
 æRRèfräZlçZD self.name äsdæÄgä■YäClāzEäIJäoðä;Nä■ÜäEÿäy■ècñäoðéZÈä;fçTlāLrçZDkeyäÄC

## ěóľěőž

æŘŘěřřăŹĺăŔřăőđċŎřăđ' ġéĈĺăĹEPythonċşzċĹ'žăĂġăŷ■ŽĎăžŤăşĈé■ŤăşŤiijŇ  
ăŇĚăŇň @classmethod āĀĀ@staticmethod āĀĀ@property iijŇċŤŽěĠşăŸř  
\_\_slots\_\_ ċĹ'žăĂġăĂĈ

éĂŽěĤĠăőŽăžĹ'ăŷĂăŷĹăŔŘěřřăŹĺiijŇă;ăăŔřăžěăĬĴăžŤăşĈă■ŤěŎăăăăĤĈċŽĎăőđă;Ňăş■ă;ĬĴ(get,  
set, delete)iijŇăžăŷăŤăŔřăőŇăĤĹěĠăőŽăžĹ'ăőĈăžŇċŽĎăăŇăŷăăĈ  
ěĤŽăŸřăŷĂăŷĹăiijăđ' ġċŽĎăăăăĤiijŇăĬĴ'ăžĤăőĈă;ăăŔřăžěăăđċŎřăĴăđ' ŽénŸċžġăĹşěĈ;iijŇăžăŷăŤăőĈă  
æŘŘěřřăŹĺċŽĎăŷĂăŷĹăřŤěĴĈăŽřăĈŤĈċŽĎăĬĴăŰžăŸřăăĈăŔĹěĈ;ăĬĴċşzċžġăĹŇěċŇăăőŽăžĹ'ŷiijŇăĂŇăŷă

```
# Does NOT work
class Point:
    def __init__(self, x, y):
        self.x = Integer('x') # No! Must be a class variable
        self.y = Integer('y')
        self.x = x
        self.y = y
```

ăŔŇăŰiijŇ\_\_get\_\_() æŰžăşŤăăđċŎřěŷăăĤăřŤċĬŇăŷĹăŎžěăĀăđ'■ăĤăăĴăŰăđ' ŽiijŽ

```
# Descriptor attribute for an integer type-checked attribute
class Integer:

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return instance.__dict__[self.name]
```

\_\_get\_\_() ċĬŇăŷĹăŎžăĬĴċĈăđ'■ăĤĈċŽĎăŎşăžăă;ŤċžŤăžăőăăđă;ŇăŔŸěĠŔăŤŇċşăŔŸěĠŔċŽĹ  
ăġĈăđĬĴăŷĂăŷĹăŔŘěřřăŹĺěċŇă;ŤăĂŽăŷĂăŷĹċşăŔŸěĠŔăĤěăĤăŰiijŇăĈĈăžĹ instance  
ăŔĈăŤřěċŇěăĴċ;ăăĹŔ None āĈĈ ĤċŽċ■ăĈĤăĤăŷŇiijŇăăĠăĠăĂŽăşŤăřşăŸřċăă■ŤċŽĎĤăŽđĤăžă

```
>>> p = Point(2,3)
>>> p.x # Calls Point.x.__get__(p, Point)
2
>>> Point.x # Calls Point.x.__get__(None, Point)
<__main__.Integer object at 0x100671890>
>>>
```

æŘŘěřřăŹĺěĂŽăŷăŸăŸřăŸřěĈăžăžă;ĤċŤĴăĹřěĈĤěăŔăŹĺăĹŰăĤĈċşzċŽĎăđ' ġăđŇăăăĤăđăŷă■ċŽĎăŷĂăŷĹċžĎă  
ăŷăăŷăă;Ňă■ŔiijŇăŷŇěĤăŸřăĂăžăŽăŽŤ ĤénŸċžġċŽĎăşăžăăŎæŘŘěřřăŹĺċŽĎăžċċăĂiijŇăžăŷăŰĹ'ăŔĹăĹŔăŷăĂ

```
# Descriptor for a type-checked attribute
class Typed:
    def __init__(self, name, expected_type):
        self.name = name
        self.expected_type = expected_type
    def __get__(self, instance, cls):
```

```

    if instance is None:
        return self
    else:
        return instance.__dict__[self.name]

def __set__(self, instance, value):
    if not isinstance(value, self.expected_type):
        raise TypeError('Expected ' + str(self.expected_type))
    instance.__dict__[self.name] = value
def __delete__(self, instance):
    del instance.__dict__[self.name]

# Class decorator that applies it to selected attributes
def typeassert(**kwargs):
    def decorate(cls):
        for name, expected_type in kwargs.items():
            # Attach a Typed descriptor to the class
            setattr(cls, name, Typed(name, expected_type))
        return cls
    return decorate

# Example use
@typeassert(name=str, shares=int, price=float)
class Stock:
    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price

```

æIJĀāRŌēēAæŃGāGzçŽDäyĀçĆzæYřiiĴNāēĆæđIJäĵääRĭæYřæČşçōĀā■TçŽDēĠāōŽāzL'æşŘäyĭçşzçŽēfŽçġ■æČĚāEĭäyNāĭ;ĤçTĭ8.6ārRèLCāzNçz■çŽDpropertyæLĀæIJřaiĵZæZt'āLāāōzæYŞāĀĆāĭŞçĭNāzŘäy■æIJL'āĭLād'ŽéĠ■ād'■āzčçāAçŽDæŪūāĀZæRRèřrāZĭāřsāĭLæIJL'çTĭāžE(ærTāēCāĭāæČsāIJĭāĭāzčçāAçŽDāĭLād'ŽāIJræŪzāĭ;ĤçTĭæRRèřrāZĭæRRāĭZçŽDāLşèČĭæLŪēĀĔārEāōCāĭI

## 10.10 8.10 äĭĤçTĭāzūēēŒşēōaçōŪāśdæĀğ

éŪōécŸ

äĭāæČşārEäyĀäyĭāRĭērzaśdæĀğāōŽāzL'æLŘäyĀäyĭpropertyiiĴNāzūāyTāRĭāIJĭēōēŪōçŽDæŪūāĀZæL'äĭEæYřäyĀæŪēècnèōēēŪōāRŌiiĴNāĭ;āāyNæIJZçzŞæđIJāĀĭjècnçĭjŞā■YēĭuæĭēiiĴNäy■çTĭærRæñæČĭāŌzèōāĭ

èğçāEşşæŪzæaĭL

āōŽāzL'äyĀäyĭāzūēēŒşāśdæĀğçŽDäyĀçġ■énYæTĭLæŪzæşTæYřēĀŽēfGāĭ;ĤçTĭäyĀäyĭæRRèřrāZĭçşziiĴN

```

class lazyproperty:
    def __init__(self, func):

```

```

        self.func = func

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            value = self.func(instance)
            setattr(instance, self.func.__name__, value)
            return value

```

äjäéIJÄèçAäČŘäyŇéÍcéŁŻæăũăIJläyĂäyŁçśzäy■ä;ŁçŤlăóČüjŽ

```

import math

class Circle:
    def __init__(self, radius):
        self.radius = radius

    @lazyproperty
    def area(self):
        print('Computing area')
        return math.pi * self.radius ** 2

    @lazyproperty
    def perimeter(self):
        print('Computing perimeter')
        return 2 * math.pi * self.radius

```

äyŇéÍcăIJläyĂäyŁăzd'ăžŠčŎřăćČăy■ăijŤçd'žăóČçŽĎă;ŁçŤlŭjŽ

```

>>> c = Circle(4.0)
>>> c.radius
4.0
>>> c.area
Computing area
50.26548245743669
>>> c.area
50.26548245743669
>>> c.perimeter
Computing perimeter
25.132741228718345
>>> c.perimeter
25.132741228718345
>>>

```

ăžŤçzEèğĆărşă;ăăijŽăŔŚçŎřăűŁăAř      Computing area      ăŠŇ      Computing  
perimeter äžĚăzĚăĜçŎřăyĂăňăăĂĆ

## èõléõž

åŁŁåđŽæŮŮåĀŽriiŃæđĐéĀäyĀäylāzŮēſšèõąõŮåśđæĀğçŽĎäyžèçĄçŽõçŽĎæŸřāyžāžĒæŘŘā■ĠæĀ  
äŁŃāçĈriiŃäĵāāŘřāžèçĀŁāĒēõąõŮēſŽāžŽāśđæĀğāĀijriiŃēŽđ' éİđāĵçIJšçŽĎéIJĀèçĀāõĈāžñāĀĈ  
èſŽéĠŃāijŤçđ' žçŽĎæŮžæāŁĀřśæŸřçŤĪæİēāõđçŎřēſŽæāũçŽĎæŤĪæđIJçŽĎriiŃ  
āŘĪāy■ēſĠāõĈæŸřéĀŽēſĠāžèçİđāyŷénŸæŤĪçŽĎæŮžāijŘāĵçŤĪæŘŘēſřāŽĪçŽĎäyĀäylçşĵāçŽçĪ'žæĀğæİē

æ■čāçĈāIJĪāĒŮāzŮārŘēĪĈ(āçĈ8.9ārŘēĪĈ)æĻ'ĀèõşçŽĎéĈçæāũriiŃāĵŞāyĀäylæŘŘēſřāŽĪçñæŤĪāĒēäy  
æřŘæñāèõſēŮõåśđæĀğæŮŮāõĈçŽĎ \_\_get\_\_() āĀĀ\_\_set\_\_() āŠŃ \_\_delete\_\_()  
æŮžæşŤāřśāijŽèçñèğçāŘŚāĀĈ äy■ēſĠriiŃāçĈæđIJäyĀäylæŘŘēſřāŽĪāzĒāzĒĒāŘĪāõŽāzĪ'āžĒäyĀäyl  
\_\_get\_\_() æŮžæşŤçŽĎēřĪriiŃāõĈæřŤéĀŽāyŷçŽĎāĒŮæIJĪæŽř'āijşçŽĎçzŚāõŽāĀĈ  
çĪ'žāĪŃāIJriiŃāŘĪæIJĪ'āĵşèçñèõſēŮõåśđæĀğäy■āIJĪāõđäĪŃāžŤāśĈçŽĎā■ŮāĒyāy■æŮŮ  
\_\_get\_\_() æŮžæşŤæĻ■āijŽèçñèğçāŘŚāĀĈ

lazyproperty çşzāĪ'çŤĪēſŽāyĀçĈriiŃāĵçŤĪ \_\_get\_\_()  
æŮžæşŤāIJĪāõđäĪŃäy■āŸāĈİēõąõŮŮāĠžæİēçŽĎāĀijriiŃ èſŽāyĪāõđäĪŃāĵçŤĪçŽyāŘŃçŽĎāŘ■ā■ŮāĵIJāyž  
èſŽæāũäyĀæİēriiŃçzşæđIJāĀijèçñā■ŸāĈĪāIJĪāõđäĪŃā■ŮāĒyāy■āžŮāyŤāžēāŘŎāřśāy■ēIJĀèçĀāĒ■āŎžèõąç  
āĵāāŘřāžēārĪēřŤæŽř æŮśāĒēçŽĎäĪŃā■ŘæİēèçĈārşçzşæđIJijŽ

```
>>> c = Circle(4.0)
>>> # Get instance variables
>>> vars(c)
{'radius': 4.0}

>>> # Compute area and observe variables afterward
>>> c.area
Computing area
50.26548245743669
>>> vars(c)
{'area': 50.26548245743669, 'radius': 4.0}

>>> # Notice access doesn't invoke property anymore
>>> c.area
50.26548245743669

>>> # Delete the variable and see property trigger again
>>> del c.area
>>> vars(c)
{'radius': 4.0}
>>> c.area
Computing area
50.26548245743669
>>>
```

èſŽçğ■æŮžæāŁæIJĪäyĀäylārŘçijžéŽŮārśæŸřèõąõŮŮāĠžçŽĎāĀijèçñāĪŽāžžāŘŎæŸřāŘřāžèèçñāſōæŤ

```
>>> c.area
Computing area
50.26548245743669
>>> c.area = 25
>>> c.area
```

```
25
>>>
```

æĈædIJä;äæNĖäĤĈèĤZäyĭéUőécYĭijNĖĈčázĹäRřázëä;ĤçTĹäyĂçğ■ā;ôæšæĈčázĹénYæTĹçŽĎăđç

```
def lazyproperty(func):
    name = '_lazy_' + func.__name__
    @property
    def lazy(self):
        if hasattr(self, name):
            return getattr(self, name)
        else:
            value = func(self)
            setattr(self, name, value)
            return value
    return lazy
```

æĈædIJä;ää;ĤçTĹèĤZäyĭçĹĹæIJñijNăřsäijŽăRŚçŎřçŎřăIJăĤôæṬzæ\$■ä;IJăũščzRăy■ècnăĖAçöyăžEĭij

```
>>> c = Circle(4.0)
>>> c.area
Computing area
50.26548245743669
>>> c.area
50.26548245743669
>>> c.area = 25
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

çĎûëĂñijNĖĤZçğ■æŰzæāĹæIJĹăyĂäyĭçijžçĈzăršæYřæĹĂæIJĹgetæ\$■ä;IJéĈ;ăĤĖéazècnăôŽăRŚăĹřæ  
getter āĜ;æṬrăyĹăŎzăĈĈèĤZäyĭèušăzNăĹ■çôĂă■ṬçŽĎăIJăôđă;Nă■ŰăĖyăy■æšĕæĹ;ăĀijçŽĎæŰzæā  
æĈædIJæĈşèŎăRŰæZĭ'ăđ'ŽăĖşăžŎpropertyăŠNăRřçôaçRĖăśđæĂğçŽĎăĤæAřĭijNăRřázëăRĈèĂĈ8.6ăřR

## 10.11 8.11 çôĂăŇŰæṬræ■őçzŞæđĎçŽĎăĹiăğNăŇŰ

### éUőécY

ă;ăăĖZăžĖā;Ĺăđ'ŽăzĖăžĖĤĹă;IJæṬræ■őçzŞæđĎçŽĎçşziiNăy■æĈşăĖZăđ'ăđ'ŽçĈçăžžçŽĎ  
\_\_init\_\_() āĜ;æṬr

### èğçăĖşæŰzæāĹ

ăRřázëăIJăyĂäyĭăşžçşzăy■ăĖZăyĂäyĭăĤĤĹçŽĎ \_\_init\_\_() āĜ;æṬrĭijŽ

```

import math

class Structure1:
    # Class variable that specifies expected fields
    _fields = []

    def __init__(self, *args):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self._
↪_fields)))
        # Set the arguments
        for name, value in zip(self._fields, args):
            setattr(self, name, value)

```

çĐúăŘŎä;řă;ăçŽĎčšćžğæL'fèĜlèfŽäyłåšžćś:

```

# Example class definitions
class Stock(Structure1):
    _fields = ['name', 'shares', 'price']

class Point(Structure1):
    _fields = ['x', 'y']

class Circle(Structure1):
    _fields = ['radius']

    def area(self):
        return math.pi * self.radius ** 2

```

ä;řçŦlèfŽăžŽćšćžŽĎčđ'žăĹŦiijŽ

```

>>> s = Stock('ACME', 50, 91.1)
>>> p = Point(2, 3)
>>> c = Circle(4.5)
>>> s2 = Stock('ACME', 50)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "structure.py", line 6, in __init__
    raise TypeError('Expected {} arguments'.format(len(self._
↪fields)))
TypeError: Expected 3 arguments

```

ăęĆăđĬjèfŸæČşæŦřăŇAăĖşéŦőă■ŮăŔĆæŦřiijŇăŔřăžěăřĚăĖşéŦőă■ŮăŔĆæŦřèőç;őăyžăőđăĹŇăşđăA

```

class Structure2:
    _fields = []

    def __init__(self, *args, **kwargs):
        if len(args) > len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self._
↪_fields)))

```



```

    # Set all of the positional arguments
    for name, value in zip(self._fields, args):
        setattr(self, name, value)

    # Set the remaining keyword arguments
    for name in self._fields[len(args):]:
        setattr(self, name, kwargs.pop(name))

    # Check for any remaining unknown arguments
    if kwargs:
        raise TypeError('Invalid argument(s): {}'.format(', '.
↪join(kwargs)))
# Example use
if __name__ == '__main__':
    class Stock(Structure2):
        _fields = ['name', 'shares', 'price']

    s1 = Stock('ACME', 50, 91.1)
    s2 = Stock('ACME', 50, price=91.1)
    s3 = Stock('ACME', shares=50, price=91.1)
    # s3 = Stock('ACME', shares=50, price=91.1, aa=1)

```

ä;äefYëČ;årEäy■āIJĲ \_fields äy■čŽDāŘ■çğrāŁāāĖēāĹrāsđæĀğäy■āŎziijŽ

```

class Structure3:
    # Class variable that specifies expected fields
    _fields = []

    def __init__(self, *args, **kwargs):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

    # Set the arguments
    for name, value in zip(self._fields, args):
        setattr(self, name, value)

    # Set the additional arguments (if any)
    extra_args = kwargs.keys() - self._fields
    for name in extra_args:
        setattr(self, name, kwargs.pop(name))

    if kwargs:
        raise TypeError('Duplicate values for {}'.format(', '.
↪join(kwargs)))

# Example use
if __name__ == '__main__':
    class Stock(Structure3):

```

```
_fields = ['name', 'shares', 'price']

s1 = Stock('ACME', 50, 91.1)
s2 = Stock('ACME', 50, 91.1, date='8/2/2012')
```

## èõìèõž

å;Şä;æIJÄëAä;£çTlåd'gëGRå;LärRçŽDæTřæ■óçzŞæđDçşzçŽDæUúåĂŽiijN  
çŽyæfTæLŊåüëäyÄäyläylåóŽázL' \_\_init\_\_() æŰzæşTëĀŊåüšiiijNä;£çTlë£Žçg■æŰzâijRâRřæžëad'gâd'g  
åIJläyLëÍççŽDåóđçÖřäy■æLSäznä;£çTlāžE setattr()  
åĜ;æTřçşzëöç;ïåşđæĀĝåĀiijijN ä;åâRřëĈ;äy■æĈşçTlë£Žçg■æŰzâijRiijNëĀŊæŸræĈşçŽt' æŎëæŽt' æŰřåó

```
class Structure:
    # Class variable that specifies expected fields
    _fields= []
    def __init__(self, *args):
        if len(args) != len(self._fields):
            raise TypeError('Expected {} arguments'.format(len(self.
↪_fields)))

        # Set the arguments (alternate)
        self.__dict__.update(zip(self._fields,args))
```

år;çõæ£ŽázşâRřæžæ■câyåüëä;IJiijNä;EæŸrâ;ŞåóŽázL'å■RçşzçŽDæUúåĂŽëŰóëçŸârşæIëāžEāĀĆ  
å;ŞäyÄäylå■RçşzåóŽázL'āžE \_\_slots\_\_ æLŰëĀĒéĂŽë£Ĝproperty(æLŰæRŘëřrāŽÍ)æIëåŊĒëçĒæşŘäylå  
éĆçāžLçŽt' æŎëëö£ëŰóåóđä;Ŋå■ŰåĒyârşäy■ëtuä;IJçTlāžEāĀĆæLSäznäyLëÍcā;£çTl  
setattr() äijZæŸç;å;ŰæŽt' éĂŽçTlāžŽiijNåŽäyžåóĈāžşéĀĈçTlāžŎå■RçşzæĈĒåĒtāĀĆ  
ë£Žçg■æŰzæşTăTřäyÄäy■æë;çŽDåIJræŰzârşæŸrâržæşŘāžŽIDEëĀŊëÍĀiijNåIJlæŸçd'žâyõåLl'åĜ;æT

```
>>> help(Stock)
Help on class Stock in module __main__:
class Stock(Structure)
...
| Methods inherited from Structure:
|
| __init__(self, *args, **kwargs)
|
...
>>>
```

årRřæžæâRĈëĀĈ9.16årRëLĈæIëâijžåLúåIJl \_\_init\_\_()
æŰzæşTäy■æŊĜåóŽâRĈæTřçŽDçşzådNç■åŘ■āĀĆ

## 10.12 8.12 ǎŏŽǎžŁ'æŎěǎŔcæŁŮèĀĚæŁ;èśǎǎŖžćśž

### éŮóécŸ

ǎǎæĈśǎŏŽǎžŁ'ǎŷĀǎŷłæŎěǎŔcæŁŮæŁ;èśǎćśžiiǎŇǎžŭǎŷŤéĀŽèĚĜæŁ'ġèǎŇćśžǎđŇæċĀæŖěæĬèçǎŏǎĬǎ■

### èġĉǎĚşæŮzæǎŁ

ǎǎĤćŤĬ abc æǎǎǎĬŮǎŔŕǎžěǎǎŁèǎžæĬćŽĎǎŏŽǎžŁ'æŁ;èśǎǎŖžćśžiiǎž

```
from abc import ABCMeta, abstractmethod

class IStream(metaclass=ABCMeta):
    @abstractmethod
    def read(self, maxbytes=-1):
        pass

    @abstractmethod
    def write(self, data):
        pass
```

æŁ;èśǎćśžćŽĎǎŷĀǎŷłćŁ'žćĈzæŸŕǎŏĈǎŷ■èĈćŽŤ' æŎěććŇǎŏđǎǎŇǎŇŮiiǎŇæŕŤǎçĈǎǎæĈśǎĈŔǎŷŇéĬèĚŽ

```
a = IStream() # TypeError: Can't instantiate abstract class
              # IStream with abstract methods read, write
```

æŁ;èśǎćśžćŽĎĈŽŏçŽĎǎŕśæŸŕèŏŕ'ǎĬŇćŽĎĈśžćžġæŁ'ĤǎŏĈǎžŭǎŏđçŎŕçŁ'žǎŏŽćŽĎæŁ;èśǎæŮzæşŤiiǎž

```
class SocketStream(IStream):
    def read(self, maxbytes=-1):
        pass

    def write(self, data):
        pass
```

æŁ;èśǎǎŖžćśžćŽĎǎŷĀǎŷłǎŷžèèĀçŤĬéĀŤæŸŕǎĬǎžćçǎĀǎŷ■æċĀæŖěæŖŔǎžŽćśžæŸŕǎŕèǎŷžćŁ'žǎŏŽćśžǎđ

```
def serialize(obj, stream):
    if not isinstance(stream, IStream):
        raise TypeError('Expected an IStream')
    pass
```

éŽđ'ǎžĚçžġæŁ'ĤèĚŽćġ■æŮžǎiǎŕǎđ'ŮiiǎŇèĚŸǎŕŕǎžèéĀŽèĚĜæşĬǎĚŇæŮžǎiǎŕǎĬèèŏŕ' æŖŔǎŷłćşžǎŏđçŎŕæ

```
import io

# Register the built-in I/O classes as supporting our interface
IStream.register(io.IOBase)
```

```
# Open a normal file and type check
f = open('foo.txt')
isinstance(f, IStream) # Returns True
```

@abstractmethod                      èŁYèĈ;æşİèğĉéIŻæĀAæŪzæşŦăĀAçşzæŪzæşŦăŠŅ  
properties   ăĀĆ ä;ăăŔİéIJĂăĔİèŕAèŁŻăyİæşİèğĉĉŦ'ğéİăİJİăĜ;æŦŕăŏŽăzL'ăL'■ă■şăŦŕİİjŽ

```
class A(metaclass=ABCMeta):
    @property
    @abstractmethod
    def name(self):
        pass

    @name.setter
    @abstractmethod
    def name(self, value):
        pass

    @classmethod
    @abstractmethod
    def method1(cls):
        pass

    @staticmethod
    @abstractmethod
    def method2():
        pass
```

## èőİèőž

æăĜăĜEăžŞăy■æIJL'ă;Ĺăd'ŽĉŦİăĹŕæĹ;èşăăşžĉşzĉŽĎăIJŕæŪzăĀĆcollections  
æİăăİŪăŏŽăzL'ăžEă;Ĺăd'ŽèüşăŏžăŽİăŠŅèŁ■ăžĉăŽİ(ăžŔăĹŪăĀAæŸăăŕĎăĀAéZEăŦĹĉ■L')æIJL'ăĔşĉŽĎæŁ  
numbers   ăžŞăŏŽăzL'ăžEèüşæŦŕă■Ūăŕžèşă(æŦŦ'æŦŕăĀAæŦŏĉĆzæŦŕăĀAæIJL'ĉŦĔæŦŦĉ■L')æIJL'ăĔşĉŽĎăş  
ăžŞăŏŽăzL'ăžEă;Ĺăd'Žèüşİ/OæŞ■ă;IJĉŽŸăĔşĉŽĎăşžĉşzăĀĆ

ă;ăăŦŕăžă;ĤĉŦİéĈĎăŏŽăzL'ĉŽĎæĹ;èşăçşzæİèæL'ğèăŅăŽŦ'éĂŽĉŦİĉŽĎĉşzăĎăĉĀæşëİİjŅă;ŅăĉĆİİjŽ

```
import collections

# Check if x is a sequence
if isinstance(x, collections.Sequence):
    ...

# Check if x is iterable
if isinstance(x, collections.Iterable):
    ...

# Check if x has a size
if isinstance(x, collections.Sized):
```

```
...

# Check if x is a mapping
if isinstance(x, collections.Mapping):
```

år;çõaABCsâRřäzëèõl' æŁŚäznâĹæŮžä;ŁçŽĎâAŽčšzādNæčĂæšëiijŇä;EæYřæŁŚäznâIJläžččăAäy■æI  
åZäâyžPythonçŽĎæIJñet' ÍæYřäyĂéŮlâĹlæĂAçijŮčlŇer■élĀiijŇăĚűçŽōçŽĎaršæYřçzŽă;ăæŽt' âd' ŽçAṭæt' ză  
ăijžăĹűçšzādNæčĂæšëæĹŮèõl' ä;ăäzččăAăRŸăĹŮæŽt' âd' ■æĬčijŇèŁŽæăũăAŽæŮăăijČăžŎèĹ■æIJñæšĆæIJ

## 10.13 8.13 ăōđçŎřæŤřæ■ōæĹăđŇçŽĎçšzādŇçžæĭš

### éŮóécŸ

ă;ăæČšăōŽăzĹæšŘăžŽăIJlăsđæĂğètŇăĀijăyĹéĹcæIJĹ'éŽŘăĹűçŽĎæŤřæ■ōçzŠæđĎăĂĆ

### èğčăEşæŮzæăĹ

ăIJlêŁŽăyĹéŮóécŸăy■iijŇă;ăéIJĂèçAăIJlărzæšŘăžŽăōđă;ŇăsđæĂğètŇăĀijæŮűèŁŽăăŇæčĂæšëăĂĆ  
æĹĂăžëă;ăèçAèĠăōŽăzĹăsđæĂğètŇăĀijăĠ;æŤřiijŇèŁŽçğ■æČĚăEṭăyŇăeIJăăë;ă;ŁçŤĹæŘŘèŁřăŽĹăĂĆ

ăyŇéĹcŽĎăžččăAă;ŁçŤĹæŘŘèŁřăŽĹăōđçŎřăžEăyĂăyĹçšzçzšçšzādŇăŠŇètŇăĀijélŇerAæăEăđüiijŽ

```
# Base class. Uses a descriptor to set a value
class Descriptor:
    def __init__(self, name=None, **opts):
        self.name = name
        for key, value in opts.items():
            setattr(self, key, value)

    def __set__(self, instance, value):
        instance.__dict__[self.name] = value

# Descriptor for enforcing types
class Typed(Descriptor):
    expected_type = type(None)

    def __set__(self, instance, value):
        if not isinstance(value, self.expected_type):
            raise TypeError('expected ' + str(self.expected_type))
        super().__set__(instance, value)

# Descriptor for enforcing values
class Unsigned(Descriptor):
    def __set__(self, instance, value):
        if value < 0:
            raise ValueError('Expected >= 0')
```

```

        super().__set__(instance, value)

class MaxSized(Descriptor):
    def __init__(self, name=None, **opts):
        if 'size' not in opts:
            raise TypeError('missing size option')
        super().__init__(name, **opts)

    def __set__(self, instance, value):
        if len(value) >= self.size:
            raise ValueError('size must be < ' + str(self.size))
        super().__set__(instance, value)

```

èŁŻăŹŹčšzăršæŸřă;ăëĕAăĹŹăžžčŹĐæŤřæ■őăĹăđŊăĹŮčšzăđŊčšzčzščŹĐăšžčăĂăđĐăžžăĹăăĹŮăĂĆăŸŊăĹăčăřšæŸřăĹŤăžŋăőđéŽĚăőŹăžĹčŹĐăŤĐčğ■ăŸ■ăŤŊčŹĐæŤřæ■őčšzăđŊĭjŹ

```

class Integer(Typed):
    expected_type = int

class UnsignedInteger(Integer, Unsigned):
    pass

class Float(Typed):
    expected_type = float

class UnsignedFloat(Float, Unsigned):
    pass

class String(Typed):
    expected_type = str

class SizedString(String, MaxSized):
    pass

```

čĐăăŤŮă;ĕčŤĹĕŹăžŹĕĠăăŹăžĹæŤřæ■őčšzăđŊĭjŊăĹŤăžŋăőŹăžĹăŸĂăŸĹčšzĭjŹ

```

class Stock:
    # Specify constraints
    name = SizedString('name', size=8)
    shares = UnsignedInteger('shares')
    price = UnsignedFloat('price')

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price

```

čĐăăŤŮăĤŊĕŤĹĕŹăŸĹčšzčŹĐăšđæĂğĕŤŊăĂĭjčžĕăĹšĭĭjŊăŤŤăŤŤčŮŤŤăŤŤăšŤŤăžŹăšđæĂğčŹĐĕŤŊăĂĭjĕĹĹ

```

>>> s.name
'ACME'
>>> s.shares = 75
>>> s.shares = -10
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example.py", line 17, in __set__
    super().__set__(instance, value)
  File "example.py", line 23, in __set__
    raise ValueError('Expected >= 0')
ValueError: Expected >= 0
>>> s.price = 'a lot'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example.py", line 16, in __set__
    raise TypeError('expected ' + str(self.expected_type))
TypeError: expected <class 'float'>
>>> s.name = 'ABRACADABRA'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example.py", line 17, in __set__
    super().__set__(instance, value)
  File "example.py", line 35, in __set__
    raise ValueError('size must be < ' + str(self.size))
ValueError: size must be < 8
>>>

```

ěĚŸæIJL'äyÄäzZæŁÄæIJřáRřázěčõÄâŇŮäyŁéİćčŽĎžččăĀıjŇăĚűäy■äyĂçğ■æŸřă;ŁçŦíçśzèčĚěěřăŽí

```

# Class decorator to apply constraints
def check_attributes(**kwargs):
    def decorate(cls):
        for key, value in kwargs.items():
            if isinstance(value, Descriptor):
                value.name = key
                setattr(cls, key, value)
            else:
                setattr(cls, key, value(key))
        return cls
    return decorate

# Example
@check_attributes(name=SizedString(size=8),
                  shares=UnsignedInteger,
                  price=UnsignedFloat)
class Stock:
    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares

```

```
self.price = price
```

ǎŘǎđ'ŮäÿĂçġ■ŮẏâĭŔæŸřăĭçŦlăĚĈşzĭĭŻ

```
# A metaclass that applies checking
class checkedmeta(type):
    def __new__(cls, clsname, bases, methods):
        # Attach attribute names to the descriptors
        for key, value in methods.items():
            if isinstance(value, Descriptor):
                value.name = key
        return type.__new__(cls, clsname, bases, methods)

# Example
class Stock2(metaclass=checkedmeta):
    name = SizedString(size=8)
    shares = UnsignedInteger()
    price = UnsignedFloat()

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price
```

ěőléőž

æIJñèŁĆăĭçŦlăžĒăĭŁăđ'ŽénŸçžġæŁĂæIJřĭĭjŇăŇĚæŇñæŔŔèřřăŽlăĂăuûăĚĚçşzăĂăsuper()
çŽĎăĭçŦlăĂăçşzèĈĚēēŕăŽlăŜŇăĚĈşzăĂĈ äÿ■ǎŔŕèĈĭăIJlêŁŽéĠŇăÿĂäÿĂèŕççžĒăŦăĭjĂæĭēēőšĭĭjŇăĭĒæŸ
ăĭĒæŸřĭĭjŇăĤŝăIJlêŁŽéĠŇèĤŸæŸŕēæĂæŔŔăÿĂäÿŇăĠăäÿlêIJĂēæĂæşlăĎŔçŽĎçĈăĂĈ

éçŮăĚĬĭĭjŇăIJĬ Descriptor âşžçşzăÿ■ăĭăĭjŽçIJŇăĤŕæIJL'äÿl
\_\_set\_\_() æŮżæşŦĭĭjŇă■Ŧ'æşqæIJL'çŽÿăžŦçŽĎ \_\_get\_\_() æŮżæşŦăĂĈ
ăçĈăđIJăÿĂäÿlăŔŔèřřăžĒăžĒæŸřăžŎăžŦăŝĈăőđăĭŇă■ŮăĚÿäÿ■ēŮăŔŮæşŔăÿlăŝđăĒăĠăăĭjçŽĎēŕĭĭjŇēĈ
\_\_get\_\_() æŮżæşŦăĂĈ

æŁ'ĂæIJL'æŔŔèřřăŽlçşzēĈĭæŸŕăşžăžŎăuûăĚĚçşzăĬēăőđçŎŕçŽĎăĂĈăŕŦăçĈ
Unsigned âŜŇ MaxSized èçĀēûşăĚŮăžŮçşžġæŁ'ĤēĠ Typed çşzăuûăĚĚăĂĈ
èĤŽéĠŇăĤl'çŦlăđ'ŽçžġæŁ'ĤæĬēăőđçŎŕçŽÿăžŦçŽĎăĤşēĈĭăĂĈ

ăuûăĚĚçşzçŽĎăÿĂäÿlăŕŦēĭĈēŽĭçŔĒēġççŽĎăIJŕæŮżæŸřĭĭjŇēŕĈçŦĬ
super() âĠĭæŦŕæŮŭĭĭjŇăĭăăžŮăÿ■çşēēĂşçĤ'ŭĉŇşēēĀēŕĈçŦlăşlăÿlăĚŮăĭşçşzăĂĈ
ăĭăēIJĂēæĀēûşăĚŮăžŮçşşçşŦŕĤăŔŎæŁ'■ēĈĭæ■çqăőçŽĎăĭçŦlĭĭjŇăžşăŕŝæŸŕăĤĒăqăŕĤăĭIJăĤ'■ēĈĭăžġçŦĬ

ăĭçŦĬçşzèĈĚēēŕăŽlăŜŇăĚĈşzēĂŽăÿÿăŔŕăžēçŎĂăŇŮăžççăĂăĂĈăÿĤĬăçÿđ'äÿlăĭŇă■Ŕăÿ■ăĭăĭjŽăŔŝ

```
# Normal
class Point:
    x = Integer('x')
    y = Integer('y')
```



```
# Metaclass
class Point(metaclass=checkedmeta):
    x = Integer()
    y = Integer()
```

æL'ÄæIJL'æŮzæşTäy■iijŇçşzècĚéērăZlæŮzæaŁăžTèrěæŸræIJĂçAţæt'zăŞŇæIJĂénŸæŸŬçŽĐăĂĆ  
 éçŮăĚĹiijŇăŏČăzúäy■ăĹĭetŮăzä;ţăĚüăzŮăŮrçŽĐăĹĂæIJrīijŇærTăęCăĚĈşzăĂĆăĚüăñaiijŇēcĚéērăZlă  
 æIJĂăŔŎiijŇēcĚéērăZlăŸĚĈăĹiijăyžæüăăĚéçşzçŽĐăŽăžçæĹĂæIJræİăăŏđçŎŕăŔŇæăŭçŽĐăŤĹăđIJ

```
# Decorator for applying type checking
def Typed(expected_type, cls=None):
    if cls is None:
        return lambda cls: Typed(expected_type, cls)
    super_set = cls.__set__

    def __set__(self, instance, value):
        if not isinstance(value, expected_type):
            raise TypeError('expected ' + str(expected_type))
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls
```

```
# Decorator for unsigned values
def Unsigned(cls):
    super_set = cls.__set__

    def __set__(self, instance, value):
        if value < 0:
            raise ValueError('Expected >= 0')
        super_set(self, instance, value)

    cls.__set__ = __set__
    return cls
```

```
# Decorator for allowing sized values
def MaxSized(cls):
    super_init = cls.__init__

    def __init__(self, name=None, **opts):
        if 'size' not in opts:
            raise TypeError('missing size option')
        super_init(self, name, **opts)

    cls.__init__ = __init__

    super_set = cls.__set__
```

```

def __set__(self, instance, value):
    if len(value) >= self.size:
        raise ValueError('size must be < ' + str(self.size))
    super_set(self, instance, value)

cls.__set__ = __set__
return cls

# Specialized descriptors
@Typed(int)
class Integer(Descriptor):
    pass

@Unsigned
class UnsignedInteger(Integer):
    pass

@Typed(float)
class Float(Descriptor):
    pass

@Unsigned
class UnsignedFloat(Float):
    pass

@Typed(str)
class String(Descriptor):
    pass

@MaxSized
class SizedString(String):
    pass

```

èĚŽçġæŰžâĳŔăŏŽăžĹčŽĎçşzèùşăžŅăĹ■çŽĎæŧĹæđĲăÿĂæăũĳĳŅèĂŅăÿŦæĹġèăŅéĂşăžęăĳĴæŽŦă  
 èőç;őăÿĂăÿĹçőĂă■ŧçŽĎçşzăđŅăşđæĂġçŽĎăĂĳĳĳŅèčĚéěŕăŽĹæŰžâĳŔèęĂæŦăžŅăĹ■çŽĎæũăăĚčçşzçŽĎ  
 çŐŕăĲăĳăăžŦèŕéăžĒăžÿèĢăũşêŕzăőŅăžĒæĲĳŅèĹĆăĚĹčĹăĒăőžăžĒăŔġĳĳş^\_

## 10.14 8.14 ăőđçŐŕèĢăőŽăžĹăőžăŽĲ

éŰőécŸ

ăĳăæČşăőđçŐŕăÿĂăÿĹèĢăőŽăžĹčŽĎçşzæĹăĹæŅşăĒĚç;őçŽĎăőžăŽĲçşzăĹşèČĳĳĳŅæŦăęĆăĹŰèăĹăŠŅ

## èġċăĖşăŮzăăĹ

collections.ăŮŽăzĹ'ăžĖăĹăd'ŽăĹ;ėşăăşžćşzĭĭjŃă;Şă;ăăĈşĕĠăăŮŽăzĹ'ăŮăZăŹĭćşzćŽĎăŮăăŽăăŮĈăŕĤăĖĈă;ăăĈşĕŮĹ'ă;ăĉŽĎćşzăĤŕăŃăĖăăžćĭĭjŃĕĈăŕşĕŮĹ'ă;ăĉŽĎćşzćžġăĹ'ă  
collections.Iterable.ăşăŖĭĭjŽ

```
import collections
class A(collections.Iterable):
    pass
```

ăŷăăĤăĠă;ăĖĬăĖăĤăăŮđĉŎŕ collections.Iterable  
ăĹ'ĂăĬĹ'ćŽĎăĹ;ėşăăŮzăăşŤĭĭjŃăŖăăĹŽăĭjŽăĹĕĕĤŽ:

```
>>> a = A()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't instantiate abstract class A with abstract methods
↳ __iter__
>>>
```

ăĵăăŖĭĕăĤăăŮđĉŎŕ \_\_iter\_\_() .ăŮzăăşŤăŕşăŷăăĭjŽăĹĕĕĤŽăžĖă(ăŖĈĖăĈĈ4.2ăŖŤŃ4.7ăŕŖĕĹĈ)ăĂĈ  
ăĵăăŖăŕăžăăĤĕŕŤĉĬăăŮăăŮđăĵŃăŃŮăŷăăŷăŕăŕžĕşăĭĭjŃăĬĭĕĤŽĕŕŕăŖŖĉđ'žăŷăăŖăŕăžăăĹ;ăĹŕĖĬăĖăĤăăŮĈăŕĤăĖĈă

```
>>> import collections
>>> collections.Sequence()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't instantiate abstract class Sequence with abstract
↳ methods \
__getitem__, __len__
>>>
```

ăŷŃĖĬăăŸŕăŷăăŷăĴăăŮăăŤĉŽĎĉđ'žă;ŃĭĭjŃĉzġăĹ'ăĖăĠăŷăĹĖĬăSequenceăĹ;ėşăăşzĭĭjŃăžăŷăŷăŤăăŮđĉŎŕăĖĬă

```
class SortedItems(collections.Sequence):
    def __init__(self, initial=None):
        self._items = sorted(initial) if initial is not None else []

    # Required sequence methods
    def __getitem__(self, index):
        return self._items[index]

    def __len__(self):
        return len(self._items)

    # Method for adding an item in the right location
    def add(self, item):
        bisect.insort(self._items, item)
```

```

items = SortedItems([5, 1, 3])
print(list(items))
print(items[0], items[-1])
items.add(2)
print(list(items))

```

řřäzëçIJŇáĹřřijŇSortedItemsèùšæŽöéĂŽçŽĎžŘáĹŮæšqázĂžžĹäyď'æăüüijŇæŤræŇAæĹ'ĂæIJĹ'ăyÿç  
 èĚŽéĠŇéĬcä;ĤçŤĬáĹrăžE bisect æĭqăĬŮüijŇăôČæŸrăyĂăyĤăIJăŎŠăžŘáĹŮèqăy■æŔŠăĚčăĚČçť'ăçŽ

## èóĬèőž

ä;ĤçŤĬ collections äy■çŽĎæĹ;èšqăšžçšžăŘrăžëçqăöăĬă;ăèĠăôŽăžĹ'çŽĎăôžăŽĬăôđçŎřăžEæĹ'ĂæĹ  
 ä;ăçŽĎèĠăôŽăžĹ'ăôžăŽĬăijŽæžăèűşăď'ġéČĬăĹEçşžăďŇăčĂăšééIJăèçAřijŇăçČăyŇăĹ'Ăçď'žüijŽ

```

>>> items = SortedItems()
>>> import collections
>>> isinstance(items, collections.Iterable)
True
>>> isinstance(items, collections.Sequence)
True
>>> isinstance(items, collections.Container)
True
>>> isinstance(items, collections.Sized)
True
>>> isinstance(items, collections.Mapping)
False
>>>

```

collections äy■ă;Ĺăď'ŽæĹ;èšqçşžăijŽăyžăyĂăžŽăyÿèġAăôžăŽĬăš■ă;IJăŔŔă;ŽézŸèôď'çŽĎăôđçŎ  
 èĚŽæăüăyĂæĬcä;ăăŔĬéIJăèçAăôđçŎŔéČčăžŽă;ăæIJăæĎšăĚť'èüčçŽĎæŮžæşŤă■şăŔăăĂČăAĠèôġă;ăçŽĎçşž  
 collections.MutableSequence üijŇăçČăyŇüijŽ

```

class Items(collections.MutableSequence):
    def __init__(self, initial=None):
        self._items = list(initial) if initial is not None else []

    # Required sequence methods
    def __getitem__(self, index):
        print('Getting:', index)
        return self._items[index]

    def __setitem__(self, index, value):
        print('Setting:', index, value)
        self._items[index] = value

    def __delitem__(self, index):
        print('Deleting:', index)
        del self._items[index]

```

```
def insert(self, index, value):
    print('Inserting:', index, value)
    self._items.insert(index, value)

def __len__(self):
    print('Len')
    return len(self._items)
```

æCædIIj;ääLZåzz Items çŽDåóðä;NrijNä;äaijZåRŞçŎřåóCæTræŊAåGääžŎæL'ĂæIJL'çŽDæăyåŁČå  
äyŊéÍcæYřä;ŁçŤlæijŤçd'žiižŽ

```
>>> a = Items([1, 2, 3])
>>> len(a)
Len
3
>>> a.append(4)
Len
Inserting: 3 4
>>> a.append(2)
Len
Inserting: 4 2
>>> a.count(2)
Getting: 0
Getting: 1
Getting: 2
Getting: 3
Getting: 4
Getting: 5
2
>>> a.remove(3)
Getting: 0
Getting: 1
Getting: 2
Deleting: 2
>>>
```

æIJnårRèLCàRlæYřåržPythonæL;èsaçşzåŁşèČ;çŽDæLZçăŮaijŤçŎL'ăĂCnumbers  
æÍaaiŮæRŘä;ŽäžEäyĂäyŁçşzäijijçŽDëuşæŤt'æŤřçşzådŊçŽyåĚşçŽDæL;èsaçşzådŊéZEăŘLăĂC  
årRřäžæåRCèĂC8.12årRèLCæIěæđĐéĂæŽt'åd'ŽèGłåóŽázL'æL;èsaşşzçşzāĂC

## 10.15 8.15 åsdæĂgçŽDäzççŘEèóÉúŎ

### éŮóécY

ä;äæČşårEæşŘäyłåóðä;ŊçŽDåsdæĂgèóéúŎäzççŘEăŁřăĚéČlăŘeäyĂäyłåóðä;Ŋäy■ăŎžiijŊçŽóçŽDă

## èġċaEşæŪzæąŁ

ċŏĂă■Tæİèèrt'ijjNăzċċŘEæYřäyĂċġ■ċijŪċlNæÍaăijRiijNăŏČăřEæŞŘäyŁæŞ■ă;IJè;ñċġzċzZăRċăd' ŪăyĂæIJĂċŏĂă■TċŽĎă;ċăijRăRřèĈ;æYřăČRăyNéÍċèŁZæăüijŽ

```
class A:
    def spam(self, x):
        pass

    def foo(self):
        pass

class B1:
    """ċŏĂă■TċŽĎăzċċŘE"""

    def __init__(self):
        self._a = A()

    def spam(self, x):
        # Delegate to the internal self._a instance
        return self._a.spam(x)

    def foo(self):
        # Delegate to the internal self._a instance
        return self._a.foo()

    def bar(self):
        pass
```

ăċČădIJăzĚăzĚărsăyd'ăyŁæŪzæşTéIJĂèċAăzċċŘEijjNéĈăzŁăČRèŁZæăüăĚăřsèüşăd'şăžEăĂĈă;EæYéĈăzŁă;ŁċŤl \_\_getattr\_\_() æŪzæşTæŁŪèŏyæŁŪæŽt'ăċ;ăžZiijŽ

```
class B2:
    """ă;ŁċŤl __getattr__ċŽĎăzċċŘEiijNăzċċŘEæŪzæşTæřTèċČăd'ŽæŪüăĂŽ"""

    def __init__(self):
        self._a = A()

    def bar(self):
        pass

    # Expose all of the methods defined on class A
    def __getattr__(self, name):
        """
        → "èŁZăyŁæŪzæşTăIJlèŏŁéŪŏċŽĎătttributeăy■ă■YăIJlċŽĎăŪüăĂŽèċnèřĈċŤl
        the __getattr__() method is actually a fallback method
        that only gets called when an attribute is not found"""
        return getattr(self._a, name)
```

\_\_getattr\_\_ æŪzæşTæYřăIJlèŏŁéŪŏătttributeăy■ă■YăIJlċŽĎăŪüăĂŽèċnèřĈċŤliijNă;ŁċŤlæijTċd'ž



éÁŽèŁĠēĠāōŽāzŁ'āsđæĀġēōŁēŮōæŮzæşŦiijŊăĵăăŦřăžēċŦlăy■ăŦŊæŮzăijŦēĠāōŽāzŁ'ăžċŦŦĚşşzēăŊ

## èőléőž

ăžċŦŦĚşşzæIJŁ'æŮŭăĂŽăŦřăžēăĴIJăyžċžġæŁ'ŁċŽĐæŽŁăžċæŮzæăŁăĂĈăĴŊăċŦiijŊăyĂăyŁċōĂă■ŦċŽĐ

```
class A:
    def spam(self, x):
        print('A.spam', x)
    def foo(self):
        print('A.foo')

class B(A):
    def spam(self, x):
        print('B.spam')
        super().spam(x)
    def bar(self):
        print('B.bar')
```

ăĴċŦlăžċŦŦĚşşŽĐŦŦiijŊăŦŦæŮŦăyŊēĴċēŁŽæăŭiijŽ

```
class A:
    def spam(self, x):
        print('A.spam', x)
    def foo(self):
        print('A.foo')

class B:
    def __init__(self):
        self._a = A()
    def spam(self, x):
        print('B.spam', x)
        self._a.spam(x)
    def bar(self):
        print('B.bar')
    def __getattr__(self, name):
        return getattr(self._a, name)
```

ăĴŞăōđċŦŦăžċŦŦĚşşŦĚăĴăăijŦæŮŭiijŊēŁŮæIJŁ'ăžŽċžĚēŁĈēIJĂēċĂæşĴăēĐŦăĂĈ  
éċŮăĒŦiijŊ\_\_getattr\_\_()ăōđēŽĚæŮŦăyĂăyŦăŦŦăđ'ĠæŮzæşŦiijŊăŦŦæIJŁ'ăIJĴăşđæĂġăy■ă■ŮăIJĴăŮŮ  
ăŽăă■đ'ŦiijŊăċŦēđIJăžċŦŦĚşşzăōđăĴŊăIJŋēžŋæIJŁ'ēŁŽăyŦăşđæĂġċŽĐŦŦiijŊēĈċăžŦăy■ăijŽēġēăŦŦēŁŽăyŦă  
ăŦēăđ'ŮiijŊ\_\_setattr\_\_()ăŦŦ\_\_delattr\_\_()éIJĂēċĂēĴăđ'ŮċŽĐē■ŦăşŦăēĴăŊăŦăŦăŦăŦăžċŦŦĚăōđ  
\_objċŽĐăşđæĂġăĂĈăyĂăyŦăĂŽăyŮċŽĐċžēăōŽăŮŦăŦăžċŦŦĚċĈăžŦăy■ăžēăyŊăŦŦŦşċžċ  
\_ăijĂăđ't'ċŽĐăşđæĂġ(ăžċŦŦĚşşzăŦŦăŽt'éIJşēċŋăžċŦŦĚşşzċŽĐăĒăŦŦşăşđæĂġ)ăĂĈ

ēŁŮæIJŁ'ăyĂċŦŦēIJĂēċĂæşĴăēĐŦċŽĐăŮŦiijŊ\_\_getattr\_\_()  
ărzăžŦăđ'ġēĴăŦăŦăžēăŦŦăyŊăŦŦŦşċžċ(ŮăijĂăġŊăŦŦŮċžŞăŦĴċŽĐăşđæĂġăžŮăy■éĂĈċŦŦăĂĈ  
ăŦŦăċŦiijŊēĂĈċēŽŦăċăyŮċŽĐċşzŭiijŽ



```
class ListLike:
    """__getattr__
    ↳âŕžăžŎăŔŇăŷŇăĹŤçžŁăıjĂăğŇăŤŇçžŞăŕçžŽĐăŨžăşŤăŸŕăŷ■èĈ;çŤĺçŽĐiıjŇéIJĂèçAăŷĂăŷłăŷł
    ↳"""

    def __init__(self):
        self._items = []

    def __getattr__(self, name):
        return getattr(self._items, name)
```

ăĕĈăđIJăŸŕăĹŽăžăŷĂăŷĹListLikeăŕžăşăııjŇăıjŽăŔŤçŎŕăŏĈăŤŕăŇAăŽŏéĂŽçŽĐăĹŨèăĹăŨžăşŤiıjŇăĴăŸŕăŇŤăŷ■ăŷ■ăŷŕăŇAĹen()ăĂAăĖĈçŤ'ăăşăăĹŷç■ĹăĂĈăŷŇăĕĈiıjŽ

```
>>> a = ListLike()
>>> a.append(2)
>>> a.insert(0, 1)
>>> a.sort()
>>> len(a)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: object of type 'ListLike' has no len()
>>> a[0]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'ListLike' object does not support indexing
>>>
```

ăŷžăžĖĕŏĹăŏĈăŤŕăŇAĕŁăžžZăŨžăşŤiıjŇăĴăăŁĖĕăžăĹŇăĹĹçŽĐăŏđçŎŕĕŁăžžZăŨžăşŤăžççŔĖiıjŽ

```
class ListLike:
    """__getattr__
    ↳âŕžăžŎăŔŇăŷŇăĹŤçžŁăıjĂăğŇăŤŇçžŞăŕçžŽĐăŨžăşŤăŸŕăŷ■èĈ;çŤĺçŽĐiıjŇéIJĂèçAăŷĂăŷłăŷł
    ↳"""

    def __init__(self):
        self._items = []

    def __getattr__(self, name):
        return getattr(self._items, name)

    # Added special methods to support certain list operations
    def __len__(self):
        return len(self._items)

    def __getitem__(self, index):
        return self._items[index]

    def __setitem__(self, index, value):
        self._items[index] = value
```

```
def __delitem__(self, index):  
    del self._items[index]
```

11.8ārĖĹĆēĲYæIJL'äyÄäyġāIJġēĲJġĹNæŪzæşTĕřČġTġĲŌrācČäy■äġĲTġāzčġŖEġZDäġNā■ŘāĲĆ

## 10.16 8.16 āĲJġśzäy■āŌŽāzL'ād'ŽäyġæđDēÄāāŽĲ

éŬŌécŸ

äġāæČşāŌđġŌřäyÄäyġśzġġjNéŽd'āžEäġĲTĲ  
æŪzæşTġd'ŪġġNēĲYæIJL'āEŪāzŪæŪzāġjRāŖrāzēāĲġāġNāNŪāŌČāĲĆ  
\_\_init\_\_()

èġčāEşæŪzæāĲ

äyžāžEāŌđġŌřād'ŽäyġæđDēÄāāŽĲġġjNäġāēĲJĲēĲäġĲĲTġāĲŖġśzæŪzæşTāĲĆäġNāēČġġjŽ

```
import time  
  
class Date:  
    """æŪzæşTäyÄġġjžāġĲĲTġśzæŪzæşT"""  
    # Primary constructor  
    def __init__(self, year, month, day):  
        self.year = year  
        self.month = month  
        self.day = day  
  
    # Alternate constructor  
    @classmethod  
    def today(cls):  
        t = time.localtime()  
        return cls(t.tm_year, t.tm_mon, t.tm_mday)
```

ġŽĲ'æŌēĲČġTġśzæŪzæşTā■şāŖĲġġjNäyNēĲæŸrāġĲĲTġd'žāġNġġjŽ

```
a = Date(2012, 12, 21) # Primary  
b = Date.today() # Alternate
```

èŌĲēŌž

ġśzæŪzæşTġZDäyÄäyġäyžēēĲĲTĲĲĲTāŖsæŸrāŌŽāzL'ād'ŽäyġæđDēÄāāŽĲāĲĆāŌČæŌēāRŪäyÄäyġ  
class äġIJäyžġñnāyÄäyġāŖČæŲĲ(Ĳs)āĲĆ äġāāžTĕřēæşĲæđRāĲŖāžEēĲŽäyġśzēcġĲĲĲēāĲZāzžāzūēĲTāZda

```
class NewDate(Date):  
    pass
```

```
c = Date.today() # Creates an instance of Date (cls=Date)
d = NewDate.today() # Creates an instance of NewDate (cls=NewDate)
```

## 10.17 8.17 $\text{aLZ}\ddot{\text{a}}\ddot{\text{z}}\ddot{\text{a}}\ddot{\text{y}}\blacksquare\text{er}\check{\text{C}}\check{\text{t}}\text{I}\text{init}\text{ae}\acute{\text{U}}\text{z}\text{ae}\text{ŧ}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$

### $\text{e}\acute{\text{U}}\acute{\text{o}}\acute{\text{e}}\acute{\text{c}}\check{\text{Y}}$

$\check{\text{a}}\check{\text{j}}\check{\text{a}}\text{ae}\check{\text{C}}\text{sa}\acute{\text{L}}\check{\text{Z}}\text{a}\ddot{\text{z}}\ddot{\text{a}}\ddot{\text{y}}\check{\text{A}}\check{\text{a}}\text{y}\text{la}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}\text{ij}\check{\text{N}}\text{a}\text{E}\text{ae}\check{\text{Y}}\text{ra}\check{\text{y}}\check{\text{N}}\text{ae}\text{IJ}\check{\text{Z}}\check{\text{c}}\check{\text{z}}\check{\text{T}}\text{e}\text{f}\check{\text{G}}\text{ae}\text{L}'\check{\text{g}}\text{ea}\check{\text{N}} \quad \_\_\text{init}\_\_\text{()}$   
 $\text{ae}\acute{\text{U}}\text{z}\text{ae}\text{ŧ}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$

### $\text{e}\check{\text{g}}\check{\text{c}}\check{\text{a}}\text{E}\check{\text{s}}\text{ae}\acute{\text{U}}\text{z}\text{ae}\text{a}\acute{\text{L}}$

$\text{a}\acute{\text{R}}\text{ra}\check{\text{z}}\text{e}\acute{\text{e}}\check{\text{A}}\check{\text{Z}}\text{e}\text{f}\check{\text{G}} \_\_\text{new}\_\_\text{()}$   $\text{ae}\acute{\text{U}}\text{z}\text{ae}\text{ŧ}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$   $\text{ae}\text{IJ}\text{a}\acute{\text{L}}\text{I}\text{a}\check{\text{g}}\check{\text{N}}\text{a}\check{\text{N}}\check{\text{U}}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$   $\text{a}\check{\text{A}}\check{\text{C}}\check{\text{a}}\check{\text{z}}\check{\text{N}}\text{ae}\check{\text{C}}\text{e}\check{\text{A}}\check{\text{C}}\text{e}\check{\text{Z}}\check{\text{S}}\check{\text{a}}$

```
class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day
```

$\check{\text{a}}\check{\text{y}}\check{\text{N}}\text{e}\text{I}\check{\text{c}}\text{ae}\text{ij}\check{\text{T}}\check{\text{c}}\text{d}'\check{\text{z}}\text{ae}\check{\text{C}}\check{\text{a}}\check{\text{z}}\check{\text{T}}\check{\text{a}}\text{y}\blacksquare\text{er}\check{\text{C}}\check{\text{t}}\text{I} \_\_\text{init}\_\_\text{()}$   $\text{ae}\acute{\text{U}}\text{z}\text{ae}\text{ŧ}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$   $\text{ae}\text{IJ}\text{a}\acute{\text{L}}\text{I}\text{a}\check{\text{g}}\check{\text{N}}\text{a}\check{\text{N}}\check{\text{U}}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$

```
>>> d = Date.__new__(Date)
>>> d
<__main__.Date object at 0x1006716d0>
>>> d.year
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'Date' object has no attribute 'year'
>>>
```

$\check{\text{c}}\check{\text{z}}\check{\text{S}}\check{\text{ae}}\text{d}\text{IJ}\text{a}\acute{\text{R}}\text{ra}\check{\text{z}}\text{e}\check{\text{c}}\text{IJ}\check{\text{N}}\text{a}\acute{\text{L}}\text{r}\check{\text{ij}}\check{\text{N}}\text{e}\text{f}\check{\text{Z}}\check{\text{a}}\text{y}\text{I}\text{Date}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$   $\check{\text{N}}\check{\text{c}}\check{\text{Z}}\check{\text{D}}\text{a}\acute{\text{o}}\check{\text{d}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$   $\check{\text{A}}\check{\text{g}}\text{year}\text{e}\text{f}\check{\text{Y}}\check{\text{a}}\text{y}\blacksquare\blacksquare\check{\text{Y}}\text{a}\text{IJ}\text{r}\check{\text{ij}}\check{\text{N}}\text{ae}\text{L}'\check{\text{A}}\check{\text{a}}\check{\text{z}}\check{\text{e}}\check{\text{a}}\check{\text{z}}\check{\text{N}}$   $\check{\text{a}}\text{e}\text{IJ}\check{\text{A}}\text{e}\text{e}\text{A}\text{ae}$

```
>>> data = {'year':2012, 'month':8, 'day':29}
>>> for key, value in data.items():
...     setattr(d, key, value)
...
>>> d.year
2012
>>> d.month
8
>>>
```

## èõléõž

ā;ŠæŁŚazñāIJlāR■āžRāŁŮāržēsāæŁŮēĀĒāōđçŎræšŘäylčszæŮzæšTæđDéĀāāĜ;æTṛæŮúéIJĀēçAçzTē  
\_\_init\_\_() æŮzæšTæĪēāŁZāžžāržēsāĀĆ ä;NāçCīijNāržäžŎäyŁéĪčŽĐDateæĪēēōšīijNæIJL'æŮūāĀŽā;ā  
today() īijŽ

```
from time import localtime

class Date:
    def __init__(self, year, month, day):
        self.year = year
        self.month = month
        self.day = day

    @classmethod
    def today(cls):
        d = cls.__new__(cls)
        t = localtime()
        d.year = t.tm_year
        d.month = t.tm_mon
        d.day = t.tm_mday
        return d
```

āŘNæūīijNāIJlā;āāR■āžRāŁŮāŃŮJSONæTṛæ■ōæŮūāžğçTšäyĀäyĪæçCäyNçŽĐā■ŮāĒyāržēsāīijŽ

```
data = { 'year': 2012, 'month': 8, 'day': 29 }
```

æçCæđIJā;āæČšārĒāōČē;ñæ■céæŁŘäyĀäyĪDateçszādNāōđä;NīijNārřäžēä;čçTĪäyŁéĪčŽĐæŁĀæIJřāĀĆ

ā;Šā;æĀŽēŁĜēŁŽçğ■ēĪđäyÿēğDæŮzāijRæĪēāŁZāžžāōđä;NçŽĐæŮūāĀŽīijNæIJĀāē;äy■ēçAçŽt' æŎēā  
āŘēāŁŽçŽĐērīijNāçCæđIJēŁZäyĪčszä;čçTĪāžĒ \_\_slots\_\_ āĀproperties āĀde-  
scriptors æŁŮāĒūāžŮēnŸçžğæŁĀæIJřçŽĐæŮūāĀŽāžççāĀāršāijŽād'sæTĪāĀĆ  
ēĀNēŁZæŮūāĀŽā;čçTĪ setattr() æŮzæšTāijŽēōĪ'ā;āçŽĐāžççāĀārŸā;ŮæŽt' āŁāēĀŽçTĪāĀĆ

## 10.18 8.18 āĪ'çTĪMixinsæĪ'āsTçszāŁšèĈ;

### éŮōécŸ

ā;āæIJL'ā;Łād'ŽæIJL'çTĪčŽĐæŮzæšTīijNæČšā;čçTĪāōČāžñæĪæĪ'āsTāĒūāžŮçszçŽĐāŁšèĈ;āĀĆā;Ēā  
āŽāæ■đ'ā;āāy■ēČ;çōĀā■TçŽĐārĒēŁZāžžæŮzæšTæTĪāĒēäyĀäyĪāšžçszīijNçĐūāŘŎēčnāĒūāžŮçszçžğæŁŁā

### èğçĀĒşæŮzæāĪ

ēĀŽāÿÿā;Šā;āæČšēĜĪāōŽāžŁçszçŽĐæŮūāĀŽāijŽççřäyŁēŁZāžžēŮōécŸāĀĆāŘrēĈ;æŸræšŘäyĪāžŠæŘ  
ā;āāŘřäžēāĪ'çTĪāōČāžñæĪæđDéĀāā;āēĜĪāūsçŽĐçszāĀĆ

āĀĜēōç;ā;āæČşæĪ'āsTæŸārĎāržēsāīijNçžŽāōČāžñæūzāŁāæŮēāĒŮāĀāTṛäyĀæĀğēōç;ōāĀAçszādĪ

```

class LoggedMappingMixin:
    """
    Add logging to get/set/delete operations for debugging.
    """
    __slots__ = ()

    def __getitem__(self, key):
        print('Getting ' + str(key))
        return super().__getitem__(key)

    def __setitem__(self, key, value):
        print('Setting {} = {}'.format(key, value))
        return super().__setitem__(key, value)

    def __delitem__(self, key):
        print('Deleting ' + str(key))
        return super().__delitem__(key)

class SetOnceMappingMixin:
    """
    Only allow a key to be set once.
    """
    __slots__ = ()

    def __setitem__(self, key, value):
        if key in self:
            raise KeyError(str(key) + ' already set')
        return super().__setitem__(key, value)

class StringKeysMappingMixin:
    """
    Restrict keys to strings only
    """
    __slots__ = ()

    def __setitem__(self, key, value):
        if not isinstance(key, str):
            raise TypeError('keys must be strings')
        return super().__setitem__(key, value)

```

æŭũăĚĕçśżĕČ;æšæIĴL'ăôďă;ŇăŘŸĕĜŘiiĵŇăŽăăÿžčŽt'æŎěăôďă;ŇăŇŮæũũăĚĕçśżæšæIĴL'ăžžă  
 ăŎČăžŇăŸřČŦăĭĕĕĂŽĕĚĜăď'ŽčžăĽ'ĤăĭăŠŇăĚũăžŮăŸăăřďăřžĕşăæũũăĚĕă;ĤčŦĭçŽďăĂČă;ŇăĕČriĵŽ

```

class LoggedDict(LoggedMappingMixin, dict):
    pass

```

```
d = LoggedDict()
```

```

d['x'] = 23
print(d['x'])
del d['x']

from collections import defaultdict

class SetOnceDefaultDict(SetOnceMappingMixin, defaultdict):
    pass

d = SetOnceDefaultDict(list)
d['x'].append(2)
d['x'].append(3)
# d['x'] = 23 # KeyError: 'x already set'

```

èŁŻäÿłä;Ńă■Řäÿ■rijŃăŔřäzèçIJŃăĹŕæüüăĚèçşzèùşăĚüăzŮăüşă■ŸăIJčŽDçşz(æŕŤæĈdictăĂđdefaultd  
çzŞăŔĹăŔŔŔăŕşèĈ;ăŔŜăŇăæ■čäÿÿăĹşæŤĹăžĚăĂĈ

## èőléőž

æüüăĚèçşzăIJăăĜăĜĚăžŞäÿ■ăĹăđ'ŽăIJŕæŮzéĈ;ăĜzçŎŕèĹĜrijŃăĚŽăÿÿéĈ;æŸŕçŤĹăĹăăĈŔäÿĹéĹcéĈ  
ăŕĈăžŋăžşæŸŕăđ'ŽçzğæĹ'ĹçŽĎäÿĂäÿłäÿzèçAçŤĹéĂŤăĂĈæŕŤăçĈrijŃă;Şă;ăçijŮăĚŽç;ŞçzIJăžççăĂăŮăăĂŽ  
ă;ăäijŽçzŔăÿÿă;ĹçŤĹ socketserver æĹăăĹŮăÿ■çŽĎ ThreadingMixin  
æĹççzŽăĚüăžŮç;ŞçzIJçŽÿăĚşçşzăçđăĹăăđ'ŽçžĹçĹŃăŤŕæŃăăĂĈ  
ăĹŃăçĈrijŃăÿŃéĹæŸŕäÿĂäÿłăđ'ŽçžĹçĹŃçŽĎXML-RPCæIJ■ăĹăijŽ

```

from xmlrpc.server import SimpleXMLRPCServer
from socketserver import ThreadingMixin
class ThreadedXMLRPCServer(ThreadingMixin, SimpleXMLRPCServer):
    pass

```

ăŔŃăŮăăIJăÿĂăžŽăđ'ğăđŃăžŞăŤŃăæĹăđŮăÿ■ăžşăijŽăŔŜçŎŕæüüăĚèçşzçŽĎă;ĹçŤĹijŃçŤĹéĂŤăŔŃăæ  
ăŕžăžŎæüüăĚèçşzrijŃăIJĹăĜăçĈzéIJăèçAèőŕă;ŔăĂĈéçŮăĚĹăŸŕijŃăüüăĚèçşzäÿ■èĈ;çŽŤ æŎèèçŋăŕ  
ăĚüăæŋrijŃăüüăĚèçşzæşqæIJĹèĜĹăüşçŽĎçĹŮăĂăăĹæAŕrijŃăžşăŕşæŸŕèŕŤ'ăŕĈăžŋăžŮăşqæIJĹăŕŽăžĹ'  
\_\_init\_\_() æŮžæşŤrijŃăžŮăÿŤăşqæIJĹăŕđă;ŃăşđăĂğăĂĈ  
èĹŽăžşæŸŕäÿžăžĂăžĹăĹŤăžŋăIJăÿĹéĹæŸŎçăŕăŕŽăžĹ'ăžĚ \_\_slots\_\_ = () äĂĈ  
èĹŸăIJĹăÿĂçğ■ăŕđçŎŕæüüăĚèçşzçŽĎăŮžăijŔăŕşæŸŕă;ĹçŤĹçşzèçĚèçŕăŽĹijŃăçCäÿŃăĹ'Ăçđ'žijŽ

```

def LoggedMapping(cls):
    """çŋŋăžŃçç■ăŮžăijŔrijŽă;ĹçŤĹçşzèçĚèçŕăŽĹ"""
    cls_getitem = cls.__getitem__
    cls_setitem = cls.__setitem__
    cls_delitem = cls.__delitem__

    def __getitem__(self, key):
        print('Getting ' + str(key))
        return cls_getitem(self, key)

```

```

def __setitem__(self, key, value):
    print('Setting {} = {}'.format(key, value))
    return cls_setitem(self, key, value)

def __delitem__(self, key):
    print('Deleting ' + str(key))
    return cls_delitem(self, key)

cls.__getitem__ = __getitem__
cls.__setitem__ = __setitem__
cls.__delitem__ = __delitem__
return cls

```

```

@LoggedMapping
class LoggedDict(dict):
    pass

```

èfŽäyŁæŦŁædIJèùşázNâL■çŽDæYřäyÄæăüçŽDrijNèÄNäyTäy■ăE■éIJĂèçAă;ŁçŦÍăd'ŽçžgæLŁăžEăÄ  
 âŦCèĂĈ8.13ârRèŁCæşççIJNæŽt'ăd'ŽæuûăĖčşzâŞNçşžèçĖččrăZÍçŽDă;Nă■RăĂĈ

## 10.19 8.19 ăôđçŎřçŁúæĀAřzèşqæŁŰèĂĖçŁúæĀAæIJž

éŰóécŸ

ä;ăæĈşăôđçŎřäyĂäyŁçŁúæĀAæIJžæŁŰèĂĖæYřăIJăy■ăRŦçŁúæĀAăyNæLğèqNæŞ■ă;IJžDărzèşqii

èğčăEşşæŰzæqŁ

âIJăŁŁăd'ŽçÍNăžRăy■rijNæIJL'ăžZărzèşqäijŽæăžæ■ŏçŁúæĀAçŽDăy■ăRŦæİæLğèqNăy■ăRŦçŽDæŞ

```

class Connection:
    """æŽŏéĂžæŰzæqŁii jNăě;ăd'ŽăyŁăŁd'æŰ■èŦ■ăŦëii jNæŦŁçŎĞă;ŎăyŦ~"""

    def __init__(self):
        self.state = 'CLOSED'

    def read(self):
        if self.state != 'OPEN':
            raise RuntimeError('Not open')
        print('reading')

    def write(self, data):
        if self.state != 'OPEN':
            raise RuntimeError('Not open')
        print('writing')

    def open(self):

```

```

    if self.state == 'OPEN':
        raise RuntimeError('Already open')
    self.state = 'OPEN'

def close(self):
    if self.state == 'CLOSED':
        raise RuntimeError('Already closed')
    self.state = 'CLOSED'

```

ẽƒŽæăăăĚŽæIJL'ăĭŁăđ'ŽćijžćĆzīijNéeŮăĚŁæŸřăžćăĂăđ'ĭăđ'■æĬCăžĒīijNăēĭăđ'ŽćŽĐăĭăžăŮăĬđ'æŮ  
 ăŽăăŸăăŸăăžăăŸăăŸăăġĂçŽĐă\$■ăĭJăērŤăçĆread()ăĂăwrite()ăērŖăñăăL'ğăăNăĬ'■éĈĭéIJăēăĂăL'ğăăNăčĂăŸă  
 äŸĂăŸĭăŽŤăēĭćŽĐăĬđăŸŤăŸřăŸăērŖăŸĭćĬŮăĂăăŏŽăžĬ'äŸĂăŸĭăŕžăēăŕijŽ

```

class Connection1:
    """æŮřæŮžæăĬăĂŤăĂŤăŕžăērŖăŸĭćĬŮăĂăăŏŽăžĬ'äŸĂăŸĭćŝž"""

    def __init__(self):
        self.new_state(ClosedConnectionState)

    def new_state(self, newstate):
        self._state = newstate
        # Delegate to the state class

    def read(self):
        return self._state.read(self)

    def write(self, data):
        return self._state.write(self, data)

    def open(self):
        return self._state.open(self)

    def close(self):
        return self._state.close(self)

# Connection state base class
class ConnectionState:
    @staticmethod
    def read(conn):
        raise NotImplementedError()

    @staticmethod
    def write(conn, data):
        raise NotImplementedError()

    @staticmethod
    def open(conn):
        raise NotImplementedError()

```



```

    @staticmethod
    def close(conn):
        raise NotImplementedError()

# Implementation of different states
class ClosedConnectionState(ConnectionState):
    @staticmethod
    def read(conn):
        raise RuntimeError('Not open')

    @staticmethod
    def write(conn, data):
        raise RuntimeError('Not open')

    @staticmethod
    def open(conn):
        conn.new_state(OpenConnectionState)

    @staticmethod
    def close(conn):
        raise RuntimeError('Already closed')

class OpenConnectionState(ConnectionState):
    @staticmethod
    def read(conn):
        print('reading')

    @staticmethod
    def write(conn, data):
        print('writing')

    @staticmethod
    def open(conn):
        raise RuntimeError('Already open')

    @staticmethod
    def close(conn):
        conn.new_state(ClosedConnectionState)

```

äyÑéÍæÝrä;ŁçŤläijŤčd'ŽüijŽ

```

>>> c = Connection()
>>> c._state
<class '__main__.ClosedConnectionState'>
>>> c.read()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example.py", line 10, in read

```

```

        return self._state.read(self)
    File "example.py", line 43, in read
        raise RuntimeError('Not open')
RuntimeError: Not open
>>> c.open()
>>> c._state
<class '__main__.OpenConnectionState'>
>>> c.read()
reading
>>> c.write('hello')
writing
>>> c.close()
>>> c._state
<class '__main__.ClosedConnectionState'>
>>>

```

## èõìèõž

æĈædĪăžĉăĀăy■ăĜžĉŎřăd'ĭăd'ŽĉŽĎăĭăžŭăĽd'æŮ■èr■ăRĕĉŽĎèrĭijNăžĉăĀărsăijŽăRŸăĭŮéŽĭăžè  
 èĚŽéĜNĉŽĎèĝĉăEşæŮžæąĹæŸřăŕEĉŕRăyĭĉĹŭæĀĀæĹĭăŔŮăĜžæĭeăōŽăžĹæĹŔăyĀăyĭĉşăăĀĈ

èĚŽéĜNĉĪJNăyĹăŎžæĪJĹĉĈžăèĜæĀĭijNăŕRăyĭĉĹŭæĀĀăŕžèşăĉĈĭăŔĭæĪJĹéĭŽæĀĀæŮžæşŤĭijNăžŭæş  
 ăōđéŽĒăyĹĭijNăĹĀæĪJĹĉĹŭæĀĀăĤăæĀŕéĈĭăŔĭă■ŸăĈĭăĪĪĭ Connection  
 ăōđăĭNăy■ăĀĈăĪJĭăşžĉşăžăy■ăōŽăžĹĉŽĎ NotImplementedError  
 æŸŕăyžăžEĉăōăĤăă■ŔĉşăăōđĉŎřăžEĉŽŸăžŤĉŽĎæŮžæşŤăĀĈ èĚŽéĜNăĭăæĹŮèōyèĤŸæĈşăĭĉĈŤĭ8.12ăŕŔèĹĈ

èōĭèōăăĭăĭjRăy■æĪJĹăyĀĉĝ■ăĭăĭjRăŕŕĭĉĹŭæĀĀăĭăĭjRĭijNăŕŽăyĀăŕŔèĹĈĉŎŮæŸŕăyĀăyĭăĹĭă■ăĭ

## 10.20 8.20 éĂŽè£Ĝă■ŮĉņęăyşèŕĈĉŤĭăŕžèşăæŮžæşŤ

### éŮŏécŸ

äĭăæĪJĹăyĀăyĭă■ŮĉņęăyşăĭĉăĭjŔĉŽĎæŮžæşŤăŔ■ĉĝŕĭijNăĈşéĂŽè£ĜăŏĈèŕĈĉŤĭăşŔăyĭăŕžèşăĉŽĎăŕžă

### èĝĉăEşæŮžæąĹ

æĪJĀĉŏĀă■ŤĉŽĎæĈĒăĒĭijNăŔŕăžèăĭĤĉŤĭă getattr() ĭijŽ

```

import math

class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __repr__(self):
        return 'Point({!r:},{!r:})'.format(self.x, self.y)

```

```
def distance(self, x, y):
    return math.hypot(self.x - x, self.y - y)

p = Point(2, 3)
d = getattr(p, 'distance')(0, 0)  # Calls p.distance(0, 0)
```

āRēād'ŪāyĀçġ■æŪzæşTæYřä;£çTÍ operator.methodcaller() iijNä;NāçCīijŽ

```
import operator
operator.methodcaller('distance', 0, 0)(p)
```

ā;Şä;ăéIJĀēēAéĀŽē£ĠçŽyāRŇçŽDāRĆæTřād'ŽæñqērČçTÍæŞŘäyŁæŪzæşTæŪūiijNä;£çTÍ  
operator.methodcaller āřsā;ŁæŪzä;£äzEāĀĆ æřTāçCä;ăéIJĀēēAæŌŞāzŘäyĀçşzāLŪçŽDçCzīijNār

```
points = [
    Point(1, 2),
    Point(3, 0),
    Point(10, -3),
    Point(-5, -7),
    Point(-1, 8),
    Point(3, 2)
]
# Sort by distance from origin (0, 0)
points.sort(key=operator.methodcaller('distance', 0, 0))
```

## ěőléőž

ērČçTÍäyĀäyŁæŪzæşTāōđéŽĚäyŁæYřäyd'ėČlçNñçñNæŞ■ä;IJiijNçññäyĀæ■ēæYřæşēæL;āśđæĀġiijNç  
āZāæ■d'iijNäyžāžEērČçTÍæŞŘäyŁæŪzæşTīijNä;āāRřäzēēēŪāĒĒéĀŽē£Ġ getattr()  
æĪæşēæL;āĹrē£ŽäyŁāśđæĀġiijNçDūāRŌāE■āŌzäzēāĠ;æTřæŪzāijRērČçTÍāōČā■şāRřāĀĆ

operator.methodcaller() āĹZāžzäyĀäyŁāRřērČçTÍāržēsaiijNāzūāRŇæŪūæRŘä;ZæL'ĀæIJL'āĹ  
çDūāRŌērČçTÍçŽDæŪūāĀŽāRĪéIJĀēēAārEāōđä;NāržēsaiijäēĀŞçzŽāōČā■şāRřiijNærTāçCīijŽ

```
>>> p = Point(3, 4)
>>> d = operator.methodcaller('distance', 0, 0)
>>> d(p)
5.0
>>>
```

éĀŽē£ĠGæŪzæşTāR■çġrā■ŪçņēäyşæĪēērČçTÍæŪzæşTēĀŽāyŷāĠççŌřāIJĪéIJĀēēAæĪæNş  
case ēř■āRēæĹŪāōđçŌřēōēŪōēĀĒæĪāijRçŽDæŪūāĀŽāĀĆ  
āRĆēĀČäyNäyĀārRēĹĆēŌūāRŪæŽřād'ŽénYçžġä;Nā■ŘāĀĆ

10.21 8.21 aóđçŎřèó£éŬóèĂĚæłąaijR

éŮőécŸ

ä;äëAåd'DçRĖçTśād'gěGRäy■āRŃçśzādNçŽDāržèsaçzDāĽRĹçŽDād'■æiCæTŗæ■ōçzŚædDīijNærRäyÄ  
ærTæçCīijNēA■āŌĖäyÄäyĽæāŚā;ççzŚædDīijNçDūāRŌæāzæ■ōærRäyĽēĽĈçCżçŽDçŻyāžTĉĽūāĀAæĽgëāN.

èġčǎẸșæŮźæǻŁ

ɛʃZéGÑéAǾǾLřčŽĐeŮóéçYǎIJłçijŮćlNéçEǎşşäy■æYřǎ;ŁæŽóéA■čŽĐiijNæIJL æŮuǎǺŽǎijŽæđDǎžžǎ  
 ǎAǾǾǾ;ǎǎēçAǎEŽǎyǎǎyǎǎłǎłčđ'zǎTřǎ■ēǎłēǎǎǎijRčŽĐćlNǎžRřijNéČčǎžLǎ;ǎǎRřéČ;ēIJǎēēAǎǎǎžǎžL'ǎēČǎyN

```
class Node:
    pass

class UnaryOperator(Node):
    def __init__(self, operand):
        self.operand = operand

class BinaryOperator(Node):
    def __init__(self, left, right):
        self.left = left
        self.right = right

class Add(BinaryOperator):
    pass

class Sub(BinaryOperator):
    pass

class Mul(BinaryOperator):
    pass

class Div(BinaryOperator):
    pass

class Negate(UnaryOperator):
    pass

class Number(Node):
    def __init__(self, value):
        self.value = value
```

čDůāŘŌáL'čTíēZāžZčsžædDāzzāŋŇāēŪæTřæ■óczSædDiiŋŇāēČävŇāL'Āčd'žiiž

```
# Representation of  $1 + 2 * (3 - 4) / 5$ 
t1 = Sub(Number(3), Number(4))
t2 = Mul(Number(2), t1)
```

```
t3 = Div(t2, Number(5))
t4 = Add(Number(1), t3)
```

èŁŻæăăĀŽčŽĐěŮóécŸæŸřřžžŌæřŘäylëǎł; ĺăijŔiijŊæřŘæňæčČ; èĕĀéĜ■æŮřăőŽăžĹăŸĂéĀ■iijŊæł  
èŁŻéĜŊæĹŤăžňă; łčŤłéőŁěŮóèĀĔæłăăijŔăŔřăžžè; ĺăĹřèŁŻæăŭčŽĐčŽőčŽĐiijŽ

```
class NodeVisitor:
    def visit(self, node):
        methname = 'visit_' + type(node).__name__
        meth = getattr(self, methname, None)
        if meth is None:
            meth = self.generic_visit
        return meth(node)

    def generic_visit(self, node):
        raise RuntimeError('No {} method'.format('visit_' +
        ↪type(node).__name__))
```

ăŸžžĒă; łčŤłéŁŻăŸłčšžiijŊăŔřăžžăőŽăžĹăŸĂăŸłčšžčžăĹăăőČăžŭăŸŤăőđčŎřăŔĐčĝ■  
visit\_Name() æŮžæšŤiijŊăĔŭăŸ■NameæŸřnodečšžăđŊăĂĆ  
ăĺŊăĕČiijŊăĕČăđĪă; äăČšăśČëǎł; ĺăijŔčŽĐăĀiijijŊăŔřăžžèèŁŻæăăăĒžiijŽ

```
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        return self.visit(node.left) + self.visit(node.right)

    def visit_Sub(self, node):
        return self.visit(node.left) - self.visit(node.right)

    def visit_Mul(self, node):
        return self.visit(node.left) * self.visit(node.right)

    def visit_Div(self, node):
        return self.visit(node.left) / self.visit(node.right)

    def visit_Negate(self, node):
        return -node.operand
```

ă; łčŤłčđ'žăĺŊiijŽ

```
>>> e = Evaluator()
>>> e.visit(t4)
0.6
>>>
```

ăĪăŸžžăŸĂăŸłăŸ■ăŔŊčŽĐăĺŊă■ŔiijŊăŸŊéłăăőŽăžĹăŸĂăŸłčšžăĪăŸĂăŸłăăĹăĹéłăăřăăŸĂăŸłëǎł; ĺăă

```

class StackCode(NodeVisitor):
    def generate_code(self, node):
        self.instructions = []
        self.visit(node)
        return self.instructions

    def visit_Number(self, node):
        self.instructions.append(('PUSH', node.value))

    def binop(self, self, node, instruction):
        self.visit(node.left)
        self.visit(node.right)
        self.instructions.append((instruction,))

    def visit_Add(self, node):
        self.binop(node, 'ADD')

    def visit_Sub(self, node):
        self.binop(node, 'SUB')

    def visit_Mul(self, node):
        self.binop(node, 'MUL')

    def visit_Div(self, node):
        self.binop(node, 'DIV')

    def unaryop(self, self, node, instruction):
        self.visit(node.operand)
        self.instructions.append((instruction,))

    def visit_Negate(self, node):
        self.unaryop(node, 'NEG')

```

ä;£çŦlçd'žä;ŦrijŽ

```

>>> s = StackCode()
>>> s.generate_code(t4)
[('PUSH', 1), ('PUSH', 2), ('PUSH', 3), ('PUSH', 4), ('SUB',),
 ('MUL',), ('PUSH', 5), ('DIV',), ('ADD',)]
>>>

```

èõlèõž

āĹŽāijĀāğŦçŽDæŨūāĀŽä;āāRrèĈ;āijŽāĒŽād'gēĠRçŽDif/elseēr■āRēæĪēāōđçŌřijŦ  
 è£ŽéĠŦēōēŨōēĀĒæĪāijRçŽDāē;ād'DārsæŸréĀŽē£Ġ  
 æĪēēŌūāRŨçŽyāžŦçŽDæŨzæşŦrijŦāzūāĹ'çŦĪéĀŠā;ŠæĪēēA■āŌĒæĹ'ĀæĪĴçŽDèĹĈçĈzīijŽ
 getattr()

```

def binop(self, node, instruction):
    self.visit(node.left)

```



```

        last_result = None
        while stack:
            try:
                last = stack[-1]
                if isinstance(last, types.GeneratorType):
                    stack.append(last.send(last_result))
                    last_result = None
                elif isinstance(last, Node):
                    stack.append(self._visit(stack.pop()))
                else:
                    last_result = stack.pop()
            except StopIteration:
                stack.pop()

        return last_result

    def _visit(self, node):
        methname = 'visit_' + type(node).__name__
        meth = getattr(self, methname, None)
        if meth is None:
            meth = self.generic_visit
        return meth(node)

    def generic_visit(self, node):
        raise RuntimeError('No {} method'.format('visit_' +
→type(node).__name__))

```

æĈædIJă;ăă;ċȚȚlêfZăylċşziijŇăzşşèĈjè;ăLřçZyăRŇçŽDæȚLædIJăĂĆăžŇăóđăyLă;ăăóŇăĚĺăŔfăžěăŕĤ  
 èĂĈèŽŚăĈCăyŇăžċĉăAĭijŇéA■ăŎĖăyĂăyĭeăĭè;ăĭjRçŽDæăŚĭijŽ

```

class UnaryOperator(Node):
    def __init__(self, operand):
        self.operand = operand

class BinaryOperator(Node):
    def __init__(self, left, right):
        self.left = left
        self.right = right

class Add(BinaryOperator):
    pass

class Sub(BinaryOperator):
    pass

class Mul(BinaryOperator):
    pass

class Div(BinaryOperator):
    pass

```



```

class Negate(UnaryOperator):
    pass

class Number(Node):
    def __init__(self, value):
        self.value = value

# A sample visitor class that evaluates expressions
class Evaluator(NodeVisitor):
    def visit_Number(self, node):
        return node.value

    def visit_Add(self, node):
        return self.visit(node.left) + self.visit(node.right)

    def visit_Sub(self, node):
        return self.visit(node.left) - self.visit(node.right)

    def visit_Mul(self, node):
        return self.visit(node.left) * self.visit(node.right)

    def visit_Div(self, node):
        return self.visit(node.left) / self.visit(node.right)

    def visit_Negate(self, node):
        return -self.visit(node.operand)

if __name__ == '__main__':
    # 1 + 2*(3-4) / 5
    t1 = Sub(Number(3), Number(4))
    t2 = Mul(Number(2), t1)
    t3 = Div(t2, Number(5))
    t4 = Add(Number(1), t3)
    # Evaluate it
    e = Evaluator()
    print(e.visit(t4)) # Outputs 0.6

```

æĆædIJăŤŇăĕŮăśĆăñăąđ'ŭăŭéĆčăžŁăÿŁăĕřċŽĐEvaluatorăřăšăijŽăđ'śăĕŤĹiijŽ

```

>>> a = Number(0)
>>> for n in range(1, 100000):
...     a = Add(a, Number(n))
...
>>> e = Evaluator()
>>> e.visit(a)
Traceback (most recent call last):
...
  File "visitor.py", line 29, in _visit
return meth(node)

```



èóìèőž

ǎRɛad' Ūäy ÄäyélIÄëeAçRĖĕğçŽDǎrsæYřcTšæLRǎZlǎy■yíeldér■ǎRěǎĀĆǎ;ŠççǎLřyíeldér■ǎRěǎŪüñj  
 äyLéíçŽDǎ;Nǎ■Rǎ;ŁçTlĕfZǎyłæLǎǎIJǎelǎžcǎŽŁǎžEéĀŠǎ;SǎĀĆǎ;NǎçĆijNǎžNǎL■ǎŁsǎžnǎYřĕfZǎǎü

çÕřåĲæ■ćæĹŔyieldèr■åŘěĭjŽ

```

    ãöČäijŽärĚ node.left ěŤãŽďczŽ visit() æŰæşŦiijŇčŮăŔŎ visit()
    æŰæşŦerČčŦleĈčäyleŁĆĆčzčŽyăžŦčŽĐ visit_Name() æŰæşŦăĂĈ yield-
    æŽĈæŰăŕĚcĭŇăžŔăĖğăĽăăŽleŏłăĜczčŽerĈĈŦleĂĖiijŇăŦŖăĽğăăŇăŔŎiijŇczŖăđĬăijŹetŇăăijczŹvă

```

çIJNáoNēŁZäyÄärRēŁCiiJNä;äazšēöyæČšāŌžārzæL;āĒūāōČæšæIJL'yieldēr■āRēčŽDæŪzæāŁāĀČā;E  
ä;NāēCiiJNäyžāžEæūŁēZd'ēĀŠā;ŠiiJNä;āāŁĒēāzēēAçzt'æŁd'äyĀäyŁæāŁčzŠædđiiJNāēČædđJäy■ā;ŁçTīčTšā  
āōdēŽĒäyŁiiJNä;ŁçTīyieldēr■āRēāRřāzēēōŁ'ä;āāEŽāGžēIdäyvyäijČāžōčŽDāžččĀAiiJNāōČæūŁēZd'āžEēĀŠā;

10.23 8.23 ǎꞤꞤŌraijTꞤTĩæTĩræ■őczŞædDꞤŽDăEĖă■ŸćóacŘE

éŮőécŸ

ä;äçŽĐćÍÑăŽŔăĹŽăžžăŽĚă;ĹăďŹŽă;ĲčŎŕăiŋŢçŢĲăŢŕă■őçŽŞăďĐ(æŦŦăĈăăŠăĂăăŽ;ăĂăĚğĈăŕşèĂĚă

èġċăẸşæŮźæąŁ

```
class Node:
```

```

def __init__(self, value):
    self.value = value
    self._parent = None
    self.children = []

def __repr__(self):
    return 'Node({!r:})'.format(self.value)

# property that manages the parent as a weak-reference
@property
def parent(self):
    return None if self._parent is None else self._parent()

@parent.setter
def parent(self, node):
    self._parent = weakref.ref(node)

def add_child(self, child):
    self.children.append(child)
    child.parent = self

```

èŁŻçġ■æŸřæĈşæŰżâijRăĚĀèőÿparentéİŻézŸçzŁæ■cǎĂĆăĬŃăęĆiijŻ

```

>>> root = Node('parent')
>>> c1 = Node('child')
>>> root.add_child(c1)
>>> print(c1.parent)
Node('parent')
>>> del root
>>> print(c1.parent)
None
>>>

```

ëöłëőż

ăĬĭçŎřâijTçŦĭçŻĐæŦřæ■őçzŞæđĐăĬĬPythonăÿ■æŸřăÿĂăÿĤăĬăçŸæĹŃçŻĐéŰőécŸiijŃăŻăăÿžæ■čăÿăĬŃăęĆèĂĈèŻŚăęĆăÿŃăžčçăĀiijŻ

```

# Class just to illustrate when deletion occurs
class Data:
    def __del__(self):
        print('Data.__del__')

# Node class involving a cycle
class Node:
    def __init__(self):
        self.data = Data()
        self.parent = None
        self.children = []

```

```
def add_child(self, child):
    self.children.append(child)
    child.parent = self
```

äyNéÍcæŁŚäzhnä;£çŦlê£ŽäyläzčçäAæIěäAŽäyÄäzŽädČäIJčäZdæŦüerŦéŦñijŽ

```
>>> a = Data()
>>> del a # Immediately deleted
Data.__del__
>>> a = Node()
>>> del a # Immediately deleted
Data.__del__
>>> a = Node()
>>> a.add_child(Node())
>>> del a # Not deleted (no message)
>>>
```

āRrāzēçIJNāLrījNæIJĀāRŌāyĀāyīçŽDāLāēŽd' æUūæL' Šā■rēr■āRēæšqæIJL' āGžçŌrāĀCāŌšāZāæYrPy  
ā;ŠāyĀāyīāržēšqçŽDāijTçTīæTŗāRŸæLRŌçŽDæUūāĀZæL■āijŽçnNā■šāLāēŽd' æŌL' āĀCēĀNāržāžŌā;īçŌr  
āZāæ■d' iijNāIJlāyLēlčā;Nā■Rāy■æIJĀāRŌēČlāLērījNçLūēŁCçČzāŠNā■l' ā■RēŁCçČzāžŠçŽyæNēæIJL' ārza

PythonæIJL'âRëad'ŮčŽDădČâIJ;ăZđæŤŭăZlăIěăyŞéŮlėŚLărză;łçŎrăijŤçŦlčŽDiiJŃă;EăŸră;ăærÿèIJ  
âRëad'Ůă;ăèŸŸăRrăzěăL'ŃăLłčŽDěğăRŚăŏČiiJŃă;EăŸrăzčçăAçIJŃăyŁăŐză;ŁăŃńiiJŽ

```
>>> import gc
>>> gc.collect() # Force collection
Data.__del__
Data.__del__
>>>
```

æĈædIJaꞤłĉŌraijȚĉȚlĉŽĐaržesæĜłauśeŸăoŽăZL'ăžEĝĜłauśĉŽĐ  
 \_\_\_\_del\_\_\_\_() æŰzæsȚiijNéCčăZŁaijŽeol' æĈĚăEȚaRŸăꞤŰăZt'ĉșșșȚăĂĈ  
 ăAĜeol'ăjăăĈRăyNéIcêŸZăăuĉzŽNodeăoŽăZL'èĜłauśĉŽĐ \_\_\_\_del\_\_\_\_() æŰzæsȚiijŽ

```
# Node class involving a cycle
class Node:
    def __init__(self):
        self.data = Data()
        self.parent = None
        self.children = []

    def add_child(self, child):
        self.children.append(child)
        child.parent = self

# NEVER DEFINE LIKE THIS.
# Only here to illustrate pathological behavior
def __del__(self):
    del self.data
    del self.parent
```

```
del.children
```

ēfŽçg■æČĚāĖtāyNriiŃāđČāIJĭāŽđæTūæryēfIJēČĭāy■āijŽāŌzāŽđæTūēfŽāyĭāržèšaçŽĎriiŃēfYāijŽārij  
āēČādIJāĭāērTçĭĀāŌžēfRēāŃāōČāijŽāRŚçŌriiŃData.\_\_del\_\_  
æŭLæAřæryēfIJāy■āijŽāGžçŌřāžE,çTŽēGşāIJĭāĭāijžāLŭāĖĚā■YāŽđæTūæŪūriiŽ

```
>>> a = Node()
>>> a.add_child(Node())
>>> del a # No message (not collected)
>>> import gc
>>> gc.collect() # No message (not collected)
>>>
```

āijsāijTçTĭlæŭLéŽd'āžĖāijTçTĭlāĭçŌřçŽĎēfŽāyĭēŪōécYriiŃæIJnèt'ĭæĭēèōšriiŃāijsāijTçTĭlāřsæYřāyĀāy  
āĭāāRřāžēēĀŽēfĜ weakref æĭēāLŽāžžāijsāijTçTĭlāĀČāĭŃāēČriiŽ

```
>>> import weakref
>>> a = Node()
>>> a_ref = weakref.ref(a)
>>> a_ref
<weakref at 0x100581f70; to 'Node' at 0x1005c5410>
>>>
```

āyžāžĖēōēfēŪōāijsāijTçTĭlæL'ĀāijTçTĭlçŽĎāržēsārijŃāĭāāRřāžēāČRāĜĭæTřāyĀæāŭāŌžērČçTĭlāōČā■şāR  
çTĭsāžŌāŌşāĝŃāržēsaçŽĎāijTçTĭlēōæTřæşææIJL'āćđāLārijŃēČčāžLārřsāRřāžēāŌzāLāēŽd'āōČāžĖāĀČāĭŃāē

```
>>> print(a_ref())
<__main__.Node object at 0x1005c5410>
>>> del a
Data.__del__
>>> print(a_ref())
None
>>>
```

ēĀŽēfĜēfŽēĜŃāijTçd'žçŽĎāijsāijTçTĭlæLĀæIJriiŃāĭāāijŽāRŚçŌřāy■āĖ■æIJL'āĭçŌřāijTçTĭlēŪōécY  
āĭāēfYēČĭāRČēĀČ8.25ārRēLČāĖşāžŌāijsāijTçTĭlçŽĎāRēād'ŪāyĀāyĭāĭŃā■RāĀČ

## 10.24 8.24 èóĭ'çşzæTřæŃAæřTèĭČæŞ■āĭĭ

### éŪōécY

āĭāæČşēōĭ'æşŘāyĭçşzçŽĎāōđāĭŃæTřæŃAæāĜāĜĖççŽĎærTèĭČēfRçşŏŪ(ærTāēČ>=,!=,<=,<ç■L')iijŃāĭĖ

### èĝčĀĖşæŪzæāĭ

PythonçşzāržærŘāyĭærTèĭČæŞ■āĭIJēČĭēIJāēēĀāōđçŌřāyĀāyĭçL'žæōLæŪzæşTæĭēæTřæŃAāĀČ  
āĭŃāēČāyžāžĖæTřæŃA>=æŞ■āĭIJçņēiijŃāĭāēIJāēēĀāōŽāžL'āyĀāyĭ\_\_\_\_ge\_\_\_\_()  
æŪzæşTāĀČārĭçōāāōŽāžL'āyĀāyĭæŪzæşTæşāžāžĀžLēŪōécYriiŃāĭĖāēČādIJēēĀĭāāōđçŌřæL'ĀæIJL'ārřē

äIäyZäçNå■RrijNæLŠäznædDäzzäyÄäZæLŁå■RrijNçDúãRÕçzZăőČäznăcđăŁăäyÄäZæLŁéŮrrijNæ

èƧŽéGŇæĹŚäzñáRlæŸřczŽHouseçşzäoŽăZl'ăžEäyď'äylæŰzæşŦiijŽ\_\_eq\_\_()    äšŇ  
 \_\_1t\_\_() iijŇăoČăřsèČjæŦræŇAæĹ'ĂæIJL'čŽDærŦèĭČăŞ■ăjIiijŽ

```
# Build a few houses, and add rooms to them
h1 = House('h1', 'Cape')
h1.add_room(Room('Master Bedroom', 14, 21))
h1.add_room(Room('Living Room', 18, 20))
h1.add_room(Room('Kitchen', 12, 16))
```

```

h1.add_room(Room('Office', 12, 12))
h2 = House('h2', 'Ranch')
h2.add_room(Room('Master Bedroom', 14, 21))
h2.add_room(Room('Living Room', 18, 20))
h2.add_room(Room('Kitchen', 12, 16))
h3 = House('h3', 'Split')
h3.add_room(Room('Master Bedroom', 14, 21))
h3.add_room(Room('Living Room', 18, 20))
h3.add_room(Room('Office', 12, 16))
h3.add_room(Room('Kitchen', 15, 17))
houses = [h1, h2, h3]
print('Is h1 bigger than h2?', h1 > h2) # prints True
print('Is h2 smaller than h3?', h2 < h3) # prints True
print('Is h2 greater than or equal to h1?', h2 >= h1) # Prints False
print('Which one is biggest?', max(houses)) # Prints 'h3: 1101-
    ↳square-foot Split'
print('Which is smallest?', min(houses)) # Prints 'h2: 846-square-
    ↳foot Ranch'

```

## èóìèõž

āĖŭāōđ                      total\_ordering                      ěčĚěērāŽlāžšæšæċĈāžĹčēđċġŸāĀĆ  
 āōČāršæŸřāōŽāžĹāžĖāŷĀāŷlāžŌæřRāŷlæřTèĹČæŤræŇAæŮžæšŤāĹræĹĀæIJĹéIJĀēĖAāōŽāžĹčŽDāĖŭāžŮ  
 æřŤāēĈāĵāāōŽāžĹāžĖ \_\_\_\_le\_\_\_\_() æŮžæšŤĭijŇēĈčāžĹāōČāršēċŋĹlāĭēæđDāžžæĹĀæIJĹāĖŭāžŮčŽDēIJĀē  
 āōđēŽĖāŷĹāršæŸřāIJĹšžēĠŇēĹċāČRāŷŇēĹċēŹæāŭāōŽāžĹāžĖāŷĀāžŽĹžæōĹæŮžæšŤĭijŽ

```

class House:
    def __eq__(self, other):
        pass
    def __lt__(self, other):
        pass
    # Methods created by @total_ordering
    __le__ = lambda self, other: self < other or self == other
    __gt__ = lambda self, other: not (self < other or self == other)
    __ge__ = lambda self, other: not (self < other)
    __ne__ = lambda self, other: not self == other

```

āĴšĈDŭĭijŇāĵæĠāŭsāŌžāĖŽāžšāĹLāōžæŸšĭijŇāĵĖæŸřāĵĸĹŤĭ      @total\_ordering  
 āřřāžēċōĀāŇŮāžčċāAĭijŇāĵāžRēĀŇāŷāŷžāŠċāĀĆ

## 10.25 8.25 āĹŽāžžċijŠā■ŸāōđāĹŇ

### éŮōécŸ

āIJĹāĹŽāžžāŷĀāŷĹċšžĈDāržēsæĖŮŭĭijŇāēĈæđIJāžŇāĹ■āĵĸĹŤĭāŖŇæāŭāŖĆæŤrāĹŽāžžēŹĠēŹāŷlāržēs  
 āĵāæĈšēŹŤāžđāōČĈŽDċijŠā■ŸāĵŤĈŤĭāĀĆ



èġčǎẸ₃æŮ́æąŁ

æfZçg■éÅžăyÿæYřăZăăyžă;ăăyŇnæIJZçZÿăRŇŇăRĈæTřăĹZăžzçŽDăřzèšəæUŭă■TăĹŇçŽDăĂĈ  
 ăIJăĹZăĹăDăŽăžSăy■Ĉ;æIJĹăôđéZĚçŽDă;Ňă■RĥijŇăřTăĕĈ logging  
 æĹăăĹUĥijŇă;ĤçTĭçZÿăRŇçŽDăR■çğřăĹZăžzçŽD loggerăôđăĹNăřÿèĤIJăRĹăIJĹăÿĂăyĹăĂĈă;ŇăĕĈĥijŽ

```
>>> import logging
>>> a = logging.getLogger('foo')
>>> b = logging.getLogger('bar')
>>> a is b
False
>>> c = logging.getLogger('foo')
>>> a is c
True
>>>
```

äyžāẸē;ǻłŖēƒZæāũçŽĐæȚŁæđIııjÑä;ǻéIǺēēAǻ;ǻçȚłäyĂäyłǻSŃçszæIıñěznǻŁEǻıjĂçŽĐăũăŎĆăĜ

```
# The class in question
class Spam:
    def __init__(self, name):
        self.name = name

# Caching support
import weakref
_spam_cache = weakref.WeakValueDictionary()
def get_spam(name):
    if name not in _spam_cache:
        s = Spam(name)
        _spam_cache[name] = s
    else:
        s = _spam_cache[name]
    return s
```

çDũãRŎãAŽäyÄäyłætNërTijjNä;ääijZãRŠçŎřeușăzNãL■éCčäyłæUëafUărzesaçŽDãŁZăzžeaŃăyžæYr.

```
>>> a = get_spam('foo')
>>> b = get_spam('bar')
>>> a is b
False
>>> c = get_spam('foo')
>>> a is c
True
>>>
```

èóìèőž

çijŮâEŽäyÄäyĭâuêâŎCâĜĭæTṛæİêäƒôæTṛæŽôéÄŽçŽDâôđäĭNâĬŽâžžèaŇNäyžéÄŽäyŷæYřäyÄäyĭærTēĭ  
äĭEæYřræĬSäžñèſYēČĭâRēæLĭĭĭLřæŽtĭaijYéŽĚčŽDëğcâEşæŮžæaĬâŚciijş

äĲNäęĆiijNä;ääRřëĈ;äijŽëÄĈëŽŚëĜ■æŮřăōŽăzL'çşzçŽĎ  
æŮžæşŤiijNăřsăĈRăyNéİcëŁZæăüiijŽ

\_\_new\_\_()

```
# Note: This code doesn't quite work
import weakref

class Spam:
    _spam_cache = weakref.WeakValueDictionary()
    def __new__(cls, name):
        if name in cls._spam_cache:
            return cls._spam_cache[name]
        else:
            self = super().__new__(cls)
            cls._spam_cache[name] = self
            return self
    def __init__(self, name):
        print('Initializing Spam')
        self.name = name
```

ăĲİçIJNèŭăİēăĲăĈRăRřăzëëĲ;ăĲřëĎDæIJşæŤĲæđIJiijNă;EæŸřéŮőëċŸæŸř  
\_\_init\_\_() æřRăňăęĈ;äijŽëċnërĈĈŤİiijNăy■çőăęŁZăyĲăōđăĲNăŸřăRřëċnċijŞă■ŸăžEăĂĈăĲNăęĆiijŽ

```
>>> s = Spam('Dave')
Initializing Spam
>>> t = Spam('Dave')
Initializing Spam
>>> s is t
True
>>>
```

ëŁZăyĲăĲŮőöyăy■æŸřă;ăæĈşëęAçŽĎæŤĲæđIJiijNăŽăæ■đ'ëŁŽçĝ■æŮžæşŤăžúăy■ăRřăŮŮăĂĈ

ăyĲéİcăĲŚăžnă;ŁçŤĲăĲřăžEăiįsăiįŤçŤĲëőăęŤřiijNăřzăžŎăđĈăIJĲăŽđæŤŭăİëëőşæŸřăĲăĲIJĲăyőăĲĲ'çŽ  
ă;ŞăĲŚăžnăĲİăNăĂăōđăĲNċijŞă■ŸăŮŭiijNă;ăăRřëĈ;ăRĲăĈşăIJĲĲNăžRăy■ă;ŁçŤĲăĲřăőĈăžnăŮŭăĲ■ăĲİă■  
ăyĂăyĲWeakValueDictionaryăōđăĲNăRĲăiįŽăĲİă■ŸëĈĉăžŽăIJĲăĲŭăőĈăIJřăŮžëŁŸăIJĲëċnă;ŁçŤĲçŽĎă  
ăRęăĲŽçŽĎëřiijNăRĲëęĂăōđăĲNăy■ăE■ëċnă;ŁçŤĲăžEiijNăőĈăřsăžŎă■ŮăĲyăy■ëċnċĝzëŽđ'ăžEăĂĈëĝĈăřşă

```
>>> a = get_spam('foo')
>>> b = get_spam('bar')
>>> c = get_spam('foo')
>>> list(_spam_cache)
['foo', 'bar']
>>> del a
>>> del c
>>> list(_spam_cache)
['bar']
>>> del b
>>> list(_spam_cache)
[]
>>>
```

ăřzăžŎăđ'ĝëĈĲăĲĲĲNăžRëĂNăŭşiijNëŁŽëĜNăžĉĉăĂăŭşçzĲăđ'şçŤĲăžEăĂĈăy■ëŁĜëŁŸăŸřăĲIJĲăyĂă

éĕŨăĚĹæŸřēŁŻéĜŃăĵŁĉŦĲăĹřăžĒăŸĀăŸĲăĚĲăŝĂăŔŸéĜŔĲĲŃăžŭăŸŦăŭēăŐĈăĜĲæŦřēŭŝĉŝăŦĲăĲĲăŸĂă

```
import weakref

class CachedSpamManager:
    def __init__(self):
        self._cache = weakref.WeakValueDictionary()

    def get_spam(self, name):
        if name not in self._cache:
            s = Spam(name)
            self._cache[name] = s
        else:
            s = self._cache[name]
        return s

    def clear(self):
        self._cache.clear()

class Spam:
    manager = CachedSpamManager()
    def __init__(self, name):
        self.name = name

    def get_spam(name):
        return Spam.manager.get_spam(name)
```

ēŁŻăăŭĉŽĎēŲăžĉĉăĂăŽĲ'ăŸĒăŽŕĲĲŃăžŭăŸŦăžŝăŽĲ'ĉĂĲăŲ'žĲĲŃăĹŝăžŃăŔřăžēăĉĎăĹăăŽĲ'ăĎ'ŽĉŽĎĉĲĲ

ēŁŸăĲĲăŸĂĉĈăŕŝăŸŕĲĲŃăĹŝăžŃăŽĲ'ēĲŝăžĒĉŝĉŽĎăăĉĎăĲŃăŨŰĉžŽĉŦĲăĹŭĲĲŃĉŦĲăĹŭăĲĲăăŝăŸŝŰ

```
>>> a = Spam('foo')
>>> b = Spam('foo')
>>> a is b
False
>>>
```

ăĲĲăĜăĉĝăŰăžăĲŔăŔřăžēēŸŝăĲĉŦĲăĹŭēŁŻăăŭăĂăŽĲĲŃĉŃăŸĀăŸĲăŸŕăŔĒĉŝĉŽĎăŔăăŰăăŝăăŦăžăŸăĉŃăžŃĉĝăŕŝăŸŕēŲ'ēŁŻăŸĲĉŝĉŽĎ \_\_init\_\_() æŰăŝŦăĹŽăĜăžăŸĀăŸĲăĲĲăŸŕĲĲŃēŲ'ăăŐăŸăĲĉĉăŃăĲ

```
class Spam:
    def __init__(self, *args, **kwargs):
        raise RuntimeError("Can't instantiate directly")

    # Alternate constructor
    @classmethod
    def __new(cls, name):
        self = cls.__new__(cls)
        self.name = name
```

ĉĎŨăŔŐăăŝăăŦăžĉĲĲŝăŸĉăăĉŔĒăŽăžĉĉăĂăĲĲŃăĲĉŦĲSpam.\_\_new()  
ăĲăăĹŽăžăăăăăĲĲŃĲŃăăŸăăŸŕĉŽĲ'ăăŐăăŔĉŦĲĲSpam() æĎĎăăăăăĜĲæŦřĲĲŽ

```
# -----æIJĀăŔŎçŽĎăĤŏă■čæŮzæąĹ-----
↪-----
class CachedSpamManager2:
    def __init__(self):
        self._cache = weakref.WeakValueDictionary()

    def get_spam(self, name):
        if name not in self._cache:
            temp = Spam3._new(name) # Modified creation
            self._cache[name] = temp
        else:
            temp = self._cache[name]
        return temp

    def clear(self):
        self._cache.clear()

class Spam3:
    def __init__(self, *args, **kwargs):
        raise RuntimeError("Can't instantiate directly")

    # Alternate constructor
    @classmethod
    def _new(cls, name):
        self = cls.__new__(cls)
        self.name = name
        return self
```

æIJĀăŔŎçŽæăŮçŽĎăŮzæąĹăŕśăŭşçzŔeŭşăđ'şăë;ăžEăĂĆ  
çijŞă■ŸăŠŇăĚŭăžŮăđĐéĂăăİăijŔëĤŸăŔŕăžëă;ĤçŤĬ9.13ăŕŔëĹĆăy■çŽĎăĤŐçşzăŏđçŎŕçŽĎăŽŤ'ăijŸéŽĚăy

# 11 çňňăžĹçňăĭijŽăĚĈçijŮćĬŃ

ëĭŕăžŭăĭjĂăŔŚéçEăşşăy■æIJĂçzŔăĚŸçŽĎăŔčăđ't'çëĚăŕśăŸŕăĂIJdonăĂŽt repeat your-  
selfăĂĬăĂĆ äžşăŕśăŸŕëŕt'ĭijŇăžză;ŤăŮŭăĂŽă;Şă;ăçŽĎćĬŇăžŔăy■ă■ŸăĬĬénŸăžëéĜăăđ'■(æĹŮëĂĚăŸŕéĂ.  
ăĬĬPythonă;Şăy■ĭijŇăĂžăyŸéÇ;ăŔŕăžëéĂŽëĤĜăĚĈçijŮćĬŃăĬëëĝčăĤşëĤŽçşzéŮŏéçŸăĂĆ  
çŏĂëĂŇĬăĂžŇĭijŇăĚĈçijŮćĬŃăŕśăŸŕăĤşăžŎăĹŽăžzæŞăă;IJăžŔăžçčăĂ(æŕŤăçĆăĤŏăŤzăĂĂçŤşăĹŔăĹŮ  
ăyžëçĂăĹĂăIJŕăŸŕă;ĤçŤĬëçĚëçŕăŽĬăĂĂçşzëçĚëçŕăŽĬăŠŇăĚĈçşzăĂĆăy■ëĤĜëĤŸăĬĬŸăĂžŽăĚŮăžŮăĹĂ  
ăŇĚăŇŇç■;ăŔ■ăŕžëşăăĂĂă;ĤçŤĬexec() æĹĝëăŇăžçčăĂăžëăŔĹăŕžăĤĚëĈĬăĜ;æŤŕăŠŇçşzçŽĎăŔ■ăŕĎăĹ  
æIJŇçŇăçŽĎăyžëçĂçŽŏçŽĎăŸŕăŔŚăđ'ĝăŏŭăžŇçz■ëĤŽăžŽăĚĈçijŮćĬŃăĹĂăIJŕĭijŇăžŭăyŤçzŽăĜăŏđă;Ňă

Contents:



```
@timethis
def countdown(n):
    pass
```

èùšâĈŔäyŊéİçèĤZæăûâĖŽâĖŮăđæŤĽæđIJæŸŕäyĂæăûçŽĎiiž

```
def countdown(n):
    pass
countdown = timethis(countdown)
```

éąžäĭĤèŕt'äyĂäyŊriiŊŊăĖĚçĭŏçŽĎèċĖĖĕŕăŽĬăŕŤăĖĈ @staticmethod,  
 @classmethod, @property ăŎšçŔĖăžšæŸŕäyĂæăûçŽĎăĂĈ  
 äĭŊăċĈiiŊŊäyŊéİçèĤZăyđ'ăylăžçċăAçĽĜăđŭæŸŕçĭĽăžûçŽĎiiž

```
class A:
    @classmethod
    def method(cls):
        pass

class B:
    # Equivalent definition of a class method
    def method(cls):
        pass
    method = classmethod(method)
```

ăĬĬăyĽéİççŽĎ wrapper() âĜĭæŤŕäyĭiiŊŊ ĕċĖĖĕŕăŽĬăĖĖĕĈĬăđZăzĽăžĖäyĂäyĭăĭĤçĬĬ  
 \*args âŖŊ \*\*kwargs æĬæŎċăŔŮăžzæĎŔăŔĈæŤŕçŽĎăĜĭæŤŕăĂĈ  
 âĬĬĕĤZăyĭăĜĭæŤŕĕĜŊéİçĕŕĈçĬĬăžĖăŎšăĝŊăĜĭæŤŕăžŮăŕĖăĖŮçzšæđIJĕĤăŽđiiŊŊäyĖĕĤĜăĭăĕĤŸăŔŕăžæăûç  
 çĎăăŔŎĕĤZăyĭăŮŕçŽĎăĜĭæŤŕăŊĖĕċĖăŽĬĕċăĭIJăyžçzšæđIJĕĤăŽđæĬăžçæZăăŎšăĝŊăĜĭæŤŕăĂĈ

éĬĂĕĖĂăijžĕŕĈçŽĎæŸŕĕċĖĖĕŕăŽĬăžŮăyĖăijŽăĤŏæŤzăŎšăĝŊăĜĭæŤŕçŽĎăŔĈæŤŕçĭăŔăăžăăŔĽĕĤăŽă  
 äĭĤçĬĬ \*args âŖŊ \*\*kwargs çŽŏçŽĎăŕŕŕæŸŕçăđăĤĬăžzăĭŤăŔĈæŤŕĕĈĭĕĈĭéĂĈçĬĬăĂĈ  
 ĕĂŊĕĤăŽđçzšæđIJăĬijăšžæĬJŊĕĈĭæŸŕĕŕĈçĬĬăŎšăĝŊăĜĭæŤŕ func(\*args,  
 \*\*kwargs) çŽĎĕĤăŽđçzšæđIJiiŊŊăĖŮăyĖfuncăŕŕŕæŸŕăŎšăĝŊăĜĭæŤŕăĂĈ

ăĬŽăijĂăĝŊăăăžăĕċĖĖĕŕăŽĬçŽĎăŮăăĂŽiiŊŊăijŽăĭĤçĬĬăyĂăžZçŏĂăĭŤçŽĎăĭŊăŔăĬĕŕt'æŸŎiiŊŊăŕ  
 äyĖĕĤĜăđđĖŽĖăĬJžæŽŕăĭĤĬăŮŮiiŊŊĕĤŸæŸŕăĬJĬăyĂăžZççĖĖĽĈĖŮĕċŸĖĖĂăŕĭăĎŔçŽĎăĂĈ  
 æŕŤăĖĈăyĽéİçăĭĤçĬĬ @wraps(func) æŖĬĕĝçæŸŕăĭĽĖĜĖĖĖĂçŽĎiiŊŊ  
 âŏĈĕĈĭăĤĬçŤăŎšăĝŊăĜĭæŤŕçŽĎăĖĈæŤŕăĖŏ(ăyŊăyĂăŕŔĕĽĈăijŽĕŏŖăĬŕ)iiŊŊăŮŕăĽŊçzŔăyŮăijŽăĤĭçĤĕĖ  
 æŎĖăyŊăĬĕçŽĎăĜăyĭăŕŔĕĽĈăĽŖăžăijŽăŽŕăĽăăŮăăĖĖçŽĎĕŏŖĕĝĕċĖĖĕŕăŽĬăĜĭæŤŕçŽĎçzĖĖĽĈĖŮĕċŸ

## 11.2 9.2 âĬZăžžĕċĖĖĕŕăŽĬăŮăăĤĭçŤZăĜĭæŤŕăĖĈăĖăĖĂŕ

### ĕŮĕĖŸ

ăĭăăĖŽăžĖäyĂäyĭĕċĖĖĕŕăŽĬăĭJçĬĬăĬJăŖŔăyĭăĜĭæŤŕăyĬiiŊŊăĖĖŕĕĤZăyĭăĜĭæŤŕçŽĎĖĜĖĖĂçŽĎăĂĈ

## èġċàEşæŮzæąŁ

äzzä;TæŮüăĂZă;ăăőŻăzL'èċĚëĕřăŹÍċŽĎăŮüăĂZiijŃëĈ;ăžTĕřăă;ĤċŤÍ functools  
ăžŞăy■ĈŽĎ @wraps èċĚëĕřăŹÍăĬëăşĬèġċăžTăśĈăŃĚèċĚăĜ;æTŕăĂĈă;ŃăęĈiijŽ

```
import time
from functools import wraps
def timethis(func):
    '''
    Decorator that reports the execution time.
    '''
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        result = func(*args, **kwargs)
        end = time.time()
        print(func.__name__, end-start)
        return result
    return wrapper
```

äyŃéĬăĹŚăžňă;ĤċŤĬëĤZăyĬèċăŃăŃĚèċĚăŔŎċŽĎăĜ;æTŕăžüăċĂăşëăőĈċŽĎăĚĈăĤăæAŕiijŽ

```
>>> @timethis
... def countdown(n):
...     '''
...     Counts down
...     '''
...     while n > 0:
...         n -= 1
...
>>> countdown(100000)
countdown 0.008917808532714844
>>> countdown.__name__
'countdown'
>>> countdown.__doc__
'\n\tCounts down\n\t'
>>> countdown.__annotations__
{'n': <class 'int'>}
>>>
```

## èőĬèőž

ăĬĬċijŮăĚZèċĚëĕřăŹÍċŽĎăŮüăĂZăđ'■ăĹüăĚĈăĤăæAŕăYŕăyĂăyĬéĬăyŷyēĜ■ëċAċŽĎëĈăĬăĤăăĂĈăęĈă  
@wraps iijŃëĈăžĹă;ăăiijŽăŔŚċŎŕëċăċĬèċĚëĕřăĜ;æTŕăyċăđ'şăžĒăĤĂăĬĹăĬĹċŤĬċŽĎăĤăæAŕăĂĈăŕŤăęĈă  
@wraps âŔŎċŽĎăŤĬăđĬăYŕăyŃéĬċëĤZăăŷċŽĎiijŽ

```
>>> countdown.__name__
'wrapper'
>>> countdown.__doc__
```

@wraps	æIJL'äyÄäyléG■èeAçL'zâ;AæYřáoČčČ;ěol'ä;äéÄŽefĜăśđæĂğ
__wrapped__	čŽt'æŎèèöfÉŮòèćnăŇĚècĚăĜ;æTřăĂĆă;ŇăćĆ:

\_\_wrapped\_\_ ąsđæĀğēfYëĈ;ěol'ěcñěĎĚěřāĠ;æTřæ■čcaőæŽt'élJšāžTāsĆčŽDāRĆæTřč■;āŘ■āfæA

äÿÄäÿl̥; ŁæŽóéA■čŽDēUóécŸæŸræĀŌæuèol'èčĒēēřāZlāŌzčŽt'æŌēad'■āLūāŌšāgNāG;æTřčŽDāRČ  
æçCædIJæCšèGłāūsæL'NāLlāōđčŌřčŽDērīēIJāēçAāAž'ad'gēGRčŽDāuēā;IJtjJNāIJĀāē;ārščōĀā■TčŽDā;čçT  
@wraps                      èčĒēēřāZlāĀC                      éĀžēŁgāžTāšćČŽD                      \_\_\_\_\_wrapped\_\_\_\_  
āšdæĀgèōŁēUōāLrāG;æTřč■;ār■āŁæAřāĀCæŽt'ad'ŽāĒšāžŌč■;ār■čŽDāEĒāōzāRřāzēāRČèĀČ9.16ārRēŁ

äyÄäyłęćĖĖēřāZīāuščzŔä;IļĶTlāIļlāyÄäyłāĢ;æTřäyLii;Nā;ăæĈşæŠd'ėTĀāōĈii;ŅĉZt'æŌēēōfėŪōāŌşā;

wrapped    åAĞëø;èċĒēřāZlæYřéÅŽèŁǦ @wraps (åŖĆèĂĈ9.2årŘèŁĆ)æİěăôđçÕřčŽDñijÑėĆczłŁä;ääŔřázéeĂŽ  
                     āsđæĀğæİĉēōŁēUőăÔşăğŃăĜ;æTñijŻ

çZt æŌëëðféŬöæIJlãNĖëçĖĖçZĎãŌşğNãĠ;æTřlIJlërČèrTãĀAãĖĖçIJAãŠNãĖŮäzŮãĠ;æTřæŞ■ä;IJæŮ  
ä;EæŸræĹSäzñèfēZēĠNĖçZĎæŮzæĹLäzĖäzĖĖĀČçTlāzŌãIJlãNĖëçĖĖäZlāy■æ■ççaöä;£çTlāzE



æĆæđIæIJL'ad'ŽäyĹăŃĕĕĚăZĹiijŃĕĆčăžĹĕoĕĹĕŮo \_\_\_\_\_wrapped\_\_\_\_  
 ăśđæĂğĉZĎĕăNăyžæŸřă■ăRĕćDĉşĕĉZĎiijŃăžTĕrĕĕĂĹăĖĕĹZăăăăĂZăĂĆ

```
from functools import wraps

def decorator1(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Decorator 1')
        return func(*args, **kwargs)
    return wrapper

def decorator2(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Decorator 2')
        return func(*args, **kwargs)
    return wrapper

@decorator1
@decorator2
def add(x, y):
    return x + y
```

```
>>> add(2, 3)
Decorator 1
Decorator 2
5
>>> add.__wrapped__(2, 3)
5
>>>
```

```
>>> add(2, 3)
Decorator 1
Decorator 2
5
>>> add.__wrapped__(2, 3)
Decorator 2
5
>>>
```

æIĬăŔŌèèAèrt'çZDæYřiiĴŇázüä■æYřæL'ĂæIJL'çZDèèĚèřăZÍléČ;ä;ŁçTíläžĚ  
@wraps iiĴŇăZăă■d'èfZéGŇçZDæŮzæŁŁăzüä■ăĚléCléĂČçTíăĂČ  
cŁ'zăŁŇcZDiiĴŇăĚĚç;őçZDèèĚèřăZÍ @staticmethod aŠŇ @classmethod

äršæšæIJL'éAṭ;łèfŽäyłçžæǎŽ (ǎǎCǎžñæLLǎŌšǎgNǎĜ;æTřǎ■ŸǎCíǎIJǎśđæǎĜ \_\_func\_\_  
äy■)ǎǎĆ

## 11.4 9.4 ǎǎŽǎžL'äyǎÄyłǎyęǎRĆæTřçŽǎĐčĚéěřǎŽí

éŮóécŸ

ä;ǎæČšǎǎŽǎžL'äyǎÄyłǎRřǎžæŎěǎRŮǎRĆæTřçŽǎĐčĚéěřǎŽí

èĝcǎEşǎŮzǎǎL

ǎĹSǎžñčTłäyǎÄyłǎ;Nǎ■RèřçžEéŸRèřǎyNǎŎěǎRŮǎRĆæTřçŽǎĐč'ĐçRĚčŁĜćíNǎǎĆ  
ǎAĜèǎ;ä;ǎæČšǎEŽäyǎÄyłèčĚéěřǎŽíiijNçžŽǎĜ;æTřæüzǎŁǎæŮěǎŁŮǎŁšèČ;iiijNǎRŊǎŮǎǎĚAèóyçTłǎŁǎǎN  
äyNéíCǎŸřèfŽäyłèčĚéěřǎŽíçŽǎǎŽǎžL'ǎŠNǎ;ŁçTłčđ'žǎ;NíijŽ

```
from functools import wraps
import logging

def logged(level, name=None, message=None):
    """
    Add logging to a function. level is the logging
    level, name is the logger name, and message is the
    log message. If name and message aren't specified,
    they default to the function's module and name.
    """
    def decorate(func):
        logname = name if name else func.__module__
        log = logging.getLogger(logname)
        logmsg = message if message else func.__name__

        @wraps(func)
        def wrapper(*args, **kwargs):
            log.log(level, logmsg)
            return func(*args, **kwargs)
        return wrapper
    return decorate

# Example use
@logged(logging.DEBUG)
def add(x, y):
    return x + y

@logged(logging.CRITICAL, 'example')
def spam():
    print('Spam!')
```

ǎĹčIJNèṭǎǎǎēiijNèfŽçg■ǎǎđçŎřçIJNǎyŁǎŎžǎ;Łǎđ'■ǎíCíijNǎ;EǎŸřæǎyǎŁČǎǎǎČšǎ;Łçǎǎǎ■TǎǎĆ  
ǎIJǎǎđ'ŮǎśČçŽǎĐǎĜ;æTř logged() æŎěǎRŮǎRĆæTřǎžǎǎřEǎǎČǎžñǎ;IJčTłǎIJǎǎĚéČłçŽǎĐčĚéěřǎŽíǎĜ;æ



```

'''
def decorate(func):
    logname = name if name else func.__module__
    log = logging.getLogger(logname)
    logmsg = message if message else func.__name__

    @wraps(func)
    def wrapper(*args, **kwargs):
        log.log(level, logmsg)
        return func(*args, **kwargs)

    # Attach setter functions
    @attach_wrapper(wrapper)
    def set_level(newlevel):
        nonlocal level
        level = newlevel

    @attach_wrapper(wrapper)
    def set_message(newmsg):
        nonlocal logmsg
        logmsg = newmsg

    return wrapper

return decorate

# Example use
@logged(logging.DEBUG)
def add(x, y):
    return x + y

@logged(logging.CRITICAL, 'example')
def spam():
    print('Spam!')

```

äyÑéÍcæYřäzd'azŠçŔřâcČäyŇçŽDä;řçTlä;Nâ■ŘijŽ

```

>>> import logging
>>> logging.basicConfig(level=logging.DEBUG)
>>> add(2, 3)
DEBUG:__main__:add
5
>>> # Change the log message
>>> add.set_message('Add called')
>>> add(2, 3)
DEBUG:__main__:Add called
5
>>> # Change the log level
>>> add.set_level(logging.WARNING)
>>> add(2, 3)

```



äyÄäyġæŕTēĭČĚŽĭçŘĚēğççŽDāIJŕæŪzāŕsæŸŕāŕzāžŌèōĚéŪōāĠ;æŦŕçŽDēçŪæŋä;ġçŦlāĂĈăĬNāçCġijŦ

```
@wraps(func)
def wrapper(*args, **kwargs):
    wrapper.log.log(wrapper.level, wrapper.logmsg)
    return func(*args, **kwargs)

# Attach adjustable attributes
wrapper.level = level
wrapper.logmsg = logmsg
wrapper.log = log
```

èĚŽäyġæŪzæŦTāzŝāŔŕèČ;æ■cāyŷāüēä;IJġijŦNä;ĒāL■æŔŔæŸŕāōČāĤĚēāzæŸŕæIJĀād'ŪāsČçŽDēçĚēēŕāž  
āçČādIJāōČçŽDäyĤēĭçēŸæIJL'āŔēād'ŪçŽDēçĚēēŕāŽĬ(æŕŦāçCāyĤēĭçæŔŔāĤŕçŽD  
@timethis ä;Nā■Ŕ)ġijŦNēČcāzĤāōČāijŽēŽŔēŪŔāžŦāsČāsđæĀġġijŦNä;ġäĭŪāĤōæŦzāōČāžŋæŝæIJL'āzzä;Ŧ  
èĀŦēĀŽēġĠ;ġçŦlēōĚéŪōāĠ;æŦŕāŕsēČ;ēĀĤāĚ■ēŸæüçŽDāsĀēŽŔæĀģāĂĈ  
æIJĀāŔŌæŔŔäyĀçČġijŦNēŸŽäyĀāŕŔēĤČçŽDæŪzæĤĤāzŝāŔŕāzēä;IJäyž9.9āŕŔēĤCāy■èçĚēēŕāŽĬçszçŽ

## 11.6 9.6 āyēāŔŕéĀL'āŔČæŦŕçŽDēçĚēēŕāŽĬ

### éŪōéçŸ

ä;āæČŝāĒŽäyĀäyġēçĚēēŕāŽĬġijŦNæŪčāŔŕāzēäy■āijāāŔČæŦŕçzŽāōČġijŦNæŕŦāçČ  
@decorator ġijŦ āžŝāŔŕāzēäijäēĀŝāŔŕéĀL'āŔČæŦŕçzŽāōČġijŦNæŕŦāçČ  
@decorator(x, y, z) āĂĈ

### ēğçĀĒŝæŪzæĤĬ

äyŦēĭçæŸŕ9.5āŕŔēĤCāy■æŪēāĤŪēçĚēēŕāŽĬçŽDäyĀäyġāĤōæŦzçĤĬæIJġijŦŽ

```
from functools import wraps, partial
import logging

def logged(func=None, *, level=logging.DEBUG, name=None, _
    ↳message=None):
    if func is None:
        return partial(logged, level=level, name=name, _
    ↳message=message)

    logname = name if name else func.__module__
    log = logging.getLogger(logname)
    logmsg = message if message else func.__name__

    @wraps(func)
    def wrapper(*args, **kwargs):
        log.log(level, logmsg)
        return func(*args, **kwargs)
```

```
    return wrapper

# Example use
@logged
def add(x, y):
    return x + y

@logged(level=logging.CRITICAL, name='example')
def spam():
    print('Spam!')
```

ãŕŕäzëçIJŇäLŕijŇ@logged èçĚëĕŕäZlãŕŕäzëãŕŇæŮüäy■äyëãŕĆæŦŕæLŮäyëãŕĆæŦŕãĂĆ

## ëőléőž

èĚŽëĜŇæŕŕäLŕçŽĎèĚäyŕlëŮőëçŸŕŕsæŸŕéĂŽäyŕæL'ĂèŕŦ'çŽĎçijŮçlŇäyĂèĜŦ'æĂġéŮőëçŸãĂĆ  
â;ŞæĹSäzŇä;ĚçŦlëçĚëĕŕäZlçŽĎæŮüãĂŽiijŇäđ'ġéĈlãĹĚçlŇäžŕãŦŸäžæĈŕäžĚëçAäzĹäy■çžZãóĈäzŇäijäéĂ  
ãĚüãőđäzŌæĹĂæIJŕäyĹæŕlëëőšiiŇŇæĹSäzŇäŕŕäzëãőZäzL'äyĂäyŕæL'ĂæIJL'ãŕĆæŦŕéĈ;æŸŕãŕŕéĂL'çŽĎèçĚ

```
@logged()
def add(x, y):
    return x+y
```

ä;ĚæŸŕiijŇèĚŽçġ■ăĚZæşŦäžüäy■çŇëãŕĹæĹSäzŇçŽĎäžæĈŕiijŇæIJL'æŮüãĂŽçlŇäžŕãŦŸäŦŸèőŕãĹäã  
èĚŽëĜŇæĹSäzŇäŕŦŕä;ääŦŦçđ'žäžĚæĈä;ŦäžëäyĂèĜŦ'çŽĎçijŮçlŇéçŌæäijæŕlãŕŇæŮüæzæüşæşæIJL'æŇŇä

äyžäžĚçŕĚĚġçäzççăAæŸŕæĈä;Ŧäüëä;IJçŽĎiijŇä;ăĚIJĂëçAéĬäyŕçĚşæĈĹèçĚëĕŕäZlæŸŕæĈä;Ŧä;IJçŦ  
ărzäžŌäyĂäyŕlãĈŕäyŇéŕlëçĚæüçŽĎçőĂă■ŦëçĚëĕŕäZlŕiijŽ

```
# Example use
@logged
def add(x, y):
    return x + y
```

èĚŽäyŕlëŕĈçŦlãžŕãĹŮëüşäyŇéŕlçç■L'äzüiijŽ

```
def add(x, y):
    return x + y

add = logged(add)
```

èĚŽæŮüãĂŽiijŇèçŇèçĚëĕŕäĜ;æŦŕäijŽèçŇä;ŞăĂŽçŇŇäyĂäyŕlãŕĆæŦŕçZŦ'æŌëäijäéĂŞçžZ  
logged èçĚëĕŕäZlãĂĆ äZæ■đ'riijŇlogged() äy■çŽĎçŇŇäyĂäyŕlãŕĆæŦŕäŕŕsæŸŕèçŇäŇĚèçĚăĜ;æŦŕæIJŇè  
ëĂŇŕŕäžŌäyĂäyŕlãyŇéŕlëçĚæüæIJL'ãŕĆæŦŕçŽĎèçĚëĕŕäZlŕiijŽ

```
@logged(level=logging.CRITICAL, name='example')
def spam():
    print('Spam!')
```

ěřČčŤlázRáLŮěũşäyÑeİćç■L'ázũijŽ

```
def spam():
    print('Spam!')
spam = logged(level=logging.CRITICAL, name='example')(spam)
```

áLİägNěřČčŤl logged() áĜ;æŤŕæŮũijÑěćnáÑĚěćĚáĜ;æŤŕázũæşæIJL'äijăĚĂŞĕŹæİăĂĆ  
ăZăæ■d'ăIJlěćĚěěřăZlăĚĚijNăôČăĚĚăzæŸŕăŔŕéĂL'çŽĎăĂĆĕŹăyŕăŔ■ĕŹĜăİăijŽĕŹná;ŹăĚũăzŮăŔĆæŤŕ  
ăZũăyŤijNă;ĚĕŹăZăZăŔĆæŤŕěćnäijăĚĂŞĕŹæİăăŔŎijNěćĚěěřăZlăĚăĚŤăZďăyĂăyŕăŔŮăyĂăyŕăĜ;æŹ  
ăyžăZĚĕŹăăũăĂZijNăĚŤăZăZă;ŹčŤlázĚăyĂăyŕăŔĂăũģijNăŕşæŸŕăŤl'çŤl functools.  
partialăĂĆăôČăijŽĕŹŤăZďăyĂăyŕăŔIJăôŤăĚlăLİägNăŤŮçŽĎĚĜĕžŹijNěZď'ăZĚěćnáÑĚěćĚáĜ;æŤŕăďŤ  
ăŔŕăZăĚăŔĆĕĂĆ7.8ăŕŔĚĚĚĚŮăŔŮăZŤăďŤZ partial() æŮZăşŤçŽĎçşĕĕŹăĂĆ

## 11.7 9.7 áŤl'çŤlěćĚěěřăZlăijZăŤŮăĜ;æŤŕăyŤçŽĎçşZăďŤăĕĂăŞĕ

éŮőécŸ

ăIJăyZăşŔçğ■çijŮćİNěğĎçZĕijNă;ăæČşăIJlăŕZăĜ;æŤŕăŔĆæŤŕĕŹĚăŤăijZăŤŮçşZăďŤăĕĂăŞĕăĂĆ

èğcăĚşæŮZăæŹŤ

ăIJăijŤçďŤăôďĚĚăZčçăĂăŤ■ijNăĚŤĕŕŤæŸŎăŤŤăZăŹçŽĎçŽŮăăĜijZĕČ;ăŕZăĜ;æŤŕăŔĆæŤŕçşZăďŤăĕĂăŞĕăĂĆ

```
>>> @typeassert(int, int)
... def add(x, y):
...     return x + y
...
>>>
>>> add(2, 3)
5
>>> add(2, 'hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "contract.py", line 33, in wrapper
TypeError: Argument y must be <class 'int'>
>>>
```

ăyÑeİćæŸŕă;ŹčŤlěćĚěěřăZlăŤĂăIJŕăİăôďçŎŕ @typeassert ijŽ

```
from inspect import signature
from functools import wraps

def typeassert(*ty_args, **ty_kwargs):
    def decorate(func):
        # If in optimized mode, disable type checking
        if not __debug__:
            return func
```



```

# Map function argument names to supplied types
sig = signature(func)
bound_types = sig.bind_partial(*ty_args, **ty_kwargs).
↳arguments

@wraps(func)
def wrapper(*args, **kwargs):
    bound_values = sig.bind(*args, **kwargs)
    # Enforce type assertions across supplied arguments
    for name, value in bound_values.arguments.items():
        if name in bound_types:
            if not isinstance(value, bound_types[name]):
                raise TypeError(
                    'Argument {} must be {}'.format(name,
↳bound_types[name])
                )
            return func(*args, **kwargs)
    return wrapper
return decorate

```

åŕŕäzëçIJŇăĜžëŁŻäylëçĚëëŕăZÍlđäyÿçAŧæt'zñijŇæŮčăŔŕäzëæŇĜăŏŽæL'ĂæIJL'ăŔCæŦŕçşzăđŇñijŇăz  
 ăzŭăyŦăŔŕäzëëĂŽëŁĜă;■ç;ŏæLŮăĚşéTŏă■ŮăĬëæŇĜăŏŽăŔCæŦŕçşzăđŇăĂçăyŇéĬcæŸŕă;ŁçŦĬçđ'žă;ŇñijŽ

```

>>> @typeassert(int, z=int)
... def spam(x, y, z=42):
...     print(x, y, z)
...
>>> spam(1, 2, 3)
1 2 3
>>> spam(1, 'hello', 3)
1 hello 3
>>> spam(1, 'hello', 'world')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "contract.py", line 33, in wrapper
TypeError: Argument z must be <class 'int'>
>>>

```

## ëŏĬëŏŽ

ëŁŽëŁCæŸŕénŸçžğëçĚëëŕăZÍçđ'žă;ŇñijŇăñijŦăĚëăžĚă;Ĭăđ'ŽëĜ■ëçAçŽĐæçCăŁŧăĂĆ

ëçŮăĚĬñijŇëçĚëëŕăZÍăŔĬăñijŽăIJăĜ;æŦŕăŏŽăžL'æŮüëçñèŕÇçŦĬăyĂæŇăăĂĆ  
 æIJL'æŮŭăĂŽă;ăăŐzæŐL'ëçĚëëŕăZÍçŽĐăŁşëçñijŇéCçăžĬă;ăăŔĬëIJĂëçAçŏĂă■ŦçŽĐëŁŦăŽđëçñëçĚëëŕăĜ,  
 äyŇéĬçŽĐăžççăĂăy■ñijŇăçCăđIJăĚĬăşĂăŔŸéĜŔăĂĂ\_\_debug\_\_  
 ëçñëŏç;ŏæLŔăžĚFalse(ă;Şă;ăă;ŁçŦĬ-OæLŮ-ŐŐăŔCæŦŕçŽĐăñijŸăŇŮăĬăñijŔæL'ğëăŇçĬŇăžŔæŮŭ)ñijŇ  
 éCçăžĬăŕşçŽŦ'æŐëçŦăŽđæIJăŁŧăŏæŦžëŁĜçŽĐăĜ;æŦŕæIJñëžññijŽ

```
def decorate(func):
    # If in optimized mode, disable type checking
    if not __debug__:
        return func
```

inspect.signature() `inspect.signature()`

```
>>> from inspect import signature
>>> def spam(x, y, z=42):
...     pass
...
>>> sig = signature(spam)
>>> print(sig)
(x, y, z=42)
>>> sig.parameters
mappingproxy(OrderedDict([('x', <Parameter at 0x10077a050 'x'>),
('y', <Parameter at 0x10077a158 'y'>), ('z', <Parameter at 0x10077a1b0 'z'>)]))
>>> sig.parameters['z'].name
'z'
>>> sig.parameters['z'].default
42
>>> sig.parameters['z'].kind
<_ParameterKind: 'POSITIONAL_OR_KEYWORD'>
>>>
```

`bind_partial()`  
`bind_partial()`  
`bind_partial()`

```
>>> bound_types = sig.bind_partial(int, z=int)
>>> bound_types
<inspect.BindArguments object at 0x10069bb50>
>>> bound_types.arguments
OrderedDict([('x', <class 'int'>), ('z', <class 'int'>)])
>>>
```

`bound_types.arguments`  
`bound_types.arguments`  
`bound_types.arguments`

`sig.bind()`  
`sig.bind()`  
`sig.bind()`  
`sig.bind()`

```
>>> bound_values = sig.bind(1, 2, 3)
>>> bound_values.arguments
OrderedDict([('x', 1), ('y', 2), ('z', 3)])
>>>
```

ä;ŁçTlëfZäylæYäârDæŁSäznâRräzëä;Łë;zaİ;çŽDăôđçŎræŁSäznçŽDăijzâŁúçşzăđNăcĂæşëiijŽ

```
>>> for name, value in bound_values.arguments.items():
...     if name in bound_types.arguments:
...         if not isinstance(value, bound_types.arguments[name]):
...             raise TypeError()
...
>>>
```

äy■ëfGëfZäylæŬzæaŁëfYæIJL'çCzârRçSTçŬiijNăôČârzăžŎæIJL'ézYëôd'ăĂijçŽDăRĆæTřázúäy■éĂ  
ærTăeČâyNéİççŽDăzččăAăRräzëæ■câyÿăüëă;IJiijNâr;çôăitemşçŽDçşşăđNăYréTŽèrrççŽDiiijŽ

```
>>> @typeassert(int, list)
... def bar(x, items=None):
...     if items is None:
...         items = []
...         items.append(x)
...     return items
>>> bar(2)
[2]
>>> bar(2, 3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "contract.py", line 33, in wrapper
TypeError: Argument items must be <class 'list'>
>>> bar(4, [1, 2, 3])
[1, 2, 3, 4]
>>>
```

æIJAăRŎäyĂçCzæYřăĚşăžŎéĂČçTlëçĚëērăŽlăRĆæTřăŠNăĜ;æTřæşlëğçăzNéŬt'çŽDăžL'èôžăĂČ  
ă;NăeČiijNăyžăzĂăžLăy■ăČRăyNéİççŽæăăăĚZăyĂăyłëçĚëērăŽlăİăæşëăL'ăĜ;æTřăy■çŽDăşlëğçăSćiijş

```
@typeassert
def spam(x:int, y, z:int = 42):
    print(x, y, z)
```

äyĂäyłăRřëČ;çŽDăŎşăŽăæYřăçCăđIJă;ŁçTlăžEăĜ;æTřăRĆæTřæşlëğçëiijNéCčăžLăřşëcnéŽRăŁúăžEă.  
ăçCăđIJăşlëğçëcñçTlăİăăAŽçşşăđNăcĂæşëârşăy■ëČ;ăAŽăĚüăžŬăžNăČĚăžEăĂČëĂNăyT  
@typeassert äy■ëČ;ăĚ■çTlăžŎă;ŁçTlăşlëğçăĂăĚüăžŬăžNăČĚçŽDăĜ;æTřăžEăĂČ  
ëĂNă;ŁçTlăyŁéİççŽDëçĚëērăŽlăRĆæTřçAłæt'zæĂğăđ'ğăđ'ŽăžEiijNăžşæŽt'ăŁăéĂŽçTlăĂČ

ăRräzëăIJİPEP 362ăžëăRŁ inspect æłăăİŬäy■ăL'ăŁrăŽt'ăđ'ŽăĚşăžŎăĜ;æTřăRĆæTřăřzëşççŽDăŁă

## 11.8 9.8 ăřĚëçĚëērăŽlăôŽăžL'äyžçşzçŽDăyĂéČlăŁĚ

éŬóécŬ

ă;ăæČşăIJłçşăy■ăôŽăžL'ëçĚëērăŽlăiijNăžŭârEăĚüă;IJçTlăIJlăĚüăžŬăĜ;æTřăŁŬăŬzæşTăyŁăĂČ

## èġċàEṣæŨzæąŁ

ǎIJłśzézĠŃéIcǎŏŽǎzŁ'èċĚéērǎŽlǎŁŁçŏĀǎ■TijŃǎ;EæYřǎ;ǎéĕŨǎĚŁèĕAçqŏèŏd'ǎŏČçŽǦǎ;ŁçTlǎŨzǎijRǎ  
ǎyŃéIcǎŁSǎzŋçTlǎŁŃǎ■RǎIééYŘèřǎŏČǎzŋçŽǦǎy■ǎRŃijŽ

```
from functools import wraps

class A:
    # Decorator as an instance method
    def decorator1(self, func):
        @wraps(func)
        def wrapper(*args, **kwargs):
            print('Decorator 1')
            return func(*args, **kwargs)
        return wrapper

    # Decorator as a class method
    @classmethod
    def decorator2(cls, func):
        @wraps(func)
        def wrapper(*args, **kwargs):
            print('Decorator 2')
            return func(*args, **kwargs)
        return wrapper
```

ǎyŃéIcǎYřǎyǎǎ;ŁçTlǎŁŃǎ■RŃijŽ

```
# As an instance method
a = A()
@a.decorator1
def spam():
    pass

# As a class method
@A.decorator2
def grok():
    pass
```

ǎžTčzEĕġCǎřšǎRřǎžčǎRŚçŎřǎyǎǎylǎYřǎŏđǎŁNèřČçTlŃijŃǎyǎǎylǎYřçśzèřČçTlǎǎČ

## èŏłèŏž

ǎIJłśzǎy■ǎŏŽǎzŁ'èċĚéērǎŽlǎŁIŁIJŃǎyŁǎŎzǎē;ǎČRǎŁLǎēĠǎĀliijŃǎ;EæYřǎIJlǎǎĠǎĠEǎžŞǎy■ǎIJL'ǎŁ  
çŁ'zǎŁŋçŽǦijŃ@property èċĚéērǎŽlǎŏđéŽĚǎyŁǎYřǎyǎǎylçśzŃijŃǎŏČéĠŃéIcǎŏŽǎzŁ'ǎžEǎyŁ'ǎylǎŨzǎēŞ  
getter(), setter(), deleter() ,ǎřRǎyǎǎylǎŨzǎēŞTéČ;ǎYřǎyǎǎylèċĚéērǎŽlǎǎČǎŁŃǎēČiiž

```
class Person:
    # Create a property instance
    first_name = property()

    # Apply decorator methods
```

```

@first_name.getter
def first_name(self):
    return self.__first_name

@first_name.setter
def first_name(self, value):
    if not isinstance(value, str):
        raise TypeError('Expected a string')
    self.__first_name = value

```

aóČäyžāzÄāzŁëeAeŁZāzŁŁāōZāzŁ'čŽDāyžēeAāŌšāZāæYřāRĐçğ■āy■āRŇčŽDèčĚēēřāZÍāŮzæsŤaijZāI  
 property aóđäĴNāyŁæŞ■ā;IJāōČčŽDčŁŮæĀĀāĀĆ āZāæ■d'īijNāzžā;ŤæŮŮāĀZāRlèeAā;āččřāŁřeIJāèeAā

āIJłčšzāy■āōZāzŁ'èčĚēēřāZÍāIJL'āyłēŽ;čRĚèğččŽDāIJřæŮzārsæYřāřzāžŌéčIād'ŮāRĆæŤř  
 self æŁŮ cls čŽDæ■čçāōā;ŁčŤlāĀĆ āř;čōæeIJĀād'ŮāsČčŽDèčĚēēřāZÍāĠ;æŤřæřŤæČ  
 decorator1() æŁŮ decorator2() éIJāèeAæRŘā;ZāyĀāył self  
 æŁŮ cls āRĆæŤřīijN ā;EæYřāIJlāyđ'āyłēčĚēēřāZÍāĚĚčlècāŁZāzžčŽD  
 wrapper() āĠ;æŤřāžŮāy■éIJāèeAāNĚāRnèŁZāył self āRĆæŤřāĀĆ  
 ā;āāŤřāyĀēIJāèeAeŁZāyłāRĆæŤřæYřāIJlā;āçāōāōđeAeōŁēŮōāNĚēčĚāZlāy■ēŁZāyłāōđäĴNčŽDæšŘāžZēČ

āřzāžŌčšzéĠNĚícaōZāzŁ'čŽDāNĚēčĚāZlèŁYæIJL'āyĀčČzæřŤē;ČéŽ;čRĚèğčīijNārsæYřāIJlæŮL'āRĚāŁ  
 āĴNāēČīijNāAĠēōĴ;ā;āæČšēōĴ āIJlĀāy■āōZāzŁ'čŽDèčĚēēřāZÍā;IJčŤlāIJlā■ŘčšzBāy■āĀĆā;āeIJāèeAāČŘāyN

```

class B(A):
    @A.decorator2
    def bar(self):
        pass

```

āžšārsæYřèř'īijNèčĚēēřāZÍlèeAècāāōZāzŁ'æŁRčšzæŮzæsŤāžŮāyŤā;āāŁĚēāzæY;āijRčŽDā;ŁčŤlčŁŮčšz  
 ā;āāy■ēČ;ā;ŁčŤl @B.decorator2 īijNāZāāyžāIJlæŮzæsŤāōZāzŁ'æŮīīijNēŁZāyłčšzBēŁYæšæeIJL'ècāŁZ

## 11.9 9.9 āřĚèčĚēēřāZÍāōZāzŁ'āyžčšz

### éŮōéčŸ

ā;āæČšā;ŁčŤlāyĀāyłēčĚēēřāZÍāŌzāNĚēčĚāĠ;æŤřīijNā;EæYřāyNæIJžēŁŤāZđāyĀāyłāRřèřČčŤlčŽDāō  
 ā;āeIJāèeAeōĴ'ā;āčŽDèčĚēēřāZÍāRřāžēāRŇæŮŮāŮēā;IJāIJłčšzāōZāzŁ'čŽDāĚĚčĹāšNād'ŮēČlāĀĆ

### èğčāEşæŮzæāŁ

āyžāžEārĚèčĚēēřāZÍāōZāzŁ'æŁRāyĀāyłāōđäĴNīijNā;āeIJāèeAçāōāŁlāōČāōđčŌřāžE  
 \_\_call\_\_() āšN \_\_get\_\_() æŮzæsŤāĀĆ āĴNāēČīijNāyNēlččŽDāžččāAāōZāzŁ'āžĚāyĀāyłčšzīijNāōČā

```

import types
from functools import wraps

class Profiled:
    def __init__(self, func):

```

```
    wraps(func)(self)
    self.ncalls = 0

    def __call__(self, *args, **kwargs):
        self.ncalls += 1
        return self.__wrapped__(*args, **kwargs)

    def __get__(self, instance, cls):
        if instance is None:
            return self
        else:
            return types.MethodType(self, instance)
```

ä;ääRräzëärEäóČä;ŠäAŽäyÄäy!æŽóéĂŽçŽĐëčĚéërăŽ!æ!ëä;ŁçTl!ijNăIJ!çszezĜŇé!cæLŮăđ'Ůé!céČ;ăRŕ

```
@Profiled
def add(x, y):
    return x + y

class Spam:
    @Profiled
    def bar(self, x):
        print(self, x)
```

åIJ!ăzd'ăžŠçŮřăcČăy■çŽĐä;ŁçTl!çd'žă;Ň!ijŽ

```
>>> add(2, 3)
5
>>> add(4, 5)
9
>>> add.ncalls
2
>>> s = Spam()
>>> s.bar(1)
<__main__.Spam object at 0x10069e9d0> 1
>>> s.bar(2)
<__main__.Spam object at 0x10069e9d0> 2
>>> s.bar(3)
<__main__.Spam object at 0x10069e9d0> 3
>>> Spam.bar.ncalls
3
```

èó!èőž

ărEëçĚéërăŽ!ăóŽăzL'æLŔçszezĂŽăyÿæYřăŁçóĂă■TçŽĐăĂČă;EæYřèŁŽéĜŇèŁYæYřæIJL'ăyĂăžŽçzE  
éęŮăĚL!ijNă;ŁçTl functools.wraps() âĜ!æTřçŽĐä;IJçTlèu\$ăzNăL'■èŁYæYřăyĂæăü!ijNărEëcŇă  
ăĚŮæŇă!ijŇéĂŽăyÿăŁLăóžæYŠăijŽăŁ;èğEăyŁé!cçŽĐ  
æŮžæşTăĂČăçČăđIJă;ăăŁ;çTëăóČřijNăŁIæNĂăĚŮăžŮăžççăĂăy■ăRŸăE■æŇăèŁŘëăŇ!ijŇ

ä;äaijŽāRŠçŌřā;Šä;ääŌžērČçŤlēcñèçĚēērāōđä;ŊæŰžæşŤæŰüāGžçŌřā;ĹæĜæĀłçŽĐēŰōécŸāĀCä;ŊæĆřĩ

```
>>> s = Spam()
>>> s.bar(3)
Traceback (most recent call last):
...
TypeError: bar() missing 1 required positional argument: 'x'
```

āĜžēŤŽāŌşāZāæŸřā;ŞæŰžæşŤāĜ;æŤřāIJläyÄäyłçşzäy■ècnæşæL;æŰüiijŊāōČāznçŽĐ  
\_\_get\_\_() æŰžæşŤä;Ĺæ■ōæRŘèřřāZĹā■ŘèōōēcnērČçŤlīijŊ  
āIJĹ.9ārŘēĹCāũşçzŘèōşēřřēĜæRŘèřřāZĹā■ŘèōōāzĚāĀCāIJĹēŤŽēĜŊiijŊ\_\_get\_\_()  
çŽĐçŽōçŽĐæŸřāĹZāzzäyÄäyłçzŚāōŽæŰžæşŤāržèşā (æIJĀçZĹāijŽçzŽēŤŽäyĹæŰžæşŤāijæĀŞselfāRĆæŤřĩ)

```
>>> s = Spam()
>>> def grok(self, x):
...     pass
...
>>> grok.__get__(s, Spam)
<bound method Spam.grok of <__main__.Spam object at 0x100671e90>>
>>>
```

\_\_get\_\_() æŰžæşŤæŸřäyžäžĚçāōāŤłçzŚāōŽæŰžæşŤāržèşāēČ;ècnæ■ççāōçŽĐāĹZāzzāĀC  
type.MethodType() æĹŊāĹĹāĹZāzzäyÄäyłçzŚāōŽæŰžæşŤæĹēä;ŤçŤĹāĀCāRĹæIJĹ'ā;Śāōđä;Ŋēcnä;ŤçŤ  
āēČādIJēŤŽäyĹæŰžæşŤæŸřāIJłçşzäyĹēĹcæĹēēōŤēŰōriijŊ ēĆčāzĹ \_\_get\_\_() äy■çŽĐin-  
stanceāRĆæŤřāijŽēcnēōç;ōæĹRŊNoneāžüçŽt æŌēēŤāŽđ Profiled āōđä;ŊæIJñèžñāĀC  
ēŤŽæāüçŽĐērĹæĹSāznārřāRřāžæēRŘāRŰāōČçŽĐ ncalls āşđæĀğāžĚāĀC

āēČādIJā;āæČşēĀŤāĒ■äyĀāžZæüüāžřĩjŊāžşāRřāžēēĀČēŽŚāRēād' ŰäyĀäyĹā;ŤçŤĹēŰ■āŊĒāŞŊ  
nonlocal āRŸēĜRāōđçŌřçŽĐēçĚēērāZĹiijŊŊēŤŽäyĹāIJĹ.5ārŘēĹCæIJĹ'èōşāĹřāĀCä;ŊæĆřĩjŽ

```
import types
from functools import wraps

def profiled(func):
    ncalls = 0
    @wraps(func)
    def wrapper(*args, **kwargs):
        nonlocal ncalls
        ncalls += 1
        return func(*args, **kwargs)
    wrapper.ncalls = lambda: ncalls
    return wrapper

# Example
@profiled
def add(x, y):
    return x + y
```

ēŤŽäyĹæŰžāijŘēũşāžŊāĹ■çŽĐæŤĹæđIJāĜāāžŌäyĀæāüiijŊēŽđ'āžĚāržāžŌ ncalls  
çŽĐēōŤēŰōçŌřāIJæŸřēĀŽēŤĜäyĀäyłēcnçzŚāōŽäyžāşđæĀğçŽĐāĜ;æŤřæĹēāōđçŌřĩjŊä;ŊæĆřĩjŽ

```
>>> add(2, 3)
5
>>> add(4, 5)
9
>>> add.ncalls()
2
>>>
```

## 11.10 9.10 äÿžćśzǎŠŇéíŽæĀAæŮzæşŦæŘŘăĭŽèčĚéěřǎŽí

éŮóécŸ

äĵăæČşçžŽćśzǎĹŮéíŽæĀAæŮzæşŦæŘŘăĭŽèčĚéěřǎŽíăĂĆ

èğčǎĖşæŮzæǎĹ

çžŽćśzǎĹŮéíŽæĀAæŮzæşŦæŘŘăĭŽèčĚéěřǎŽíæŸřǎĭĹçõĂǎ■ŦçŽďĭĭŇăÿ■èĤĞèĕAçǎõăĤlèčĚéěřǎŽíăIJ  
@classmethod æĹŮ @staticmethod äžŇǎĹ■ăĂĆăĭŇǎĕĆĭĭž

```
import time
from functools import wraps

# A simple decorator
def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.time()
        r = func(*args, **kwargs)
        end = time.time()
        print(end-start)
        return r
    return wrapper

# Class illustrating application of the decorator to different
↳ kinds of methods
class Spam:
    @timethis
    def instance_method(self, n):
        print(self, n)
        while n > 0:
            n -= 1

    @classmethod
    @timethis
    def class_method(cls, n):
        print(cls, n)
        while n > 0:
```



```
        n -= 1

    @staticmethod
    @timethis
    def static_method(n):
        print(n)
        while n > 0:
            n -= 1
```

èċĒēēřăŔŎċŽĐċşăŝŇéİŽæĀAæŪzæşŦăŔŕæ■căyŷăũëă;IiijŇăŔlăy■ēġăċđăĹăăžĒċċĭăđ'ŪċŽĐĊŏqæŪ

```
>>> s = Spam()
>>> s.instance_method(1000000)
<__main__.Spam object at 0x1006a6050> 1000000
0.11817407608032227
>>> Spam.class_method(1000000)
<class '__main__.Spam'> 1000000
0.11334395408630371
>>> Spam.static_method(1000000)
1000000
0.11740279197692871
>>>
```

ëŏĹëŏž

ăĕĈăđĬJă;ăæĹĹċĒĒēēřăŽĬċŽĐăqžăžŔăĒŽĕŤŽăžĒăŕşăiŷŽăĠžĕŤŽăĂĈă;ŇăĕĈiijŇăAĠġĕŏ;ă;ăăĈŔăyŇéĬċ

```
class Spam:
    @timethis
    @staticmethod
    def static_method(n):
        print(n)
        while n > 0:
            n -= 1
```

éĈċăžĹă;ăĕŕĈċŦĹĕġZăyĹéİŽæĀAæŪzæşŦæŪăŕşăiŷŽæĹĕĕŤŽiijŽ

```
>>> Spam.static_method(1000000)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "timethis.py", line 6, in wrapper
start = time.time()
TypeError: 'staticmethod' object is not callable
>>>
```

éŬŏĕċŸăĬĴăžŎ @classmethod âŖŇ @staticmethod  
ăŏđĕŽĒăyĹăžăăy■ăiŷŽăĹŽăžăŕŕċŽŦ æŐĕĕŕĈċŦĹċŽĐăŕžĕşăiijŇ  
ĕĂŇæŸŕăĹŽăžžċĹ'žæŏĹċŽĐăŔŔĕĕŕăŽĹăŕžĕşă(ăŔĈĕĂĈ8.9ăŕŔĕĹĈ)ăĂĈăZăæ■đ'ă;Şă;ăĕŕŦċĬĂăĬĴăĒăžŪĕċ  
ċăŏăĹĕġŽċğ■ĕċĒēēřăŽĹăĠžċŐŕăĬĹĕċĒēēřăŽĹĕŞ;ăy■ċŽĐċŇăyĂăyĹă;■ċ;ŏăŔŕăžĕăġŏăđ'■ĕġZăyĹĕŬŏĕċŸăĂĈ

ā;ŠæĹŚāznāIJĹæĹ;èšāšžčšžy■āōŽāzĹ'čšžæŪzæšTāŠŅéiŽæĀAæŪzæšT(āRCèĀČ8.12ārRèĹĆ)æŪīijĹ  
ā;ŅāēČīijŅāēČæđIJā;āæČšāōŽāzĹ'āyĀāyĹæĹ;èšāčšžæŪzæšTīijŅāRfāzēā;ččTīčšžāijijāyŅéiČčŽDāžččāAīijŽ

```
from abc import ABCMeta, abstractmethod
class A(metaclass=ABCMeta):
    @classmethod
    @abstractmethod
    def method(cls):
        pass
```

āIJlèĹŽæōtāžččāAāy■īijŅ@classmethod èu\$ @abstractmethod  
āyđ'èĀĒčŽDēāžāžRæŸfæIJĹ'èōščl'ūčŽDīijŅāēČæđIJā;āērČæ■cāōČāznčŽDēāžāžRārsāijŽāGžéŤZāĀĆ

## 11.11 9.11 èĈĒēēřāŽlāyžècñāŅĒèĉĒāĜ;æŤřāćđāĹāāRĆæŤř

éŬóécŸ

ā;āæČšāIJlèĉĒēēřāŽlāy■čžŽècñāŅĒèĉĒāĜ;æŤřāćđāĹāéćĹāđ'ŪčŽDāRĆæŤřīijŅā;ĒæŸřāy■èČ;ā;śā\$■èĹZ

èğčāĒšæŪzæāĹ

āRfāzēā;ččTīāĒšéŤōā■ŪāRĆæŤřālèčžŽècñāŅĒèĉĒāĜ;æŤřāćđāĹāéćĹāđ'ŪāRĆæŤřāĀĆèĀČèŽŚāyŅéiČ

```
from functools import wraps

def optional_debug(func):
    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)

    return wrapper
```

```
>>> @optional_debug
... def spam(a,b,c):
...     print(a,b,c)
...
>>> spam(1,2,3)
1 2 3
>>> spam(1,2,3, debug=True)
Calling spam
1 2 3
>>>
```

éĀžēfĠècĒēēřāZlælēçzZècñāNĒècĒāĠ;æTřácđāŁāāRĆæTřçŽĐāŽæşTāzūäy■āyÿèġAāĀĆ  
 řř;çōāāēĆā■d'ijjNæIJL æUūāĀŽāōĆāRřāzēēAāāĒāyĀāžŽēĠ■ād'■āzčcāAāĀĆā;NāēCřijNāēĆāđIJā;ăæIJL'

éĆčázŁä;ǻǻŔřázěǻřĚǻĚűéĜ■æđĎǻĹŘèŁZæǻűijŽ

```

    ěfZčg■āōđčŎřæŮzæqLāzŇæL'ĀāzēēāŇā;ŮēĀŽiijŇāIĴāžŎāijžāLūāĚšēTōā■ŮāRĆæTřā;LāōzæŸŠēčná
    *args āšŇ **kwargs āRĆæTřčŽDāĜ;æTřäy■āĀĆ ēĀŽēĜā;ĤčTīāijžāLūāĚšēTōā■ŮāRĆæTřiijŇāōČēčná
    āzūāyTāŎēāyŇāĪēāzĚāzĚā;ĤčTīāL'ā;ŽčŽDā;■č;ōāšŇāĚšēTōā■ŮāRĆæTřāŎžēřČčTīēfZāyĪāĜ;æTřæŮūiij
    āžšāršæŸřēřtīijŇāōČāzūāy■āijŽēčncžšāĚēāĴr **kwargs āy■āŎžāĀĆ

```

ɛ̯f̥ȳæIJL̥äy̯Ääy̯l̥ēZ̥;çC̥Z̥a̯rs̥æȳra̯c̥ä;T̥a̯Ō̯z̥a̯d̥D̥çR̥Ė̯e̯c̥n̥æ̯u̯z̥a̯L̥äçZ̥D̥a̯R̥C̥æTr̥äy̯Ō̯e̯c̥n̥a̯N̥Ė̯e̯c̥Ė̯a̯Ĝ̥;æTr̥äR̥C̥

ä;ŇäĉĆiijŇäĉĆädIJēčĚēřǺÍ @optional\_debug ä;IJçŤlǺIJlǺyÄäyĽǻšçzŔäŇäeIJL'äyÄäyĽ  
debug ǺŔĆäŤŕçŽĐǺĜ;ǺŤŕǺyĽäŮüäijŽäIJL'ēŮóéćŸǺǺĆ ēĚŽéĜŇäĽSǺžǺǺđǺĽǺǺžĚǺyÄäēǺŔǺǺŮǺčǺǺ  
äyĽēĽēçŽĐäŮžǺǺĽēŸǺŔŕǺžēǺŽŤ ǺōŇç;ŎäyǺĈçziiijŇǺŽǺäyžçšç;ǺŸŎçŽĐçĽŇǺžŔǺŸŸǺžŤēřǺŔŖçŎŕǺž

```
>>> @optional_debug
... def add(x,y):
...     return x+y
...
>>> import inspect
>>> print(inspect.signature(add))
(x, y)
>>>
```

éǺŽēĚĜǺĉĆäyŇçŽĐǺĽōǺŤziiijŇǺŔŕǺžēēĝčǺĚşēĚŽǺyĽēŮóéćŸiiijŽ

```
from functools import wraps
import inspect

def optional_debug(func):
    if 'debug' in inspect.getargspec(func).args:
        raise TypeError('debug argument already defined')

    @wraps(func)
    def wrapper(*args, debug=False, **kwargs):
        if debug:
            print('Calling', func.__name__)
        return func(*args, **kwargs)

    sig = inspect.signature(func)
    parms = list(sig.parameters.values())
    parms.append(inspect.Parameter('debug',
                                    inspect.Parameter.KEYWORD_ONLY,
                                    default=False))
    wrapper.__signature__ = sig.replace(parameters=parms)
    return wrapper
```

éǺŽēĚĜēĚŽǺǺüçŽĐǺĽōǺŤziiijŇǺŇĚēčĚǺŔŎçŽĐǺĜ;ǺŤŕçǺǺǺŔǺŕşēĈ;ǺǺççǺōçŽĐäŸçđ'ž  
debug ǺŔĆäŤŕçŽĐǺǺŸǺIJǺžĚǺǺĆä;ŇäĉĆiijŽ

```
>>> @optional_debug
... def add(x,y):
...     return x+y
...
>>> print(inspect.signature(add))
(x, y, *, debug=False)
>>> add(2,3)
5
>>>
```

ǺŔĆēǺĈ9.16ǺŕŔēĽĆēŎüǺŔŮǺžŤŤǺđ'ŽǺĚşǺžŎǺĜ;ǺŤŕçǺǺǺŔǺŽĐǺĽǺǺǺŔǺǺĈ

čšzèĚēēřăZléĀŽăyŷăRfăzēă;IJăyžăĚūăzŮénŸčžġēĹĂēIJrăřŤăĉCăuăăĚēăĹŮăĚĈşşçŽĎăŸĂçğ■ēĭđă  
ăřŤăĉCīijŊăŷĹēĭcđŷă;Ŋăŷ■ĈŽĎăRēăđŮăŸĂçğ■ăōđĉŌřă;ġĉŦĭăĹŕĉžġēĹĤīijŽ

```
class LoggedGetattribute:
    def __getattribute__(self, name):
        print('getting:', name)
        return super().__getattribute__(name)

# Example:
class A(LoggedGetattribute):
    def __init__(self, x):
        self.x = x
    def spam(self):
        pass
```

èŁŻçġæŰzæŁăzşèaŃăĹ ŰéĂŹiijŃăĹEæŸřăŷzăEăŌzçŖEèġcăŌCŕiijŃăĹăăřsăŁĒéəzçşééAşæŰzæşŤřČ  
ăzèăŖĹăĒŰăŏČ8.7ăŖĒĹCăzŃçzġŻDçzgæŁŁçşşèřEăĂĆ æşŖçġġĹŃăzèăŷŁæĹèèŏŕiijŃçşzèçĒéěřăŹĹăŰzæă  
ăZăăŷzăŰăzŰăŷăġĹĹŤŰ super() âĢĵæŤŕăĂĆ

ăĕĆăđĬăĵăçşzăĆşăĬĬăŷĂăŷĹçşzăŷĹĹcăĵŁçŤĹăđ'ŹăŷĹçşzèçĒéěřăŹĹiijŃéĆcăzĹăăřséĬĂĒĕAæşĹăĎŖăŷŃé  
ăĹŃăĕCŕiijŃăŷĂăŷĹççĒéěřăŹĹĂăijŹăŖEăĒŰèçĒéěřçŹĐăŰzæşŤăŏŃăŤŤ æŹŁæġcăĹŖăŖăŷĂçġăăŏđçŎŕiijŃ  
èĂŃăŖăŷĂăŷĹççĒéěřăŹĹBăŖĹăŸřçŏĂăŤŤŹĐăĬĬăĒŰèçĒéěřçŹĐăŰzæşŤăŷăăŷăŁăçĆzéćĹăđ' ŰéĂzèĹŠăĂ  
éĆcăzĹĹĕŹăŰăĂŹèçĒéěřăŹĹĂăřséĬĂĒĕAæŤĹăĬĬççĒéěřăŹĹBçŹĐăĹăĹcăĂĆ

ăĵăĕŸăŖăřăzèăŹđéăĵăŷăŷŃ8.13ăŖĒĹCăŖăđ' ŰăŷĂăŷĹăĒşăžŎçşzèçĒéěřăŹĹçŹĐăĬĬçŤĹçŹĐăĹŃăŖŖ

## 11.13 9.13 äĹŁçŤĹăĒĆçşzæŎġăĹŰăŏđăĹŃçŹĐăĹZăžž

### éŰŏéćŸ

ăĵăæČşéĂŹĕŁĢăŤzăŖŸăŏđăĹŃăĹZăžžæŰzăĵŖăĹăŏđçŎŕăŤăĹŃăĂăĵijşăŸăĹŰăĒŰăzŰçşzăĵijçŹĐ

### èġcăĒşæŰzæăĹ

PythonçĹŃăžŖăŖŸĒçĹçşééAşŷiijŃăĕĆăđĬăĵăăăŹăzĹăžEăŷĂăŷĹçşzŷiijŃăřséČĹăČŖăĢĵæŤŕăŷĂăăŷçŹĐè

```
class Spam:
    def __init__(self, name):
        self.name = name

a = Spam('Guido')
b = Spam('Diana')
```

ăĕĆăđĬăĵăăăČşèĢăŏŹăzĹĹĕŁŹăŷĹăăĕĹđ'ŕiijŃăĵăăŖăřăžăŏŹăzĹăŷĂăŷĹăĒĆçşzăžŰéĢăŷăăŏđçŎŕ  
\_\_call\_\_() æŰzæşŤăĂĆ

ăŷžăžEăĵijŤçđ' ŷiijŃăĂĢèŏĹăĵăăŷăăČşăžzăĵăžžăĹZăžžèŁŹăŷĹçşzçŹĐăŏđăĹŃiijŹ

```
class NoInstances(type):
    def __call__(self, *args, **kwargs):
        raise TypeError("Can't instantiate directly")
```

```
# Example
class Spam(metaclass=NoInstances):
    @staticmethod
    def grok(x):
        print('Spam.grok')
```

è£ZæũçŽĐèrĩĩjNçTlæLũaRlèCjèrCçTlè£ZäyłçszçŽĐéiZæĀAæŪzæşTĩijNèĀNäy■èCjă;£çTlèĀŽäyç;

```
>>> Spam.grok(42)
Spam.grok
>>> s = Spam()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "example1.py", line 7, in __call__
    raise TypeError("Can't instantiate directly")
TypeError: Can't instantiate directly
>>>
```

çÖråIJiijÑāAĞāēĆä;ǣČśāōđçÖřā■Ță;ŊæĺāiijRiijLăRĭēČ;ǎLZǎzzaŤrăyĂăōđă;ŊçZĐçśziiLriijÑăōđçĆ

```
class Singleton(type):
    def __init__(self, *args, **kwargs):
        self.__instance = None
        super().__init__(*args, **kwargs)

    def __call__(self, *args, **kwargs):
        if self.__instance is None:
            self.__instance = super().__call__(*args, **kwargs)
            return self.__instance
        else:
            return self.__instance

# Example
class Spam(metaclass=Singleton):
    def __init__(self):
        print('Creating Spam')
```

éĆčázŁSpamçsżåršårĹèĈ;ǎŁZǎzǎǎŤřǎyǎÇŽĐǎođǎĹNǎžĚiijŇǎeijŤčđ'žǎeĆǎyŇiijŽ

```
>>> a = Spam()
Creating Spam
>>> b = Spam()
>>> a is b
True
>>> c = Spam()
>>> a is c
True
>>>
```

æIJAǎRŎijNǎAǾèø;ǎǎæČšǎLZǎzz8.25ǎrRèLĆäy■éĆcæǎuçZǾçijŠǎ■Yǎođä;NǎĀĆäyNéIcæĹSǎznǎRǎ

```

import weakref

class Cached(type):
    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.__cache = weakref.WeakValueDictionary()

    def __call__(self, *args):
        if args in self.__cache:
            return self.__cache[args]
        else:
            obj = super().__call__(*args)
            self.__cache[args] = obj
            return obj

# Example
class Spam(metaclass=Cached):
    def __init__(self, name):
        print('Creating Spam({!r})'.format(name))
        self.name = name

```

čDúăŔŌæĹSăz\$æĬæŧNêŕTăyĂăyŊiijŽ

```

>>> a = Spam('Guido')
Creating Spam('Guido')
>>> b = Spam('Diana')
Creating Spam('Diana')
>>> c = Spam('Guido') # Cached
>>> a is b
False
>>> a is c # Cached value returned
True
>>>

```

èóĭèőž

ăĹĹ'çŦĭăĚčŝzăôđçŎŕăđ'Žçğ■ăôđăĭŊăĹZăzžæĭăiijRéĂŽăyÿëęAæŕTăy■ăĭ;ŁçŦĭăĚčŝzçŽĐæŰzăiijŔăiijŸ  
ăĂĠèőĭăĭăăy■ăĭ;ŁçŦĭăĚčŝzĭiijŊăĭăăŕŕèčĭéĬĂèęAăŕEçŝzéŽŔèŰŔăĬĬăŝŔăžZăûëăŎĆăĠ;æŧŕăŔŎéĭcăĂ  
æŕŦăęCăyžăžEăôđçŎŕăyĂăyĭă■ŦăĭŊiijŊăĭăăăăŕŕèčĭăiijŽăčŔăyŊéĭcèŁŽæăăăEŽiijŽ

```

class _Spam:
    def __init__(self):
        print('Creating Spam')

_spam_instance = None

def Spam():
    global _spam_instance

```



```
if _spam_instance is not None:
    return _spam_instance
else:
    _spam_instance = _Spam()
    return _spam_instance
```

år;çõä;£çŦlăĚĈşzâRřèĈ;äijŽæŭL'âRŁăLræŦè;ĈénŸçžğĈzçŽDæŁĂæIJřijŇă;EæŸřăõĈçŽDăžçăA  
æŽt'ad'ŽăĚşzžŎăLŽăžžçijŞă■Ÿăõđă;ŇăĂăiįsăijŦçŦlç■L'ăĚĚăõžiiŇŇěruăRĈèĂĈ8.25ărRèŁĈăĂĈ

## 11.14 9.14 æ■ŦèŎŭçşzçŽDăşđæĂğăŎŽăžL'éąžăžŘ

éŬŏécŸ

ă;ăæĈşèĠlăLlèŏřă;ŦăŸĂăŸlçşzăŸ■ăşđæĂğăŇŇæŰzæşŦăŏŽăžL'çŽDéąžăžŘiiŇ  
çDŭăRŎăRřăžăăLŦçŦlăŏĈăĭăĂŽă;Lăđ'ŽăŞ■ă;IJiiŦLæŦŦăĈăžRăLŬăŇŬăĂĂæŸăărDăLŦæŦŦæ■ŏăžç■L'ç

èğĉăĚşæŰzæąĹ

ăLŦ'çŦlăĚĈşzâRřăžăă;LăŏžæŸŞçŽDæ■ŦèŎŭçşzçŽDăŏŽăžL'ăĤăæĂřăĂĈăŸŇéĬæŸřăŸĂăŸlă;Ňă■ŘiiŇ

```
from collections import OrderedDict

# A set of descriptors for various types
class Typed:
    _expected_type = type(None)
    def __init__(self, name=None):
        self._name = name

    def __set__(self, instance, value):
        if not isinstance(value, self._expected_type):
            raise TypeError('Expected ' + str(self._expected_type))
        instance.__dict__[self._name] = value

class Integer(Typed):
    _expected_type = int

class Float(Typed):
    _expected_type = float

class String(Typed):
    _expected_type = str

# Metaclass that uses an OrderedDict for class body
class OrderedMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        d = dict(clsdict)
```

```

        order = []
        for name, value in clsdict.items():
            if isinstance(value, Typed):
                value._name = name
                order.append(name)
        d['_order'] = order
        return type.__new__(cls, clsname, bases, d)

    @classmethod
    def __prepare__(cls, clsname, bases):
        return OrderedDict()

```

OrderedDict`  
 ``\_order`  
 OrderedDict`  
 ``\_order`

```

class Structure(metaclass=OrderedMeta):
    def as_csv(self):
        return ','.join(str(getattr(self, name)) for name in self._
        order)

# Example use
class Stock(Structure):
    name = String()
    shares = Integer()
    price = Float()

    def __init__(self, name, shares, price):
        self.name = name
        self.shares = shares
        self.price = price

```

OrderedDict`

```

>>> s = Stock('GOOG', 100, 490.1)
>>> s.name
'GOOG'
>>> s.as_csv()
'GOOG,100,490.1'
>>> t = Stock('AAPL', 'a lot', 610.23)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "dupmethod.py", line 34, in __init__
TypeError: shares expects <class 'int'>
>>>

```

## ěóíěőž

æIJñĚĹĆäyÄäylăĚşéŤőçĆzărşæŸrOrderedMetaăĚĈşşzäy■ăőŽăzĹ'čŽĎ “ \_\_pre-  
pare\_\_()“ æŮzæşŤăĂĆ ěĚŽăylăŮzæşŤăijŽăIJăijĂăġŃăőŽăzĹ'çşşăŤŃăőĈçŽĎĈĹúçşşçŽĎæŮúăĂŽěćŋæĹ'ğ  
æĹŚăznĚĚŹéĠŇéĂŽĚĚĠĚŤăŽďăžĚăyĹOrderedDictĚĂŇăy■æŸrăyÄäylăŽőéĂŽçŽĎă■ŮăĚyġijŇăŔrăžĚăĹăőžæŸşçŽĎæĹ'ŤăşŤĚĚŽăylăĹşĚĈ;ă

ăĚĈăđIJă;ăæĈşăđĎĚĂăĚĠăŮşçŽĎçşşă■ŮăĚyăržĚşăġijŇăŔrăžĚăĹăőžæŸşçŽĎæĹ'ŤăşŤĚĚŽăylăĹşĚĈ;ă

```
from collections import OrderedDict

class NoDupOrderedDict(OrderedDict):
    def __init__(self, clsname):
        self.clsname = clsname
        super().__init__()
    def __setitem__(self, name, value):
        if name in self:
            raise TypeError('{} already defined in {}'.format(name, self.clsname))
        super().__setitem__(name, value)

class OrderedMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        d = dict(clsdict)
        d['_order'] = [name for name in clsdict if name[0] != '_']
        return type.__new__(cls, clsname, bases, d)

    @classmethod
    def __prepare__(cls, clsname, bases):
        return NoDupOrderedDict(clsname)
```

ăyŇéĹćæĹŚăznăŤŇĚŤĚĠăđ'■čŽĎăőŽăzĹ'ăijŽăĠĠçŤőřăžĂăžĹăĈĚăĚġijŽ

```
>>> class A(metaclass=OrderedMeta):
...     def spam(self):
...     pass
...     def spam(self):
...     pass
...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 4, in A
  File "dupmethod2.py", line 25, in __setitem__
    (name, self.clsname))
TypeError: spam already defined in A
>>>
```

æIJăăŔŔŔőĚŸæIJĹăyĂçĆzăĹĹéĠ■ĚĚĂġijŇăŕşæŸŕăIJĹ \_\_new\_\_()  
æŮzæşŤăy■ăržăžŔăĚĈşşzäy■ĚćŋăġőăŤžă■ŮăĚyçŽĎăđ'ĎĈŔĚăĂĆ  
ăr;çőăçşşă;ċŤĹăžĚăŔĚăđ'ŮăyÄäylă■ŮăĚyăĹĚăőŽăzĹ'ġijŇăIJăđĎĚĂăæIJĂçzĹçŽĎ class  
ăržĚşăçŽĎæŮúăĂŽġijŇæĹŚăznăž■ĎŮĚIJĂĚĚĂărĚĚŤăylă■ŮăĚyĚ;ŋă■ćăyžăyÄäylă■čćăőçŽĎ  
dict ăőđăĹŇăĂĆ ĚĂŽĚĚĠĚăŔĚ d = dict(clsdict)



```

# Custom processing
pass
return super().__prepare__(name, bases)

# Required
def __new__(cls, name, bases, ns, *, debug=False, ↵
↵synchronize=False):
    # Custom processing
    pass
    return super().__new__(cls, name, bases, ns)

# Required
def __init__(self, name, bases, ns, *, debug=False, ↵
↵synchronize=False):
    # Custom processing
    pass
    super().__init__(name, bases, ns)

```

## èóìèőž

čžŽäyÄäyĹaĚČšzæûzâĹ.ääRréĀL'āĖšéTōā■ŪāRĆæTřéIJĀèēAä;āāōNāĒĹāijDæĠCšzāĹZāzzčŽDæL'Āa  
āZāyžèĚŽāžŽāRĆæTřāijŽècnāijæĀŠčžZæRāyÄäyĹčŽyāĖščŽDæŪzæšTāĀĆ  
\_\_prepare\_\_() æŪzæšTāIJĹæL'ĀæIJL'čšzāōŽāzL'āijAāgNæL'gèaŊāL'■ēēŪāĒĹècnèrČčTĹijNčTĹæĹēāĹZ.  
éĀŽāyŷāĹēēōšrijNèĚŽāyĹæŪzæšTāRĹæYřčōĀā■TčŽDèĚTāZđāyÄäyĹa■ŪāĖyæĹŪāĒūāzŪāYāārDāržēšāāĀĆ  
\_\_new\_\_() æŪzæšTčēncTĹæĹēāōđā;ŊāNŪāIJĀčžĹčŽDčšzāržēšāāĀĆāōČāIJčšžčŽDāyžā;ŠècnæL'gèaŊāō  
\_\_init\_\_() æŪzæšTæIJĀāRŌècnèrČčTĹijNčTĹæĹæL'gèaŊāĒūāzŪčŽDāyÄāžZāĹiāgŊāNŪāūēā;IJāĀĆ  
ā;ŠæĹSāznæđDēĀāāĚČšžčŽDæŪūāĀŽrijNēĀŽāyŷāRĹēIJĀèēAāōŽāzL'āyÄäyĹ  
\_\_new\_\_() æĹŪ \_\_init\_\_() æŪzæšTrijŊā;Ėāy■æYřāyđ'āyĹēČ;āōŽāzL'āĀĆ  
ā;ĖæYřrijŊāēČæđIJēIJĀèēAæŌēāRŪāĒūāzŪčŽDāĖšéTōā■ŪāRĆæTřčŽDèřrijNèĚŽāyđ'āyĹæŪzæšTāršēēAā  
ézYēōđ'čŽD \_\_prepare\_\_() æŪzæšTæŌēāRŪāzzæĎRčŽDāĖšéTōā■ŪāRĆæTřrijŊā;ĖæYřāijŽāĤ;čTēāō  
æL'ĀāžēāRĹæIJĹā;ŠēĚŽāžŽēčĹāđ'ŪčŽDāRĆæTřāRřēČ;āijZā;sāŠ■āĹřčšzāŠ;āR■čĹ'žēŪřčŽDāĹZāzzæŪūā;āā  
\_\_prepare\_\_() æŪzæšTāĀĆ  
éĀŽèĚGā;ĤčTĹāijžāĹūāĖšéTōā■ŪāRĆæTřrijŊāIJčšžčŽDāĹZāzzèĚĠčĹNāy■æĹSāznāĤĖēāžéĀŽèĚGāĖš  
ā;ĤčTĹāĖšéTōā■ŪāRĆæTřēĒ■č;ōāyÄäyĹaĚČšžzèĚYāRřāžēēgĖā;IJāržčšzāRŸéĠRčŽDāyĀčg■æZĤāžčæĹ

```

class Spam(metaclass=MyMeta):
    debug = True
    synchronize = True
    pass

```

ārĖēĚŽāžZāšđæĀgāōŽāzL'āyžāRĆæTřčŽDāē;āđ'ĎāIJĹāžŌāōČāznāy■āijŽæšāēšŠčšžčŽDāR■čġřčĹ'žēŪř  
èĚŽāžZāšđæĀgāžĖāžĖāRĹāžŌāšđāžŌčšžčŽDāĹZāžžēYūāōřrijNēĀŊāy■æYřčšžāy■čŽDèř■āRēæL'gèaŊēYūā  
āRēāđ'ŪrijŊāōČāznāIJĹ \_\_prepare\_\_() æŪzæšTāy■æYřāRřāžēēcnēōĖēŪōčŽDrijŊāZāāyžèĚŽāyĹæŪzæšT  
ā;ĖæYřčšzāRŸéĠRāRĹēČ;āIJĹāĚČšžčŽD \_\_new\_\_() āŠŊ \_\_init\_\_()  
æŪzæšTāy■āRřēgĖāāĀĆ

```
>>> def func(*args, **kwargs):
...     bound_values = sig.bind(*args, **kwargs)
...     for name, value in bound_values.arguments.items():
...         print(name, value)
...
>>> # Try various examples
>>> func(1, 2, z=3)
x 1
y 2
z 3
>>> func(1)
x 1
>>> func(1, z=3)
x 1
z 3
>>> func(y=2, x=1)
x 1
y 2
>>> func(1, 2, 3, 4)
Traceback (most recent call last):
...
```

```

File "/usr/local/lib/python3.3/inspect.py", line 1972, in _bind
    raise TypeError('too many positional arguments')
TypeError: too many positional arguments
>>> func(y=2)
Traceback (most recent call last):
...
File "/usr/local/lib/python3.3/inspect.py", line 1961, in _bind
    raise TypeError(msg) from None
TypeError: 'x' parameter lacking default value
>>> func(1, y=2, x=3)
Traceback (most recent call last):
...
File "/usr/local/lib/python3.3/inspect.py", line 1985, in _bind
    '{arg!r}'.format(arg=param.name))
TypeError: multiple values for argument 'x'
>>>

```

aŕŕäzëçIJŇäGzæİëijNéÄŽëfGârEç■;äŕ■äŠNäijäëÄŠçŽDäŕCæTŕçzŠäóŽëtuæİëijNäŕŕäzëäijžäLúäG;
 äyNéİcæYŕäyÄäyläijžäLúäG;æTŕç■;äŕ■æŽt'äËüä;ŞçŽDä;Nä■ŔäÄCäIJläzççäAäy■ijNæLŠäznäIJläşçç
 \_\_init\_\_() æŰzæşTüijN çDüäŔÖæLŠäznäijžäLúäL'ÄæIJL'çŽDä■ŔçşzäſËéazæŔŔä;ŽäyÄäylçL'žäóŽçŽ

```

from inspect import Signature, Parameter

def make_sig(*names):
    parms = [Parameter(name, Parameter.POSITIONAL_OR_KEYWORD)
              for name in names]
    return Signature(parms)

class Structure:
    __signature__ = make_sig()
    def __init__(self, *args, **kwargs):
        bound_values = self.__signature__.bind(*args, **kwargs)
        for name, value in bound_values.arguments.items():
            setattr(self, name, value)

# Example use
class Stock(Structure):
    __signature__ = make_sig('name', 'shares', 'price')

class Point(Structure):
    __signature__ = make_sig('x', 'y')

```

äyNéİcæYŕä;ŧçTİëfZäyI Stock çşççŽDçd'žä;NüijŽ

```

>>> import inspect
>>> print(inspect.signature(Stock))
(name, shares, price)
>>> s1 = Stock('ACME', 100, 490.1)
>>> s2 = Stock('ACME', 100)
Traceback (most recent call last):

```





```
>>> import inspect
>>> print(inspect.signature(Stock))
(name, shares, price)
>>> print(inspect.signature(Point))
(x, y)
>>>
```

## 11.17 9.17 aIłĆśżäyŁaijżăŁúä;£çŤíçijŮćíNèğĐçžę

### éŮóécŸ

ä;ăçŽĐćíNăžŔăNĚăŔnăyĂăyłă;Łăd'ğçŽĐçśžçžğæŁ'£ă;ŞçşzriiNă;ăăyNăIJŽăijżăŁúæŁ'ğèąNă\$ŔăžŽçij

### èğcăEşæŮzæąŁ

ăęĆăđIJă;ăæČşçŽŚæŮğçśžçŽĐăŏŽăžŁ'riiNéĂŽăyyăŔŕăžëéĂŽë£ĞăŏŽăžŁ'ăyĂăyłăĚČçśžăĂĆăyĂăyłăş  
 type ăžúéĞăăŏŽăžŁ'ăŏČçŽĐ \_\_new\_\_() æŮžæşŤ æŁŮèĂĚæŸŕ \_\_init\_\_()  
 æŮžæşŤăĂĆăŕŤăęĆriiŽ

```
class MyMeta(type):
    def __new__(self, clsname, bases, clsdict):
        # clsname is name of class being defined
        # bases is tuple of base classes
        # clsdict is class dictionary
        return super().__new__(cls, clsname, bases, clsdict)
```

ăŔęăyĂçğăæŸriiNăŏŽăžŁ' \_\_init\_\_() æŮžæşŤriiŽ

```
class MyMeta(type):
    def __init__(self, clsname, bases, clsdict):
        super().__init__(clsname, bases, clsdict)
        # clsname is name of class being defined
        # bases is tuple of base classes
        # clsdict is class dictionary
```

ăyžăžEă;£çŤíë£ŽăyłăĚČçśzriiNă;ăéĂŽăyyëëAăŕEăŏČăŤ;ăŁŕăŁŕăyĂăyłăęăŭçžğçŁúçśžăŏŽăžŁ'ăyriiNçł

```
class Root(metaclass=MyMeta):
    pass

class A(Root):
    pass

class B(Root):
    pass
```

ãĖĈçşzçŽĎäyÄäyĭäĖşéŤôçL'zçCzæŸřăôĈăĖĀēōyăĭăăĭĬăôŽăzL'çŽĎæŮŭăĂŽæĈĂæşēçşzçŽĎăĖĖăôză  
\_\_init\_\_() æŰzæşŤäyĭĭijŇăĭăăŖřăzēăĬĖ;zæĬçŽĎæĈĂæşēçşzăŮăĖŸăĂĀçĬŮçşzçĬ'çĬ'ăĂĈăzŭăyŤ  
ăŽăæĎ'ĭijŇăyÄäyĭäēĀēđŭçŽĎăđĎăzžēĂĖăřşēĈĭăĭĬăđ'găđŇçŽĎçzğæL'făĭŞçşzăyĖĂŽēĤĖçzŽăyÄäyĭēăŭ  
ăĭĬăyžăyÄäyĭäĖŭăŞçŽĎăžŤçŤĭăĬŇăŖĭijŇăyŇéĭăôŽăzL'ăžĖăyÄäyĭäĖĈçşziijŇăôĈăijŽăŇŞçziăzăăĭ

```
class NoMixedCaseMeta(type):
    def __new__(cls, clsname, bases, clsdict):
        for name in clsdict:
            if name.lower() != name:
                raise TypeError('Bad attribute name: ' + name)
        return super().__new__(cls, clsname, bases, clsdict)

class Root(metaclass=NoMixedCaseMeta):
    pass

class A(Root):
    def foo_bar(self): # Ok
        pass

class B(Root):
    def fooBar(self): # TypeError
        pass
```

ăĭĬăyžăZŧénŸçžğăŇăôđçŤĭçŽĎăĬŇăŖĭijŇăyŇéĭăĖĬĬăyÄäyĭäĖĈçşziijŇăôĈçŤĭăĭēæĈĂæĭŇéĖĖĭĭ

```
from inspect import signature
import logging

class MatchSignaturesMeta(type):

    def __init__(self, clsname, bases, clsdict):
        super().__init__(clsname, bases, clsdict)
        sup = super(self, self)
        for name, value in clsdict.items():
            if name.startswith('_') or not callable(value):
                continue
            # Get the previous definition (if any) and compare the_
            ↪signatures
            prev_dfn = getattr(sup, name, None)
            if prev_dfn:
                prev_sig = signature(prev_dfn)
                val_sig = signature(value)
                if prev_sig != val_sig:
                    logging.warning('Signature mismatch in %s. %s !
                    ↪= %s',
                                value.__qualname__, prev_sig,
                    ↪val_sig)

# Example
class Root(metaclass=MatchSignaturesMeta):
    pass
```

```

class A(Root):
    def foo(self, x, y):
        pass

    def spam(self, x, *, z):
        pass

# Class with redefined methods, but slightly different signatures
class B(A):
    def foo(self, a, b):
        pass

    def spam(self, x, z):
        pass

```

æĈædIJä;æĤRëaÑëĤZæŏtăzĉăAriiÑărsăijŽă;ŮăLrăyÑéİcëĤZæăuĉŽDë;ŠăĠžĉzŠædIJrijŽ

```

WARNING:root:Signature mismatch in B.spam. (self, x, *, z) != (self,
→ x, z)
WARNING:root:Signature mismatch in B.foo. (self, x, y) != (self, a,
→ b)

```

èĤŽġë■ēāŚĹăĤæĤrărzăŌæ■TëŌüăyĂăžŽă;ŏæŽĉŽDĉĹNăžRbugæYřă;ĹæIJĹĉTĹĉŽDăĂĈă;NăĉCrij  
éĈăžĹă;Šă■RĉszæTžăRŸăRĈæTřăR■ăŮĉŽDæŮăăĂžărsăijŽërĈĉTĹăĠžëTŽăĂĈ

## ëŏİëŏž

ăIJĹăđ'ġăđÑéİcăRŚăržësăĉŽDĉĹNăžRăy■rijNéĂžăyyărĤĉszĉŽDăŏŽăžĹæTĹăIJĹăĤĈĉszăy■æŌġăĹŮæYřă  
ăĤĈĉszăRřăžëĉŽŚăŌġĉszĉŽDăŏŽăžĹrijNë■ēāŚĹĉijŮĉĹNăžžăŚŸæŠřăžŽæšăæIJĹæšĹăĎRăĹrĉŽDăRřëĈ;ăĠ

æIJĹăžžăRřëĈ;ăijŽër'rijNăĈRëĤZæăuĉŽDëTŽërrăRřăžëéĂžëĤĠNăžRăĹĤæđRăăüăăĤŮăĹŮİDEăŌžă  
ăĤæYřrijNăĈædIJä;ăăIJĹăđDăžžăyĂăyĹæăĤæđŮăĹŮăĠ;æTřăžŠă;ŽăĤŮăžŮăžžă;ĤĉTĹrijNëĈăžĹă;ăæšăăĹ  
ăŽăæ■đ'rijNăřăžăŌëĤŽġ■ĉszăđNĉŽDĉĹNăžRrijNăĈædIJăRřăžăăIJĹăĤĈĉszăy■ăĂžăæĈăĤNăĹŮëŏyăRřăžë

ăIJĹăĤĈĉszăy■ēĂĹæŤ'ëĠæŮřăŏŽăžĹ \_\_\_\_\_new\_\_() æŮžæšTëĤŸæYř  
\_\_init\_\_() æŮžæšTăRŮăĤşăžŌă;ăæĈşæĂŌæăüă;ĤĉTĹĉzŠædIJĉszăĂĈ \_\_\_\_\_new\_\_()  
æŮžæšTăIJĹĉszăĹŽăžžăžNăĹ■ëĉnërĈĉTĹrijNéĂžăyĉTĹăžŌéĂžëĤĠNăžRăĹĤæđRăăüăăĤŮăĹŮİDEăŌžă  
èĂŤ\_\_init\_\_() æŮžæšTăYřăIJĹĉszëĉnăĹŽăžžăžNăRŌëĉnërĈĉTĹrijNă;Šă;ăēIJăĤëĂăŏNăT'æđDăžžĉsză  
ăIJăIJăĂŖŌăyĂăyĹă;Nă■Răy■rijNëĤZæYřăĤĤëĤĤŽDrijNăŽăăyžăŏĈă;ĤĉTĹăžĤ super()  
ăĠ;æTřăĤăRĹĤ'ĉăžNăĹ■ĉŽDăŏŽăžĹăĂĈăŏĈăRĹëĈ;ăIJĹĉszĉŽDăŏđă;NëĉnăĹŽăžžăžNăRŌrijNăžŮăyTĉŽ

æIJăĂŖŌăyĂăyĹă;Nă■RëĤŸăijTĉđ'žăžĤPythonĉŽDăĠ;æTřĉ■ăŖăăržësăĉŽDă;ĤĉTĹăĂĈ  
ăŏđéŽĤăyĹrijNăĤĈĉszăřĤæRăyĹăRřërĈĉTĹăŏŽăžĹæTĹăIJăyĂăyĹĉszăy■rijNăRĹĤ'ĉăĹ■ăyĂăyĹăŏŽăžĹrijĹ  
ĉĎŮăRŌéĂžëĤĠĤ inspect.signature() æİëĉŏĂă■TĉŽDăřTë;ĈăŏĈăžĉĉŽDërĈĉTĹ■ăŖăĂĈ

æIJăĂŖŌăyĂĉĈrijNăžĉăĂăy■æIJĹăyĂăăNă;ĤĉTĹăžĤ super(self, self)  
ăžŮăy■æYřăŌŞĹĹëTŽërrăĂĈă;Šă;ĤĉTĹăĤĈĉszĉŽDæŮăăĂžrijNăĹSăžĉnëăĂăŮăĹLžëŏră;RăyĂĉĈăřşæY  
selfăŏđéŽĤăyĹæYřăyĂăyĹĉszăřžësăăĂĈăŽăæ■đ'rijNëĤZæİăĤăŖăăŮăŏđăřşæYřĉTĹăİăăržăĹă;ăăžŌĉz  
selfĉĹŮĉszĉŽDăŏŽăžĹăĂĈ

## 11.18 9.18 äžëçijŮćíNæŮzaijŘăŮŽăzL'çśž

### éŮőécŸ

ä;ääIJlăEZăyĂæőtäžččăArijŇæIJĂçzLéIJĂèĕAăLŽăžžăyĂăyłæŮřçŽĎçśžăržèśăăĂĆă;ăĕĂĈèŽŚărĖçśžçž  
ăžŮăyŤă;łçŤlăĜ;æŤřærŤăĈ exec() ælĕæL'ğĕăŇăőĈijŇă;ĖæŸřă;ăæĈşăržæL'ăyĂăyłæŽt'ăLăaijŸéŽĖçŽ

### èğĉăĖşæŮzæąŁ

ä;ăăŖřăžĕă;łçŤlăĜ;æŤř types.new\_class() ælĕăLlăğŇăŇŮæŮřçŽĎçśžăržèśăăĂĆ  
ä;ăĕIJĂèĕAăAŽçŽĎăŖłæŸřærŖă;ŽçśžçŽĎăŖă■ăŮăĂAçŁŮçśžăĖĈçžĎăĂăĖşĕŤőă■ŮăŖĆæŤřijŇăžĕăŖŁ

```
# stock.py
# Example of making a class manually from parts

# Methods
def __init__(self, name, shares, price):
    self.name = name
    self.shares = shares
    self.price = price
def cost(self):
    return self.shares * self.price

cls_dict = {
    '__init__': __init__,
    'cost': cost,
}

# Make a class
import types

Stock = types.new_class('Stock', (), {}, lambda ns: ns.update(cls_dict))
Stock.__module__ = __name__
```

èłŽçğ■æŮzaijŘaijŽăđDăžžăyĂăyłæŽőĕĂŽçŽĎçśžăržèśăăijŇăžŮăyŤæŇL'çĖğă;ăçŽĎăIJşæIJŽăŮĕă;IJi

```
>>> s = Stock('ACME', 50, 91.1)
>>> s
<stock.Stock object at 0x1006a9b10>
>>> s.cost()
4555.0
>>>
```

èłŽçğ■æŮzæşŤăy■ijŇăyĂăyłærŤĕ;ĈĕŽ;çŖĖĖğĉçŽĎăIJřæŮzæŸřăIJlĕřĈçŤlăŮŇ  
types.new\_class()ărž Stock.\_\_module\_\_ çŽĎĕŤŇăĀijăĂĆ  
ærŖăĕăă;ŞăyĂăyłçśžĕĉăăŮŽăzL'ăŖŮijŇăăĈçŽĎ \_\_module\_\_  
ăśđăĂğăŇĖăŖăăŮŽăzL'ăăĈçŽĎăłăłŮăŖă■ăĂĆ èłŽăyłăŖă■ăŮçŤlăžŮçŤşăŁŖ

```
>>> import abc
>>> Stock = types.new_class('Stock', (), {'metaclass': abc.ABCMeta},
...                             lambda ns: ns.update(cls_dict))
...
>>> Stock.__module__ = __name__
>>> Stock
<class '__main__.Stock'>
>>> type(Stock)
<class 'abc.ABCMeta'>
>>>
```

```
class Spam(Base, debug=True, typecheck=False):
    pass
```

```
Spam = types.new_class('Spam', (Base,),
                       {'debug': True, 'typecheck': False},
                       lambda ns: ns.update(cls_dict))
```

èóìèőž

```
>>> Stock = collections.namedtuple('Stock', ['name', 'shares',
↪ 'price'])
>>> Stock
<class '__main__.Stock'>
>>>
```

namedtuple() ä¡çŦl exec() èĂÑäy■æŸräyŁéícăžŦçz■çŽĐæŁĂæIJřăĂĆă;EęŸriiŹNäyNéÍcéĂŽè  
æŁŚăžŋçŽt'æŦŦăŁZăžzăyĂăylçśziiŹ

```

import operator
import types
import sys

def named_tuple(classname, fieldnames):
    # Populate a dictionary of field property accessors
    cls_dict = { name: property(operator.itemgetter(n))
                  for n, name in enumerate(fieldnames) }

    # Make a __new__ function and add to the class dict
    def __new__(cls, *args):
        if len(args) != len(fieldnames):
            raise TypeError('Expected {} arguments'.
format(len(fieldnames)))
        return tuple.__new__(cls, args)

    cls_dict['__new__'] = __new__

    # Make the class
    cls = types.new_class(classname, (tuple,), {},
                          lambda ns: ns.update(cls_dict))

    # Set the module to that of the caller
    cls.__module__ = sys._getframe(1).f_globals['__name__']
    return cls

```

æŹæŵtazççäAçŽDæIJÄäRÖéČlälEä;ŹçŤlāžEäyÄäylæL'ÄërŞçŽDäÄlææEæđué■ŤæşŤäÄliijNéĂŽeŹGē  
 sys.\_getframe()  
 æİēēŬäRŪërČçŤİēÄĖçŽDæİäİŬäR■āĂĆ  
 āŘead'ŬäyÄäylææEæđué■ŤæşŤä;Ŋā■ŘāIJl2.15ārRēLĆäy■æIJL'āzŊçz■ēŹGāĂĆ  
 äyNéİčçŽDä;Ŋā■ŘæijŤçd'žāžEāL'■ēİčçŽDäzççäAæŸřæĆä;Ťäüēä;IJçŽDriijŽ

```

>>> Point = named_tuple('Point', ['x', 'y'])
>>> Point
<class '__main__.Point'>
>>> p = Point(4, 5)
>>> len(p)
2
>>> p.x
4
>>> p.y
5
>>> p.x = 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>> print('%s %s' % p)
4 5
>>>

```

èŹZéazæLÄæIJřäyÄäylä;ŹēG■ēçAçŽDæŬzéİcæŸřæŮČärzäžŬäĖČçşzçŽDæ■ççäŵä;ŹçŤlāĂĆ

äjäãRřëČĭăČRÉĂŽëfGçŽt' æŎëăöďăĭNăŇŮăyĂăyĭăĚČçşzæĭëçŽt' æŎëăĹZăžžăyĂăyĭçşzĭijŽ

```
Stock = type('Stock', (), cls_dict)
```

èĚŽçg■æŮzæşŤçŽĐëŮöécŸăĬĭăžŎăŏČăĚĭçŤëăžĒăyĂăžŽăĚşéŤŏæ■éēĭd'ĭijNăŕŤăĉCăŕzăžŎăĚČçşzăy■  
\_\_prepare\_\_() æŮzæşŤçŽĐëŤČŤĭăĂČ éĂŽëfGăĭçŤĭ types.new\_class()  
ĭijNăĭăăRřăžëăĬëŕĂæĹ'ĂæĬĬçŽĐăĚĚëĀăĹĭăğNăŇŮæ■éēĭd' éČĭëČĭăŮăĹŕăĹ'ğëăNăĂČ  
ăŕŤăĉČĭijNtypes.new\_class() çňňăŽZăyĭăŕČæŤŕçŽĐăŽđëŕČăĜĭæŤŕæŎëăŕŮ  
\_\_prepare\_\_() æŮzæşŤëĚŤăŽđçŽĐæŸăăŕĐăŕžëşăĂČ  
ăĉČăđĬăĭăăžĚăžĒăŕĭæŸŕăČşæĹ'ğëăNăŖĒăđ' Ĝæ■éēĭd'ĭijNăŕŕăžëăĭçŤĭ types.  
prepare\_class() äĂČăĭNăĉČĭijŽ

```
import types
metaclass, kwargs, ns = types.prepare_class('Stock', (), {'metaclass
↳': type})
```

ăŏČăĭjŽæşëæĹ'ăŕĹéĂČçŽĐăĚČçşzăžŭëŕČŤĭăŏČçŽĐ \_\_prepare\_\_()  
æŮzæşŤăĂČ çĐŭăŕŎëfZăyĭăĚČçşzăĬă■ŸăŏČçŽĐăĚşéŤŏă■ŮăŕČæŤŕĭijNăĜĒăđ' ĜăŞĭăŕ■çĭ'žéŮŕ'ăŕŎëćŕ  
æŽt'ăđ'ŽăĤăĀŕ, ěŕŭăŕČëĂČ PEP 3115 , äžëăŕĹ Python documentation .

## 11.19 9.19 äĬĭăŏŽăžĹçŽĐæŮŭăĂŽăĹĭăğNăŇŮçşzçŽĐæĹŕăŞŸ

### éŮöécŸ

äjäæČşăĬĬçşzëćŕăŏŽăžĹçŽĐæŮŭăĂŽăŕşăĹĭăğNăŇŮăyĂéČĭăĹĚçşzçŽĐæĹŕăŞŸĭijNëĂNăy■æŸŕëĀç

### èğčăĒşæŮzæăĹ

ăĬĬçşzăŏŽăžĹæŮŭăŕşăĹ'ğëăNăĹĭăğNăŇŮăĹŮëŏĭçĭŏæŞ■ăĭĬăŸŕăĚČçşzçŽĐăyĂăyĭăĚyăđNăžŤçŤĭăĬ  
èĚŽæŮŭăĂŽăĭăăŕŕăžëæĹ'ğëăNăyĂăžŽéćĭăđ' ŮçŽĐæŞ■ăĭĬăĂČ

ăyNëĬăĒŸŕăyĂăyĭăĭNă■ŕĭijNăĹ'çŤĭëfZăyĭăĬëŭăĬëăĹZăžžçşzăĭijăžŎ  
collections æĭăăĬŮăy■çŽĐăŞĭăŕ■ăĚČçžĐçŽĐçşzĭijŽ

```
import operator

class StructTupleMeta(type):
    def __init__(cls, *args, **kwargs):
        super().__init__(*args, **kwargs)
        for n, name in enumerate(cls._fields):
            setattr(cls, name, property(operator.itemgetter(n)))

class StructTuple(tuple, metaclass=StructTupleMeta):
    _fields = []
    def __new__(cls, *args):
        if len(args) != len(cls._fields):
```

```
        raise ValueError('{} arguments required'.format(len(cls.
↪ _fields)))
    return super().__new__(cls, args)
```

èŁŻæŁłāzččăĀăŔŕāzēçŦīāēĭăŏŽāzĹčŏĀăŦčŽĎăšžāžŎăĚĈçzĎçŽĎăŦŕăŏçzŠăđĎŕijŇăēĆăyŇăĹĂç

```
class Stock(StructTuple):
    _fields = ['name', 'shares', 'price']

class Point(StructTuple):
    _fields = ['x', 'y']
```

äyŇéĭcäijŦçđ'žăŏĈăēĆăŦăŭēăĭĬijŽ

```
>>> s = Stock('ACME', 50, 91.1)
>>> s
('ACME', 50, 91.1)
>>> s[0]
'ACME'
>>> s.name
'ACME'
>>> s.shares * s.price
4555.0
>>> s.shares = 23
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: can't set attribute
>>>
```

## èŏĭèŏž

èŁŻăyĀăŕŔēĹĆăyŇijŇçšž StructTupleMeta èŎŭăŔŦŭăĹŕçšžăšđăĀğ \_fields  
äyŇçŽĎăšđăĀğăŔăăŭăĹŦēăĭijŇçĎŭăŔŎăŕĒăŏĈăzŇēĭŇăŇcăĹŕçŽyăžŦçŽĎăŕfēŏēŭŏçĹ'žăŏŽăĚĈçzĎăç  
operator.itemgetter() âĹŽăžžăyĀăyĭŏēŕŕŭŏăŽĭăĜĭăŦŕijŇçĎŭăŔŎ property()  
ăĜĭăŦŕăŕĒăĚŭēĭŇăŇcăĹŕăyĀăyĭăšđăĀğăĂĈ

ăĬŇŇēĹĆăĬĬĂēŽĭăĜĈçŽĎēĈĭăĹĒăŦŕçšēēĀšăyŇăŕŇçŽĎăĹĭăğŇăŇŦŭăŇēĭđ'ăŦŕăzĂăžĹăŦŭăĂŽăŕŔ  
StructTupleMeta äyŇçŽĎ \_\_init\_\_() äŦžăşŦăŔĭăĬĭăŕŔăyĭçšžēĭŇăŏŽăžĹăŦŭēĭŇēŕĈçŦĭăyĀăŇăă  
cls âŔĈăŦŕăŕŝăŦŕēĈăyĭŇēĭŇăŏŽăžĹçŽĎçšžăĂĈăŏđēŽĚăyĹijŇăyĹēŕăzččăĀăĭçŦĭăžĒ  
\_fields çšžăŔŦŕēĜŔăĭăăĹăŦŦăŦŕçŽĎēĭŇăŏŽăžĹçŽĎçšžijŇ  
çĎŭăŔŎçzŽăŏĈăĒăŭăăĹăăyĂçĈçăŦŕçŽĎăyĬJēēăĂĈ

StructTuple çšžăĬĬăyžăyĀăyĭăŽŏēĂŽçŽĎăšžçšžijŇăŦŽăŦŭăžŦŭăĭçŦĭēĂĚăĭēçžğăĹŕăĂĈ  
èŁŻăyĭçšžăyŇçŽĎ \_\_new\_\_() äŦžăşŦçŦĭăĭăđĎēĂăăŦŕçŽĎăŏđăŦăĂĈ èŁŽēĜŇăĭçŦĭ  
\_\_new\_\_() âžŭăyŇăŦŕăĭĹăyŕēğĀijŇăyžēēĀăŦŕăŽăăyžăĹŝăžŇēēĀăŕŏăŦžăĚĈçzĎçŽĎēŕĈçŦĭçăŕŇijŇ  
ăĭăăŦŕăŝăŦŕăžăăĈŔăŽŏēĂŽçŽĎăŏđăŦŕçŦĭēĈçăŭăăĹăžžăŏđăŦăĂĈăŕŝăĈŔăyŇéĭcēŁŻăăŭijŽ

```
s = Stock('ACME', 50, 91.1) # OK
s = Stock(('ACME', 50, 91.1)) # Error
```



```
# multiiple.py
import inspect
import types

class MultiMethod:
    '''
    Represents a single multimethod.
    '''
    def __init__(self, name):
        self._methods = {}
        self.__name__ = name
```

```

def register(self, meth):
    '''
    Register a new method as a multimethod
    '''
    sig = inspect.signature(meth)

    # Build a type signature from the method's annotations
    types = []
    for name, parm in sig.parameters.items():
        if name == 'self':
            continue
        if parm.annotation is inspect.Parameter.empty:
            raise TypeError(
                'Argument {} must be annotated with a type'.
↪format(name)
            )
        if not isinstance(parm.annotation, type):
            raise TypeError(
                'Argument {} annotation must be a type'.
↪format(name)
            )
        if parm.default is not inspect.Parameter.empty:
            self._methods[tuple(types)] = meth
        types.append(parm.annotation)

    self._methods[tuple(types)] = meth

def __call__(self, *args):
    '''
    Call a method based on type signature of the arguments
    '''
    types = tuple(type(arg) for arg in args[1:])
    meth = self._methods.get(types, None)
    if meth:
        return meth(*args)
    else:
        raise TypeError('No matching method for types {}'.
↪format(types))

def __get__(self, instance, cls):
    '''
    Descriptor method needed to make calls work in a class
    '''
    if instance is not None:
        return types.MethodType(self, instance)
    else:
        return self

class MultiDict(dict):
    '''

```

```

Special dictionary to build multimethods in a metaclass
'''
def __setitem__(self, key, value):
    if key in self:
        # If key already exists, it must be a multimethod or
        ↪ callable
        current_value = self[key]
        if isinstance(current_value, MultiMethod):
            current_value.register(value)
        else:
            mvalue = MultiMethod(key)
            mvalue.register(current_value)
            mvalue.register(value)
            super().__setitem__(key, mvalue)
    else:
        super().__setitem__(key, value)

class MultipleMeta(type):
    '''
    Metaclass that allows multiple dispatch of methods
    '''
    def __new__(cls, clsname, bases, clsdict):
        return type.__new__(cls, clsname, bases, dict(clsdict))

    @classmethod
    def __prepare__(cls, clsname, bases):
        return MultiDict()

```

äyžāzĖā;fcŦlĕfZāyŦçszñijŇā;āāŦřāzĕāČŦřāyŇéŦcĕfZæāũāEŽñijŽ

```

class Spam(metaclass=MultipleMeta):
    def bar(self, x:int, y:int):
        print('Bar 1:', x, y)

    def bar(self, s:str, n:int = 0):
        print('Bar 2:', s, n)

# Example: overloaded __init__
import time

class Date(metaclass=MultipleMeta):
    def __init__(self, year: int, month:int, day:int):
        self.year = year
        self.month = month
        self.day = day

    def __init__(self):
        t = time.localtime()
        self.__init__(t.tm_year, t.tm_mon, t.tm_mday)

```

äyŇéŦcæŸřāyĀäyŦāzđ'āžŠçđ'žā;ŇæŦĕĕŦŇĕřĀăŦČĕČ;æ■čçāŵçŽĐāũĕä;IJñijŽ

```

>>> s = Spam()
>>> s.bar(2, 3)
Bar 1: 2 3
>>> s.bar('hello')
Bar 2: hello 0
>>> s.bar('hello', 5)
Bar 2: hello 5
>>> s.bar(2, 'hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "multiple.py", line 42, in __call__
    raise TypeError('No matching method for types {}'.
↪format(types))
TypeError: No matching method for types (<class 'int'>, <class 'str'
↪'>)
>>> # Overloaded __init__
>>> d = Date(2012, 12, 21)
>>> # Get today's date
>>> e = Date()
>>> e.year
2012
>>> e.month
12
>>> e.day
3
>>>

```

## èõìèõž

àìççŽ;æìèèõřijŇçŽýàrzäžŎéĀŽäyÿçŽĎäzççäAèĀŇâušæIJñèŁĆä;ŁçŤlĀŁřäžEā;Ĺāđ'ŽçŽĎé■ŤæšŤäzçç;
 ä;EæŸřijŇāōČā■'èČ;èōl'æĹŚāznæuśāĒēçŘĒçġcāĒČçšzāŠŇæĀŔēřāŽlçŽĎāžŤāsČāuēä;IJāŎšçĒEřijŇ
 āžūēČ;āĹāæuśāržèŁŽāžZæçČāŁççŽĎā■řèsaāĀČāŽāæ■d'ijŇāršçōŮä;āāžūäy■āijŽçñŇā■šāŎžāžŤçŤlæIJñèŁĆ
 āōČçŽĎäyĀāžŽāžŤāsČæĀĬæČšā■'āijŽā;śāš■āĹŔāĒūāōČæūĹ'āŔĹāĹŔāĒČçšzāĀAæĀŔēřāŽlāŠŇāĠ;æŤřæš

æIJñèŁĆçŽĎāōđčŎřäy■çŽĎäyžèèAæĀĬèuŕāĒūāōđæŸŕā;ĹçōĀā■ŤçŽĎāĀĆMutipleMeta
 āĒČçšzā;ŁçŤlāōČçŽĎ \_\_prepare\_\_() æŮžæšŤ æĬæĀŔĀ;ŽäyĀäyĹā;IJäyž MultiDict
 āōđä;ŇçŽĎēĠāōŽāžĹā■ŮāĒyāĀČēŁŽäyĹēušæŽōéĀŽā■ŮāĒyāy■äyĀæāuçŽĎæŸřijŇ
 MultiDict āijŽāIJlāĒČçŤ'æēčñèō;ç;ōçŽĎæŮūāĀŽæčĀæšēæŸŕāŔēāušçžŔā■ŸāIJřijŇāēČæđIJā■ŸāIJlçŽĎ
 MultiMethod āōđä;Ňäy■āŔĹāžūāĀĆ

MultiMethod āōđä;ŇéĀŽēŁĠæđĎāžžāžŎçšzādŇç■;āŔ■āĹŔāĠ;æŤřçŽĎæŸāārĎæĬæŤūéZEæŮžæšŤ
 āIJlēŁŽäyĹæđĎāžžēŁĠŇäy■ijŇāĠ;æŤřæšĹēçčēčñçŤlæĬæŤūéZEēŁŽāžŽç■;āŔ■çĎūāŔŎæđĎāžžēŁŽäyĹæŸ
 ēŁŽäyĹēŁĠŇāIJĹ MultiMethod.register() æŮžæšŤäy■āōđčŎřāĀĆ
 ēŁŽçġ■æŸāārĎçŽĎäyĀäyĹāĒšēŤŏçĹ'žçČzæŸŕāržāžŎāđ'ŽäyĹæŮžæšŤřijŇæĹ'ĀæIJĹ'āŔČæŤřçšzādŇēČ;āŁĒē

äyžāžEèōl' MultiMethod āōđä;ŇælāæŇšäyĀäyĹēŔČçŤlřijŇāōČçŽĎ
 \_\_call\_\_() æŮžæšŤēčñāōđčŎřāžEāĀĆ ēŁŽäyĹæŮžæšŤäžŎæĹ'ĀæIJĹ'æŎŠēŽd' slef
 çŽĎāŔČæŤřäy■æđĎāžžäyĀäyĹçšzādŇāĒČçžĎřijŇāIJlāĒEĒēĆlmapäy■æšēæĹ;ēŁŽäyĹæŮžæšŤřijŇ
 çĎūāŔŎērČçŤlçŽyāžŤçŽĎæŮžæšŤāĀČäyžāžEēČ;èōl' MultiMethod

áoďäĭŇáIJłçśzáoŽžŁ'æŮŮæ■čçaőæ\$■äĭIJiĭŇ\_\_get\_\_() æŸřăĤĚéazăĭŮăőđçŎřçŽďăĂĆ  
áoČěćŋĤlăĭăđĐăžžæ■čçaőçŽďçzŚăőŽæŮžæşŤăĂĆăřŤăęĆĭĭŽ

```
>>> b = s.bar
>>> b
<bound method Spam.bar of <__main__.Spam object at 0x1006a46d0>>
>>> b.__self__
<__main__.Spam object at 0x1006a46d0>
>>> b.__func__
<__main__.MultiMethod object at 0x1006a4d50>
>>> b(2, 3)
Bar 1: 2 3
>>> b('hello')
Bar 2: hello 0
>>>
```

äÿ■ēĤGăIJñēŁĆçŽďăőđçŎřēĤŸæIJŁ'äÿĂăžŽēŽŖăĹŭĭĭŇăŸŮăÿ■äÿĂăÿĤæŸřăőČăÿ■ēČĭăĤçŤlăĚşēŤőă■

```
>>> s.bar(x=2, y=3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: __call__() got an unexpected keyword argument 'y'

>>> s.bar(s='hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: __call__() got an unexpected keyword argument 's'
>>>
```

ăžşëöÿæIJŁ'ăĚŮăžŮçŽďæŮžæşŤēČĭæŭžăĹăēĤŽçğ■æŤŕæŇăĭĭŇăĭĤæŸřăőČēIJĂēęĂăÿĂăÿĤăőŇăĚĭăÿ■  
éŮőéćŸăĭĭăžŎăĚşēŤőă■ŮăŖĆæŤŕçŽďăĢçŎŖæŸŕæşqæIJŁ'éąžăžŖçŽďăĂĆăĭŞăőČëŭşăĭ■çĭőăŖĆæŤŕæŭŭăĭ  
éĆčăĭçŽďăŖĆæŤŕăŕşăĭĤăŖŸăĭŮăŕŤēĭČæŭŭăžşăžĤĭĭŇēĤŽæŮŮăĂŽăĭăÿ■ăĭŮăÿ■ăĭĭ  
\_\_call\_\_() æŮžæşŤăÿ■ăĚĹăŎžăĂžăÿĤæŎşăžŖăĂĆ

ăŖŇăăŭăŕžăžŎçžğæŁĤăžşæŸŕæIJŁ'éŽŖăĹŭçŽďĭĭŇăĭŇăęĆĭĭŇçşzăĭĭĭăÿŇēĭĤēĤŽçğ■ăžççăĂăŕşăÿ■ēČĭ

```
class A:
    pass

class B(A):
    pass

class C:
    pass

class Spam(metaclass=MultipleMeta):
    def foo(self, x:A):
        print('Foo 1:', x)

    def foo(self, x:C):
        print('Foo 2:', x)
```

ǎŎŖšǎŽǎæŸřǎŽǎäŸž x : A æşlèğçäŸ■èĈ;æŁŖǎŁšǎŇzeĚ■ǎ■Ŗçşzǎǒđǎ;ŇiijŁæŖŤǎęĈBęŻĐǎǒđǎ;ŇiijŁ'iiŇŃǎ

```
>>> s = Spam()
>>> a = A()
>>> s.foo(a)
Foo 1: <__main__.A object at 0x1006a5310>
>>> c = C()
>>> s.foo(c)
Foo 2: <__main__.C object at 0x1007a1910>
>>> b = B()
>>> s.foo(b)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "multiple.py", line 44, in __call__
    raise TypeError('No matching method for types {}'.
↪format(types))
TypeError: No matching method for types (<class '__main__.B'>,)
>>>
```

ä;IäŸžǎ;řçŤlǎĚĈçşzǎŖŇæşlèğççŻĐäŸÄçğ■æŽfǎžçæŰzæǎŁiijŇǎŖřǎžééǺŽęŁĜæŖŖęŖǎŽlǎĚǎǒđçŎŖç

```
import types

class multimethod:
    def __init__(self, func):
        self._methods = {}
        self.__name__ = func.__name__
        self._default = func

    def match(self, *types):
        def register(func):
            ndefaults = len(func.__defaults__) if func.__defaults__
↪else 0
            for n in range(ndefaults+1):
                self._methods[types[:len(types) - n]] = func
            return self
        return register

    def __call__(self, *args):
        types = tuple(type(arg) for arg in args[1:])
        meth = self._methods.get(types, None)
        if meth:
            return meth(*args)
        else:
            return self._default(*args)

    def __get__(self, instance, cls):
        if instance is not None:
            return types.MethodType(self, instance)
        else:
            return self
```

äyžäZĖä;ŁçŁłæRRĖŁřäZłŁŁæIJñijNä;ăĖIJĂĕĖAăČŘäyNéİċĕŁZăăŭăĖZiijŽ

```
class Spam:
    @multimethod
    def bar(self, *args):
        # Default method called if no match
        raise TypeError('No matching method for bar')

    @bar.match(int, int)
    def bar(self, x, y):
        print('Bar 1:', x, y)

    @bar.match(str, int)
    def bar(self, s, n = 0):
        print('Bar 2:', s, n)
```

æRRĖŁřäZłŁæŰzăăŁăRŃăăŭăžšæIJL'ăL'■ĖłċæRRăŁřçŽĐĖŽŘăŁŰiijLăy■æŁřăNĂăĖšĖŁŕă■ŰăRĆăŁřăš

æL'ĂæIJL'ăžNċL'ĖĈ;æŸřăžšç■L'çŽĐřijNæIJL'ăĕ;æIJL'ăĪřijNăžšĖĕŷæIJĂăĕ;çŽĐăŁđăşŁăřsăŸřăĪJăŁ

ăy■ĖĖŁæIJL'ăžZċL'zăĕŁăĈĖăĖĭăyNĖŁŸăŸřăIJL'ăĐŘăžL'çŽĐřijNăřŁăĈăšžăžŎăĭăĭjRăNzéĖ■çŽĐăŰžă

ăyŁăyŁă;Nă■RiijN8.21ăřRĖŁĈăy■çŽĐĖĕĖĖŎĖĂĖăĭăĭjRăRřăžăăĖĕăŁăŰžăyžăyĂăyŁă;ŁçŁłæŰzăşŁĖĖ■Ė;ŁçŽ

ă;ĖăŸřijNĖŽđ'ăžĖĖŁăyŁăžĖăđ'ŰiijNĖĂŽăyŷăy■ăžŁĖĖă;ŁçŁłæŰzăşŁĖĖ■Ė;ŁiijLăřšċŎĂă■ŁçŽĐă;ŁçŁłăy■ă

ăĪĪPythonċđ'ŁăŃžăřăžăŎăĕđĈŎăŰzăşŁĖĖ■Ė;ŁçŽĐĖĕĖĕžăŭşçŁçŁłăĖăŭşăžĖăĂĈ

ăřăžăŎăĭjŁăŘšĖŁăyŁăžL'ĖĕžçŽĐăŎšăŽăiijNăRřăžăăRĈĖĂĈăyNĖGuido van

RossumçŽĐĖŁçŁłă■ŽăĕċiijŽ [Five-Minute Multimethods in Python](#)

## 11.21 9.21 éAŁăĖ■ĖĖăđ'■çŽĐăśđăĂĖăŰzăşŁ

éŰĕĖŸ

ă;ăăĪĪċşşăy■ĖIJĂĕĖAĖĖăđ'■çŽĐăŕăžăžL'ăyĂăžZăL'ĖĕăNċŽŷăRŃĖĂžĖ;ŚçŽĐăśđăĂĖăŰzăşŁiijNăřŁ

ĖĖĈăĖşşăŰzăăŁ

ĖĂĈĖŽŚăyNăyĂăyŁċŎĂă■ŁçŽĐċşşiijNăŕĈçŽĐăśđăĂĖĈŁśăśđăĂĖăŰzăşŁăNĖĖĖiijŽ

```
class Person:
    def __init__(self, name ,age):
        self.name = name
        self.age = age

    @property
    def name(self):
        return self._name

    @name.setter
    def name(self, value):
        if not isinstance(value, str):
```

```
        raise TypeError('name must be a string')
    self._name = value

    @property
    def age(self):
        return self._age

    @age.setter
    def age(self, value):
        if not isinstance(value, int):
            raise TypeError('age must be an int')
        self._age = value
```

ãŖŕäzëçIJŇăĹŕiijŇäyžăžĚăőđçŎŕăşđæĂğăĂijçŽĎçşzăđŇæčĂæşëæĹŚăznăĚŽăžĚăĹăđ'ŽçŽĎéĜăđ'ăăŕlëçAăjăăžëăŖŎçIJŇăĹŕçşzăijijëĚŽăăüçŽĎăžçăĂiijŇăjăăéĈjăžŤerëæĈşăĹđæşŤăŎžçóĂăŇŮăőĈăĂĈăyĂăyĹăŖŕëăŇçŽĎăŮzæşŤăŸŕăĹŽăžžăyĂăyĹăĜjăŤŕçŤĹăĹăőŽăžĹăśđæĂğăžüëĚŤăŽđăőĈăĂĈăĹŇăçĈiijŽ

```
def typed_property(name, expected_type):
    storage_name = '_' + name

    @property
    def prop(self):
        return getattr(self, storage_name)

    @prop.setter
    def prop(self, value):
        if not isinstance(value, expected_type):
            raise TypeError('{} must be a {}'.format(name, expected_
→type))
        setattr(self, storage_name, value)

    return prop

# Example use
class Person:
    name = typed_property('name', str)
    age = typed_property('age', int)

    def __init__(self, name, age):
        self.name = name
        self.age = age
```

## ëőĹëőž

æIJŇëĹĈæĹŚăznăijŤçđ'žăĚĚéĈĹăĜjăŤŕæĹŮëĂĚéŮăăŇĚçŽĎăyĂăyĹéĜăăçAçĹ'žăĂğiijŇăőĈăznăĹăĹă typed\_property() çIJŇăyĹăŎžăIJĹçĈzéŽĹçŖĚëğçiijŇăĚŮăăđăăĈăĹ'ĂăĂŹçŽĎăžĚăžĚăŕşæŸŕăyžăjăçăăŽăăăđ'iijŇăjăŖăIJăyĂăyĹçşzăyăăjăçŤĹăăĈçŽĎăŮăăĂŹiijŇăŤĹăđIJëüşăŕĚăăĈéĜŇéĹççŽĎăžçăĂăŤĹăĹŕăŕjçóăşđæĂğçŽĎ getterăŖŇ setterăŮzæşŤëőĚéŮăăžĚăIJŇăIJŕăŖŸéĜŖăçĈă name , expected\_typeăžëăŖĹă storage\_name iijŇëĚăyĹăĹăăăçăyŷiijŇëĚăžăžăŖŸéĜŖçŽĎăĂijăijŽăĚĹăăŸ



æĹsāzñēŸāRřāzēä;ŁçŦĹ functools.partial() æĹčĹčĹæŦžāRŸäyNēčZäyĹä;NāĹŘĭijNāĹŁæIJĹ

```
from functools import partial

String = partial(typed_property, expected_type=str)
Integer = partial(typed_property, expected_type=int)

# Example:
class Person:
    name = String('name')
    age = Integer('age')

    def __init__(self, name, age):
        self.name = name
        self.age = age
```

ǎĖŭǎōđǎ;ǎǎŔǎzēǎŔŚĆŎŕiĭjNēfZēGŇčŽĎǎzččǎAēŭ§8.13ǎŕRēŁĆäy■čŽĎčśǎđŇčšzčzšǎŔŔēfǎŹlǎzččǎ

11.22 9.22 aóŽázL'äýŁäýNæŨĞçóaçRĖaŽÍçŽĎčóĀa■TæŨzæşT

éŮőécŸ

ä;äæÇşèĞlaúşåŒzaódçŒřäyÄäylæŨřçŽďäyläyNæŨĞçoaçŘĚåŽlíiŋNäzëä;Ěä;ĚçŤlīwīthēř■āŘēāĂĆ

èġčǎẸșæŮźæąŁ

a0dcOraYAAyLaUrcZDyLayNaUGcoacReaZlczDaeIAcooAaTczDaeUzaetarsaeYra;fctI  
 contexlib                      alaaiUayczD                      @contextmanager                      eEeerZlaAC  
 ayNeIcaYrayAAyLa0dcOraZEazccaAaiUeoaaUuaLseC;czDyLayNaUGcoacReaZla;NaRrijZ

```
import time
from contextlib import contextmanager

@contextmanager
def timethis(label):
    start = time.time()
    try:
        yield
    finally:
        end = time.time()
        print('{}: {}'.format(label, end - start))

# Example use
with timethis('counting'):
    n = 10000000
    while n > 0:
        n -= 1
```

```
    aIjIaGjæTřtimethis() äyñijÑyield äzÑaL'■çŽDäzčçäAäijŽaIjIäyLäyNæŮGçõaçŘEāZlāy■ajIäy
__enter__() æŮzæşTjæL'gèaÑñijÑ æL'ĂæIJL'āIJl yield äzÑaRŮçŽDäzčçäAäijŽajIäyž
__exit__() æŮzæşTjæL'gèaÑāĀĆ æÇæđIjāGžçŮřāžEāijČäyñijÑajČäyñijŽaIjIyield-
ér■āRēéĆcéGÑæLŽāGžāĀĆ
```

äyÑéIcæYřäyĂäyIæŽt'āLăénYçžgäyĂçĆzçŽDäyLäyNæŮGçõaçŘEāZlñijÑāōđçŮřāžEāLŮèaIărzèśäyL

```
@contextmanager
def list_transaction(orig_list):
    working = list(orig_list)
    yield working
    orig_list[:] = working
```

èĚŽæōřāzčçäAçŽDä;IjçTlæYřāzžä;TjārzāLŮèaIçŽDäĚōæTžāRlæIJL'ā;ŞæL'ĂæIJL'äzčçäAèĚŘèaÑāōÑæ
äyÑéIcæLŠāznæIæijTçd'žäyĂäyÑñijŽ

```
>>> items = [1, 2, 3]
>>> with list_transaction(items) as working:
...     working.append(4)
...     working.append(5)
...
>>> items
[1, 2, 3, 4, 5]
>>> with list_transaction(items) as working:
...     working.append(6)
...     working.append(7)
...     raise RuntimeError('oops')
...
Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
RuntimeError: oops
>>> items
[1, 2, 3, 4, 5]
>>>
```

## èőIèőŽ

éĀŽäyñæČĚāEřäyÑñijNæÇæđIjèeAāEŽäyĂäyIäyLäyNæŮGçõaçŘEāZlñijNä;ăeIjĀèeAāōŽāzL'äyĂäyIç
\_\_enter\_\_() āŠNäyĂäyI \_\_exit\_\_() æŮzæşTñijNæČäyNæL'Āçd'žñijŽ

```
import time

class timethis:
    def __init__(self, label):
        self.label = label

    def __enter__(self):
        self.start = time.time()

    def __exit__(self, exc_ty, exc_val, exc_tb):
```

```
end = time.time()
print('{{: }}'.format(self.label, end - self.start))
```

är;çøæfZäyläzšäy■ēZ;āEŻiijNä;EæYřçZÿæřTē;ČāEŻäyÄäylçōĀā■TçŽDä;fçTí  
@contextmanager æšlègčçŽDāĠ;æTřèĀNēĀēfYæYřçl■æY;äzRāSšāĀĆ

@contextmanager äžTērēāzĒāzĒçTlāleāEŻēĠāNĒāRñçŽDäyLäyNæŮĠçōaçŘēāĠ;æTřāĀĆ  
æČæđIJā;āæIJL'äyĀāžŽāržèšā(æřTāçČäyÄäylæŮĠäzūāĀAç;ŠçzIJèđæŌēĀĹŮéTĀ)iiijNéIJĀēçAæTřæĀNā  
with ēř■āŘēiiijNéCčāžLā;āāršēIJĀēçAā■TçNñāōđçŎř \_\_enter\_\_() æŮžæšTāŠN  
\_\_exit\_\_() æŮžæšTāĀĆ

## 11.23 9.23 āIJlāsĀēČlāRŸéĠRāššäy■æL'gèāNāzčçāA

### éŮóécŸ

ä;āæČšāIJlā;fçTlēNČāŽt'āEĒæL'gèāNæšŘäyläzčçāAçL'ĠæōřiiijNāžūāyTāyNæIJŽāIJlæL'gèāNāRŎæL'Ā

### èğčāEşæŮzæqĹ

äyžāžEçŘēèğçēfZäylēŮóécŸiiijNāĒLēřTēřTäyÄäylçōĀā■TāIJžæŽřāĀĆéçŮāĒLiiijNāIJlāĒlāsĀāš;āŘ■ç

```
>>> a = 13
>>> exec('b = a + 1')
>>> print(b)
14
>>>
```

çDŮāRŎřiiijNāE■āIJlāyÄäylāĠ;æTřäy■æL'gèāNāRŃNæūçŽDäzčçāAiiijŽ

```
>>> def test():
...     a = 13
...     exec('b = a + 1')
...     print(b)
...
>>> test()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 4, in test
NameError: global name 'b' is not defined
>>>
```

āŘräžēçIJNāĠžiiijNæIJĀāRŎæLŽāĠžāžEäyÄäylNameErrorāijČäyŸiiijNāršēūšāIJl  
exec() ēř■āŘēāzŎæšqæL'gèāNēfĠäyĀæūāĀĆ ēçAæYřā;āæČšāIJlāRŎēlççŽDēōaçōŮäy■ā;fçTlāLř  
exec() æL'gèāNçzšşæđIJçŽDērĹāršāijŽæIJL'éŮóécŸäžEāĀĆ

äyžāžEāfōæ■çēfZæūçŽDēTŽērřiiijNā;āēIJĀēçAāIJlērČçTí exec()  
äžNāL'■ā;fçTí locals() āĠ;æTřāleā;ŮāLřäyÄäylāsĀēČlāRŸéĠRā■ŮāĒyāĀĆ  
äžNāRŎā;āāršēČ;äzŎāsĀēČlā■ŮāĒyāy■ēŎūāRŮāfōæTžēfĠāRŎççŽDāRŸéĠRāĀijāžEāĀĆ;NāēČiiijŽ

```
>>> def test():
...     a = 13
...     loc = locals()
...     exec('b = a + 1')
...     b = loc['b']
...     print(b)
...
>>> test()
14
>>>
```

## ěóľěőž

ăóděŽĚäyŁárzäžŎ exec() çŽDæ■ççaőă;ŁçTlæYřæfTè;ČéŽçŽDăĂĆăd'ğăd'ŽæTřæČĚăĚtăyNă;Šă;ăē  
exec() çŽDæŮŭăĂŽřijN èŁYæIJL'ăRęăd'ŮæŽt'ăē;çŽDèğčăĚşæŮzæąŁřijLæfTăēČēcĚéčřăŽlăĂăĚŮ■ăNĚă

çDűëĂNřijNăēĆăđIJă;ăäz■çDűëĚĂă;ŁçTl exec() řijNăIJnêŁĆăLŮăĠzăžĚăyĂăžZăēĆă;Tăē■ççaőă;Łç  
ézYēōd'æČĚăĚtăyNřijNexec() äijŽăIJlêřČçTlêĂĚăśĂéĆlăŚNăĚlăśĂēNČăŽt'ăĚĚăL'ğēăNăžččăĂăĂĆčDű  
äijăēĂŠçzŽ exec() çŽDăśĂéĆlêNČăŽt'æYřæNűet'ĬăóděŽĚăśĂéĆlăRŸéĠRçzDăĤRçŽDăyĂăyĽă■ŮăĚyăĂ  
ăŽăă■d'řijNăēĆăđIJ exec() âēĆăđIJăL'ğēăNăžĚăĽŏăTzăŞ■ă;IJřijNêŁŽçğ■ăĽŏăTzăŘŎçŽDçzŞăđIJăřzăč  
ăyNēlăēYřăRęăd'ŮăyĂăyĽăijTçd'žăŏČçŽDă;Nă■RřijŽ

```
>>> def test1():
...     x = 0
...     exec('x += 1')
...     print(x)
...
>>> test1()
0
>>>
```

ăyĽēlăčăžččăĂēĠNřijNă;Šă;ăērČçTl locals() èŎŭăRŮăśĂéĆlăRŸéĠRăŮŭřijNă;ăēŎŭă;ŮçŽDæYřăi  
exec() çŽDăśĂéĆlăRŸéĠRçzDăyĂăyĽăNűet'ĬăĂĆ éĂŽēŁĠăIJlăžččăĂăL'ğēăNăŘŎăŏăşēēŁŽăyĽă■ŮăĚyă

```
>>> def test2():
...     x = 0
...     loc = locals()
...     print('before:', loc)
...     exec('x += 1')
...     print('after:', loc)
...     print('x =', x)
...
>>> test2()
before: {'x': 0}
after: {'loc': {...}, 'x': 1}
x = 0
>>>
```

ăžTçzĚğČăřşæIJĂăŘŎăyĂăēçŽDè;ŞăĠžřijNéŽd'ēlđă;ăăřĚ

loc

äy■écñä£öæŤzãRŎçŽDãĀijæL'NãŁlètNãĀijçzŽxiiĴNãRëãŁŽxãRŸéGRãĀijæŸřäy■äijŽãRŸçŽDãĀĆ

āIJlä;£çŤl locals () çŽDæŮüãĀŽiiĴNä;ăéIJăëAæşlæĐRæŞ■ä;IJéažãžRãĀĆæřRæñãóČècñèřČçŤlç;  
locals () äijŽèŎüãRŮásĀéČlãRŸéGRãĀijäy■çŽDãĀijãžüëEçŽŮã■ŮãËyäy■çŽyãžŤçŽDãRŸéGRãĀĆ  
èřüæşlæĐRëğCãřşäyNäyNéIcè£ŽäyŤerŤetNçŽDè;ŞãGžçzŞæđIJiiĴŽ

```
>>> def test3():
...     x = 0
...     loc = locals()
...     print(loc)
...     exec('x += 1')
...     print(loc)
...     locals()
...     print(loc)
...
>>> test3()
{'x': 0}
{'loc': {...}, 'x': 1}
{'loc': {...}, 'x': 0}
>>>
```

æşlæĐRæIJĀãRŎäyĀæñæèřČçŤl locals () çŽDæŮüãĀŽxçŽDãĀijæŸřäçCä;ŤècñèçEçŽŮæŎLçŽDã.

ä;IJäyž locals () çŽDäyĀäyŤæŽ£äzçæŮžæãŁiiĴNä;ăãRřæžä;£çŤlã;ăèGŤãüšçŽDã■ŮãËyiiĴNãžüãřEãó  
exec () āĀĆä;NãçCiiĴŽ

```
>>> def test4():
...     a = 13
...     loc = { 'a' : a }
...     glb = { }
...     exec('b = a + 1', glb, loc)
...     b = loc['b']
...     print(b)
...
>>> test4()
14
>>>
```

ãđ'ğéČlãŁEæČĚãEřäyNiiĴNè£Žçg■æŮžäijRæŸřä;£çŤl exec () çŽDæIJĀä;şãóđèüřãĀĆ  
ä;ăãRŤéIJăëAă£İerAăĒlãşĀãŞNãşĀéČlã■ŮãËyãIJlãRŎİcãžççãAăèð£éŮöæŮüãüšçzRècñãŁIãgNãNŮãĀĆ

è£ŸæIJL'äyĀçCžiiĴNãIJlä;£çŤl exec () äžNãŁ■iiĴNä;ăãRřèČ;éIJăëAéŮöäyNèGŤãüšæŸřãRëæIJL'ãĚ  
ãđ'ğãđ'ŽæŤřæČĚãEřäyNã;Şä;ăëçAăĀČèŽŚä;£çŤl exec () çŽDæŮüãĀŽiiĴN  
è£ŸæIJL'ãRëãđ'ŮæŽŤ'ăë;çŽDèğçãEşæŮžæãŁiiĴNæřŤæçCèçĚéçřãŽlãĀAéŮ■ãNĚãĀAăĒČçşziiĴNæŁŮãĚüãžŮ

## 11.24 9.24 èğçæđŘäyŎãŁEæđŘPythonæžŘçãĀ

### éŮöécŸ

ä;ăæČşãEŽèğçæđŘãžüãŁEæđŘPythonæžŘäzççãĀçŽDçlNãžRãĀĆ

## èġċăEşæŮzæąŁ

ăđ'ġéČlăLEęłŃăžŔăŚŸçşééAşPythonèČ;ăđ'şèőăçőŮăŁŮăL'ġëąŃă■Ůçņęäyşă;ćăijŔçŽĐăžŔăžčăĂă

```
>>> x = 42
>>> eval('2 + 3*4 + x')
56
>>> exec('for i in range(10): print(i)')
0
1
2
3
4
5
6
7
8
9
>>>
```

ăr;çőăăęĆă■đ'iijŃast æłăăIŮèČ;èćŋćŦlălěărEPythonæžŔçăAçijŮërŚăĹŔăyĂăyłăŔfēcŋăĹEăđŔçŽĐă

```
>>> import ast
>>> ex = ast.parse('2 + 3*4 + x', mode='eval')
>>> ex
<_ast.Expression object at 0x1007473d0>
>>> ast.dump(ex)
"Expression(body=BinOp(left=BinOp(left=Num(n=2), op=Add(),
right=BinOp(left=Num(n=3), op=Mult(), right=Num(n=4))), op=Add(),
right=Name(id='x', ctx=Load())))"

>>> top = ast.parse('for i in range(10): print(i)', mode='exec')
>>> top
<_ast.Module object at 0x100747390>
>>> ast.dump(top)
"Module(body=[For(target=Name(id='i', ctx=Store()),
iter=Call(func=Name(id='range', ctx=Load()), args=[Num(n=10)],
keywords=[], starargs=None, kwargs=None),
body=[Expr(value=Call(func=Name(id='print', ctx=Load()),
args=[Name(id='i', ctx=Load())], keywords=[], starargs=None,
kwargs=None)], or_else=[])])"
>>>
```

ăĹEăđŔăžŔçăĂăăŚéIJĂèęĂă;ăëĠlăŮşăŹŦăđ'ŽçŽĐă■ăžăiijŃăőĆăŸŕçŦşăyĂçşzăĹŮASTēĹĆçĆżçŽĐă  
ăĹEăđŔēĲŽăžŽēĹĆçĆżăIJĂçőĂă■ŦçŽĐăŮżæşŦăŕşăŸŕăőŽăžĹ'ăyĂăyłēőĲéŮőēĂĒçşzîijŃăőđçŎŕăĹăđ'Ž  
visit\_NodeName() æŮżæşŦiijŃNodeName() âŃzéĚ■éĆčăžŽă;ăăĐşăĲŦ'èűčçŽĐēĹĆçĆżăĂčăyŃēĹcă

```
import ast

class CodeAnalyzer(ast.NodeVisitor):
    def __init__(self):
```

```

        self.loaded = set()
        self.stored = set()
        self.deleted = set()

    def visit_Name(self, node):
        if isinstance(node.ctx, ast.Load):
            self.loaded.add(node.id)
        elif isinstance(node.ctx, ast.Store):
            self.stored.add(node.id)
        elif isinstance(node.ctx, ast.Del):
            self.deleted.add(node.id)

# Sample usage
if __name__ == '__main__':
    # Some Python code
    code = '''
    for i in range(10):
        print(i)
    del i
    '''

    # Parse into an AST
    top = ast.parse(code, mode='exec')

    # Feed the AST to analyze name usage
    c = CodeAnalyzer()
    c.visit(top)
    print('Loaded:', c.loaded)
    print('Stored:', c.stored)
    print('Deleted:', c.deleted)

```

æĈædĪĴ;æĕĤRèqÑeĤZäyĤĭNāžRĭjNä;äaijŽăĤUăĤrăyÑeĭcèĤZæăuĥŽDèĤŞăĤžĭjŽ

```

Loaded: {'i', 'range', 'print'}
Stored: {'i'}
Deleted: {'i'}

```

æĪĴăĤŔŔŔĭjÑASTăĤŕăžééĂŽĕĤĢ compile() äĢ;æĤŕæĭĕĥijŬeŕŞăžŭæĤġèqNăĂĈăĤNăĕĈĭjŽ

```

>>> exec(compile(top, '<stdin>', 'exec'))
0
1
2
3
4
5
6
7
8
9
>>>

```

## ëõlëõž

ä;Šä;äëČ;äd'šāĽĒæđRæžRäzčçăAăžüäžŌäy■ēŌuāRŪăfæAŗçŽĐæŪuāĂŽīījNă;ăārseČ;ăĒŽăĹăđ'Žăž  
ăĹNăēČīījNčŽyærTčŽšçŽŌçŽĐăījăēĂŠăyĂăžŽăžčçăAçĽĠăōtăĹŗçszăīīj  
exec() äĢ;æTŗăy■īījNă;ăāRfăžēăĒĹăŗĒăōČē;ňæ■ćæĹRăyĂăyĹASTīījN  
çĎŭăRŌēğĆârşăōČçŽĐçzĒēĽCçIJNăōČăĹrăžTæYŗæĂŌæăŭăĂŽçŽĐăĂĆ  
ă;ăēĽYăRfăžēăĒŽăyĂăžŽăŭēăĒŭăĹēæşēçIJNăşRăyĹăĹăīŪçŽĐăĒĹēĽĹăžRçăAīījNăžŭăyTăĹĹă■đ'ăşžçăĂăy  
éĹĂēēAæşĹăĎRçŽĐæYŗīījNăēĆăđĹJă;ăçşēēĂşēĢăŭşăĹĹăžşăTŗēīījNă;ăēĽYēČ;äd'şēĢ■ăĒŽASTăĹēă  
ăyNēĹăYŗăyĂăyĹēçĒēēăZĹă;Nă■RīījNăRfăžēăĂŽēĽĢēĢ■ăŪrēğçăđRăĢ;æTŗă;ŞăžRçăĂăĂă  
éĢ■ăĒŽASTăžŭēĢ■ăŪrăĹŽăžăĢ;æTŗăžççăĂăŗžēşăēĹăŗĒăĒĹăşĂēōĽēŪōăRŸéĢRÉŽ■ăyžăĢ;æTŗă;Şă;ĹçT

```
# namelower.py
import ast
import inspect

# Node visitor that lowers globally accessed names into
# the function body as local variables.
class NameLower(ast.NodeVisitor):
    def __init__(self, lowered_names):
        self.lowered_names = lowered_names

    def visit_FunctionDef(self, node):
        # Compile some assignments to lower the constants
        code = '__globals = globals()\n'
        code += '\n'.join("{0} = __globals['{0}']".format(name)
                           for name in self.lowered_names)
        code_ast = ast.parse(code, mode='exec')

        # Inject new statements into the function body
        node.body[:0] = code_ast.body

        # Save the function object
        self.func = node

# Decorator that turns global names into locals
def lower_names(*namelist):
    def lower(func):
        srclines = inspect.getsource(func).splitlines()
        # Skip source lines prior to the @lower_names decorator
        for n, line in enumerate(srclines):
            if '@lower_names' in line:
                break

        src = '\n'.join(srclines[n+1:])
        # Hack to deal with indented code
        if src.startswith((' ', '\t')):
            src = 'if 1:\n' + src
```



```

top = ast.parse(src, mode='exec')

# Transform the AST
cl = NameLower(namelist)
cl.visit(top)

# Execute the modified AST
temp = {}
exec(compile(top, '', 'exec'), temp, temp)

# Pull out the modified code object
func.__code__ = temp[func.__name__].__code__
return func
return lower

```

äyžažEä;fçTlèfZäyłazčçăAıijNă;ăăRfrazêăČRăyNéİcèfZæăûăEŻiijŽ

```

INCR = 1
@lower_names('INCR')
def countdown(n):
    while n > 0:
        n -= INCR

```

èčĚéěřăŽlăijŽăřE countdown() âĜ;æTřéĜăăEŻäyžçşzăijijăyNéİcèfZæăûăăŘiijŽ

```

def countdown(n):
    __globals = globals()
    INCR = __globals['INCR']
    while n > 0:
        n -= INCR

```

ăIJlæĂgèČ;ætNerTăyııijNăôČăijŽeöl'âĜ;æTřèfRèqNăfñ20%

çŎřăIJlıijNă;ăæYřăııæYřæČşăyžă;ăæL'ĂæIJL'çŽDăĜ;æTřéČ;ăLăăyLèfZăyłèčĚéěřăŽlăŚcıijşæLŰèöyă  
 ä;EăYřıijNèfZăıı'æYřărzăžŎăyĂăžZénYçžğæLĂæIJrærTăeČASTæŞă;IJăĂAăzŘçăAăŞă;IJçıL'çıL'çŽDă

æIJnèLČăRŰăRèad'ŰăyĂăyłăIJlActiveStateăyııad'DçŘEPythonăŰèLČçăAçŽDçnăèLČçŽDăŘç  
 ä;fçTlASTæYřăyĂăyłæZt'ăLăénYçžğçČçŽDăLĂæIJrıijNăzŰăyTăžşæZt'çŏĂăTăžZăĂČăRČèĂČăyNéİcă

## 11.25 9.25 æŊEğçPythonăŰèLČçăA

éŰöécŸ

ä;ăæČşéĂŽèfĜăřEä;ăçŽDăzčçăAăRıçijŰërŚæLŘă;ŎçžğçŽDăŰèLČçăAăİæşèçIJNăôČăžTăşČçŽDă

èğçăEşşæŰzæăŁ

dis æłăăİŰăRřăzèèçñçTlăİèè;ŞăĜžăžă;TPythonăĜ;æTřçŽDăRıçijŰërŚçzŞşadIJăĂČă;NăçCıijŽ

```
>>> def countdown(n):
...     while n > 0:
...         print('T-minus', n)
...         n -= 1
...     print('Blastoff!')
...
>>> import dis
>>> dis.dis(countdown)
...
>>>
```

## èóìéőž

ā;Šā;āæČšèēAçšéēAšā;ăçŽĐĹNăžRăžTăšĆçŽĐèŁRëąNăIJžăĹúcŽĐæŮúăĂŽiijŃdis  
æĹaaiŮæŸřă;ĹæIJĹčTĹčŽĐăĂĆærTăęCăęCăđIJă;ăæČšërTçĹĂçŘEèğčæĂğèČ;çĹ'žă;AăĂĆ  
èćn dis() āĢ;æTřèğčæđŘçŽĐăŮšăġNă■ŮèĹĆçăAăęCăyNăĹ'Ăçđ'žiižŽ

```
>>> countdown.__code__.co_code
b"x
↳ '\x00|\x00\x00d\x01\x00k\x04\x00r)\x00t\x00\x00d\x02\x00|\x00\x00\x83
\x02\x00\x01|\x00\x00d\x03\x008}
↳ \x00\x00q\x03\x00Wt\x00\x00d\x04\x00\x83
\x01\x00\x01d\x00\x00S"
>>>
```

ăęĆăđIJă;ăæČšèĢăūsèğčéĢĹèŁŽăôțăžçăĂiijŃă;ăéIJĂèęAă;ŁçTĹăyĂăžZăIJĹ opcode  
æĹaaiŮăy■ăőŽăžĹ'çŽĐăyŷéĢRăĂĆă;ŃăęCăiijŽ

```
>>> c = countdown.__code__.co_code
>>> import opcode
>>> opcode.opname[c[0]]
>>> opcode.opname[c[0]]
'SETUP_LOOP'
>>> opcode.opname[c[3]]
'LOAD_FAST'
>>>
```

ăęĢæĂĹçŽĐæŸriijŃăIJĹ dis æĹaaiŮăy■ăžăæšăæIJĹ'ăĢ;æTřèŮĹ'ă;ăăžèçijŮçĹNăŮžăijRă;ĹăôžæŸšçŽĐ.  
ăy■ăŁĢiijŃăyŃéĹççŽĐçTšæĹRăŽĹăĢ;æTřăRřăžèărĒăŮšăġNă■ŮèĹĆçăAăžRăĹŮè;Ńă■ăĹR  
opcodes āŠŃăRĆæTřăĂĆ

```
import opcode

def generate_opcodes(codebytes):
    extended_arg = 0
    i = 0
    n = len(codebytes)
    while i < n:
        op = codebytes[i]
```

```

i += 1
if op >= opcode.HAVE_ARGUMENT:
    oparg = codebytes[i] + codebytes[i+1]*256 + extended_arg
    extended_arg = 0
    i += 2
    if op == opcode.EXTENDED_ARG:
        extended_arg = oparg * 65536
        continue
else:
    oparg = None
yield (op, oparg)

```

ä;£çŦíæŨzæşŦæĆäyŦiijŽ

```

>>> for op, oparg in generate_opcodes(countdown.__code__.co_code):
...     print(op, opcode.opname[op], oparg)

```

è£Žçğ■æŨzäijRä;ŁärŦæIJL'äzžçşēēAŞiijŦnä;ääRräzēāL'çŦláōĆæŽ£æ■cäzzä;Ŧä;ääČşēēAæŽ£æ■ćçŽĐ  
äyŦéÍcäĹSäzñçŦläyÄäyĴçd'žä;ŦæĹæijŦŦçd'žæŦŦ'äyĴē£ĠçĹŦiijŽ

```

>>> def add(x, y):
...     return x + y
...
>>> c = add.__code__
>>> c
<code object add at 0x1007beed0, file "<stdin>", line 1>
>>> c.co_code
b'|\x00\x00|\x01\x00\x17S'
>>>
>>> # Make a completely new code object with bogus byte code
>>> import types
>>> newbytecode = b'xxxxxxx'
>>> nc = types.CodeType(c.co_argcount, c.co_kwonlyargcount,
...     c.co_nlocals, c.co_stacksize, c.co_flags, newbytecode, c.co_
↳consts,
...     c.co_names, c.co_varnames, c.co_filename, c.co_name,
...     c.co_firstlineno, c.co_lnotab)
>>> nc
<code object add at 0x10069fe40, file "<stdin>", line 1>
>>> add.__code__ = nc
>>> add(2,3)
Segmentation fault

```

ä;ääRräzēāČRè£ŽæäüēÄ■ād'ğæŦŽèōŦ'èğçēĠLäZĹæŦæžČäĀĆä;EæŦŦiijŦärzäžŦōçijŦŦäEŽæŽŦ'énŦçžğäi  
äzŦäzñäRŦrēČ;çIJşçŽĐēIJÄēēAēĠ■äEŽä■ŦēĹĆçäAäĀĆæIJñēĹĆæIJÄäŦŦōçŽĐēČĹäĹEæijŦŦçd'žäžEè£ŽäyĴæ'  
this code on [ActiveState](#)

## 12 çññ■AçñäïijŽælaaiUäyÓäÑĚ

ælaaiUäyÓäÑĚæYřäzzä;Täd'ğädNçlNäzRçŽDæyāfČiijNärseēdPythonāōL'èčĚlNāzRæIJnèžnāzšæYřä

Contents:

### 12.1 10.1 ædDāzzäyÄäylælaaiUçŽDāsĆçžgāÑĚ

#### éUóécŸ

ä;äæČšārEä;äçŽDäzčçäAçzDçzGæLRçTšā;Lād'ŽāLEāsĆælaaiUædDæLRçŽDāÑĚāĆ

#### èğčāEşæÚzæaL

ārAèčĚæLRāÑĚæYřā;LçōĀā■TçŽDāĀĆāIJæŮGäzūçšzçzšäyŁçzDçzGä;äçŽDäzčçäAïijNāzūçāōāĬærf  
ä;NāeČiijŽ

```
graphics/  
  __init__.py  
  primitive/  
    __init__.py  
    line.py  
    fill.py  
    text.py  
  formats/  
    __init__.py  
    png.py  
    jpg.py
```

äyÄæUçä;āāAŽāLRäzEēfŽäyĀçĆziijNä;āāžTēreèČ;ād'šæL'gèaNāRDçg■importèr■āRēiijNāeCāyNriijŽ

```
import graphics.primitive.line  
from graphics.primitive import line  
import graphics.formats.jpg as jpg
```

#### èóléōž

āōŽāzL'ælaaiUçŽDāsĆæñaçzŠædDārśāČRāIJæŮGäzūçšzçzšäyŁāzzçñNçŽōā;TçzŠædDäyÄæūāōzæY  
æŮGäzū\_\_init\_\_.pyçŽDçŽōçŽDæYřeēAāÑĚāRñäy■āRÑeēRēaNçžgāLñçŽDāÑĚçŽDāRréĀLçŽDāĬiāgNāÑ  
äy;äylä;Nā■RiijNāeCædIJä;äæL'gèaNāžEēr■āRēimport graphicsiijN æŮGäzūgraph-  
ics/\_\_init\_\_.pyārEècñārijaĚē,āzžçñNgraphicsāS;āR■çl'žēŮt'çŽDāEēāōzāĀĆāČRimport  
graphics.format.jpgēfŽæūārijaĚēiijNæŮGäzūgraphics/\_\_init\_\_.pyāšNæŮGäzūgraphics/formats/\_\_init\_\_.py

çziād'gēČlāLEæŮūāĀŽēōl'\_\_init\_\_.pyçl'žçĬĀārsāē;āĀĆä;EæYřæIJL'āžZæČĚāEŮäyNāRrēČ;āÑĚāRñāz  
äy;äylä;Nā■RiijN\_\_init\_\_.pyèČ;ād'šçTlāĬēēGĬāĬāLāē;ā■RælaaiU:

```
# graphics/formats/__init__.py
from . import jpg
from . import png
```

ħĈRèfZæăüăÿĂăÿlæŨĜăžű,çŦlæŁuăŔfăzěăžĚăžĚěĂŽěĚĜimport  
 pics.formatsæİăăžĉăŽĚimport graphics.formats.jpgăžăăŔĬimport graphics.formats.pngăĂĈ  
 \_\_init\_\_.pyçŽĐăĚŭăžŨăÿÿçŦlçŦlæşŦăŇĚăŇăŕĚăđ'ŽăÿlæŨĜăžűăŔĬăžűăĹăŕăÿĂăÿlæĂžè,ŚăŚ;ăŔ■çl'žă  
 æŦŔĚŦŔçŽĐçĬŇăžŔăŚŸăiĵŽăŔŔçŖŕiĵŇă■şă;ĤæşăæIĴL\_\_init\_\_.pyæŨĜăžűă■ŸăIĴiĵŇpythonăž■çĐŭă

## 12.2 10.2 æŌğáĹúæłǵǻıŰèćńǻĚléĈíǻrįǵǻĚęçŽďǻĚǻőǻ

éŮőécŸ

ǎ;Šă;ŁçTlâĂZfrom module import \*ăĂZ ər■ăRěăUũijNăyŃăIJZărzázŌălăaiŮăŁŮăŃěărijaGčzŽDçņē

èġčǎEşæŮźæąŁ

**ǎIjlä;äçŽDælaǎlUäy■āōžāzL'äyÄäytärŸéĜR \_\_all\_\_ ælēæYÕçaõâIJrâlŬăGžēIJĂèęAârījaĞżçŽDăEęă**

**äy; äylä; Nă■Ř:**

```
# somemodule.py
def spam():
    pass

def grok():
    pass

blah = 42
# Only export 'spam' and 'grok'
__all__ = ['spam', 'grok']
```

èóíèőž

```

    āŕ;çōāiĵçĈĹāŖ■āŕzā;ĤçĤĪ      āĀŸfrom      module      import      *āĀŽ,
    ä;EæŸŕāIJĹāōŽāzĹ'āžEāđ'gēGRāŖŸéGRāŖ■çŽDāĹāāIŪäy■écŞçzAä;ĤçĤĪāĀĈ
    æĈCæđIJā;āāy■āAŽāzā;ŦāzŊ, ēĤZæāũçŽDāŕijāĒēāŕĒāijŽāŕijāĒēāĹ'ĀæIJĹ'āy■āzēāyŊāĹŞçzĤāijĀād't'çŽDā
    āŖēäyĀæŪzéĹc,æĈCæđIJāōŽāzĹ'āžE__all__, éĈçāzĹāŖĤæIJĹ'ēcŋāĹŪäy;āĠGžçŽDāyIJēēĤāijŽēcŋāŕijāĠGzāĀĈ

    æĈCæđIJā;āāŕE      __all__      āōŽāzĹ'æĹŖāyĀäyĤç'zāĹŪēāĹ,
    æşqæIJĹ'āyIJēēĤāŕEēcŋāŕijāĒēāĀĈ      æĈCæđIJ      __all__      āŊĒāŖŋæIJāōŽāzĹ'çŽDāŖ■ā■Ū,
    āIJĹāŕijāĒēāŪūāijŦètŭAttributeErrorāĀĈ

```

## 12.3 10.3 ä;ŁçŦłçŻÿárzèùrâ;ĎâŦ■ārijāĚēāŇĚäÿ■ā■ŦāēlāāIŮ

### éŮóéčŸ

ārĚäzččāAçzĎçzĠæĹŦāŇĚ,æČšçŦlimportēr■āŦĚäzŎāŦĚäÿĀäÿlāŇĚāŦ■æšāæIJL'çañçijŮçāAèŁĠçŻĎā

### èğčāĚşæŮzæāĹ

ä;ŁçŦlāŇĚçŻĎçŻÿárzārijāĚērijŇā;ŁäÿĀäÿlāēlāāIŮārijāĚēāŦŇäÿĀäÿlāŇĚçŻĎāŦĚäÿĀäÿlāēlāāIŮ  
äÿ;äÿlā;Ňā■ŦrijŇāAĠGèŮ;āIJlā;äçŻĎæŮĠäzūçşçzşäÿŁæIJL mypackageāŇĚrijŇçzĎçzĠæČäÿŦrijŻ

```
mypackage/  
  __init__.py  
  A/  
    __init__.py  
    spam.py  
    grok.py  
  B/  
    __init__.py  
    bar.py
```

āēČädIJāēlāāIŮmypackage.A.spamèēAārijāĚēāŦŇçŻŏā;ŦäÿŇçŻĎāēlāāIŮgrokiiŇāŏČāžŦērēāŇĚæŇñç

```
# mypackage/A/spam.py  
from . import grok
```

āēČädIJāēlāāIŮmypackage.A.spamèēAārijāĚēäÿ■āŦŇçŻŏā;ŦäÿŇçŻĎāēlāāIŮB.bariiŇāŏČāžŦērēā;ŁçŦl

```
# mypackage/A/spam.py  
from ..B import bar
```

äÿd'äÿlimportēr■āŦĚéČ;æšāāŇĚāŦŦĚāŮāšČāŇĚāŦ■rijŇēĀŇæŸŦā;ŁçŦlāžĚspam.pyçŻĎçŻÿárzèùrâ;ĎâŦ■ārijāĚēāŦŇç

### èŏlēŏž

āIJlāŇĚāĚērijŇæŮčāŦŦäzēä;ŁçŦłçŻÿárzèùrâ;ĎäžşāŦŦäzēä;ŁçŦłçzĹārzèùrâ;ĎāēlārijāĚēāŦŇç  
äÿ;äÿlā;Ňā■ŦrijŻ

```
# mypackage/A/spam.py  
from mypackage.A import grok # OK  
from . import grok # OK  
import grok # Error (not found)
```

āČŦmypackage.AèŁŦæäüä;ŁçŦłçzĹārzèùrâ;ĎâŦ■çŻĎäÿ■āĹ'āžŇād'ĎæŸŦēŁŦārĚāŮāšČāŇĚāŦ■çañçij  
äÿ;äÿlā;Ňā■ŦrijŇāēČädIJā;āæŦzāŦŸāžĚāŇĚāŦ■rijŇā;āārşāŁĚēāzæčĀæşēæĹ'ĀæIJL'æŮĠäzūæĹēāŁŏæ■čæ;  
āŦŦæäürijŇçañçijŮçāAçŻĎâŦ■çğŦrijŻā;ŁçğzāĹlāzččāAāŦŸā;ŮāŦŦēŦ;āĀČäÿ;äÿlā;Ňā■ŦrijŇāžşèŮÿæIJL'ā  
āēČädIJā;ŁçŦłçŻÿárzārijāĚērijŇéČäÿĀāĹĠéČ;ŏkiiŇçŻĎēĀŇā;ŁçŦłçzĹārzèùrâ;ĎâŦ■ā;ĹāŦŦēČ;āijŻāĠžéŮ

importer■āRēcŽĐ . āšŇ . . çIJNètũæIěāŁæzŚçĹ,  
ä;EāōČæŇĠāōŽçŽōā;ŦāR■.äyžā;ŠāL■çŽōā;ŦiijŇ..BäyžçŽōā;Ŧ./BāĀČèŁŽçĝ■ēr■æſŦāRléĀČçŦlāžŌimport  
äyŁäyŁä;Ňā■RiijŽ

```
from . import grok # OK
import .grok # ERROR
```

ār;çōāā;ŁçŦlçŽyāržārījāĒēcIJNètũæIěāČRæYřætŘèĠŁæŮĠžũçſçzçſiijŇä;EæYřäy■èČ;āŁrāōŽāzŁ'āŇĒ  
æIJĀāRŌiijŇçŽyāržārījāĒēāRléĀČçŦlāžŌāIJlāRŁéĀČçŽĐāŇĒäy■çŽĐælaaiŮāĀČārd'āĒūæYřāIJléaúā  
ä;ŇāēČiijŽ

```
% python3 mypackage/A/spam.py # Relative imports fail
```

āRēäyĀæŮzéÍçījŇāēČādIJä;äā;ŁçŦlPythonçŽĐ-méĀŁ'éazæIěæL'ĝèāŇāĒĹāL■çŽĐèĎŽæIJnīijŇçŽyār  
ä;ŇāēČiijŽ

```
% python3 -m mypackage.A.spam # Relative imports work
```

æŽt'ād'ŽçŽĐāŇĒçŽĐçŽyāržārījāĒēcŽĐèČŇæŽrcšèèfE,èfũçIJN [PEP 328](#) .

## 12.4 10.4 āRĒælaaiŮāŁĒāL'sæŁRād'ŽäyŁæŮĠžũ

### éŮóécŸ

ä;āæČšārEäyĀäyŁælaaiŮāŁĒāL'sæŁRād'ŽäyŁæŮĠžũāĀČä;EæYřä;ääy■æČšārEāŁĒççzçŽĐæŮĠžũçz

### èĝčāEşæŮzæaŁ

çlŇāžRælaaiŮāRrāžēēĀŽèŁĠāRŸæŁRāŇĒæIěāŁĒāL'sæŁRād'ŽäyŁçŇŇçŇŇçŽĐæŮĠžũāĀČèĀČèŽŚāy

```
# mymodule.py
class A:
    def spam(self):
        print('A.spam')

class B(A):
    def bar(self):
        print('B.bar')
```

āĀĠèō;ä;āæČšmymodule.pyāŁĒäyžäyd'äyŁæŮĠžũiijŇæfRäyŁāōŽāzŁçŽĐäyĀäyŁçſzāĀČèĕAāAŽāŁrē  
èŁŽèŁŽäyŁçŽōā;ŦäyŇiijŇāŁŽāžžæäyŇæŮĠžũiijŽ

```
mymodule/
__init__.py
a.py
b.py
```

åIÍa.pyæŮĜäzűäy■æŘŠăĚěäzěäyNäzččăAïijŽ

```
# a.py
class A:
    def spam(self):
        print('A.spam')
```

åIÍb.pyæŮĜäzűäy■æRŠåĚěäzěäyNäzččăAïijŽ

```
# b.py
from .a import A
class B(A):
    def bar(self):
        print('B.bar')
```

æIJAăRŎiijŃaIłI\_\_init\_\_.py äy■iijŃaŕE2äylæŮĞäzũçşŸaŔŁaIłläyĂetuiiijŽ

```
# __init__.py
from .a import A
from .b import B
```

æĈædIæŃLĈĖĝeĴZăZă■ēēld'ijŃæL'ĂăžĝĉTŝĉZďăŃĚMyModuleăřĖă;IjăyžăyĂăyĵă■TăyĂĉZďéĂz

```
>>> import mymodule
>>> a = mymodule.A()
>>> a.spam()
A.spam
>>> b = mymodule.B()
>>> b.bar()
B.bar
>>>
```

**èóíèőž**

ǎIǐēfZävIçnäēŁĆäy■čŽDävzēēAēŮōēćŸæŸrävǞäyļēō;ēōāēŮōēćŸīijŃäy■čōā;ǎæŸrǎRēǎyŃæIJZçŦlǎ

```
from mymodule.a import A
from mymodule.b import B
...
```

èfZæăùèĈ;ăùèä;IiijNă;ÈèfZèó'čTlæLúæL'făRŮæŽt'ad'ŽčŽDèr'šæNĚiijNčTlæLúèèAçššéAŞäy■ăRŇ

```
from mymodule import A, B
```

árzaŘŎëĀĚëĀÑëlĀīijNēōl' mymoduleæĹRäyžāyĀäyġad'gçŽDæzŘæŮGäzūæYřæIJĀäyÿèğAçŽDăĂCăj  
 èŁŻæuũaAŻçŽDăEşęTőæYřaĹZāzzāyĀäyġaÑĖÇŽôajTīijNăjfcTĪ\_\_init\_\_.py  
 æŮGäzūæİearEærRéĆİaĹEçšYãRLaIJläyÄætũaĂĆ

a;ŞäYÄäyŧæŧaİUëcñåŁEåŁ'siijNä;äëIJÄèeAçŁ'žåŁnæşŧæĐRăžd'åRL'åijTçTŧçŽĐæŨGăžũăR■ăĂCăyŁăy  
 from .a import A æŧëëŌũăRŨăĂĈ



æTʁ'äylçnäèLĆéČĭä;ŁçTĭáNĚčŽĎçŽyárfzářijaĚěæİéeAŁăĚ■ăřEéąűásCăláăIŮăR■çañçijŮčăAăĹřæžRăž  
ă;IJăyžèŁZăyĂçnäèLĆçŽĎăžűăijyĭijNăřEăžNçz■ăžűèŁšăřijaĚěăĂĆăęCăŽĭæL'Ăçd'žĭijN\_\_init\_\_.pyæŮ  
èęAăAŽăĹrèŁZăyĂçCžĭijN\_\_init\_\_.pyæIJŁçzEăĭőçŽĎăRŸăNŮĭijŽ

```
# __init__.py
def A():
    from .a import A
    return A()

def B():
    from .b import B
    return B()
```

ăIJĹèŁZăyŁçL'ŁæIJăy■ĭijNçszAăŠNçszBèćnäŽŁæ■căyžăIJĹçñňăyĂæñăèőŁéŮőæŮűăŁăèĭ;æL'ĂéIJĂçŽĭ  
ăĭNăęĆĭijŽ

```
>>> import mymodule
>>> a = mymodule.A()
>>> a.spam()
A.spam
>>>
```

ăžűèŁšăŁăèĭ;çŽĎăyžèęAçijžçĆzæŸřçžgæL'ŁăŠNçszăđNăcĂæšěăRřèČĭăijŽăy■ăŮ■ăĂĆă;ăăRřèČĭăijŽ

```
if isinstance(x, mymodule.A): # Error
...

if isinstance(x, mymodule.a.A): # Ok
...
```

ăžűèŁšăŁăèĭ;çŽĎçIJšăőđăĭNă■Ř, èğAăăGăĜEăžŞ multiprocessing/\_\_init\_\_.py  
çŽĎăžŘçăA.

## 12.5 10.5 áĹ'çTĭáŚĭăR■çĭ'zéŮťăřijaĚěçŽăăĭTăĹEăTĭççŽĎăžčçăA

éŮőéćŸ

ăĭăăRřèČĭæIJŁ'ăđ'gėĜRçŽĎăžčçăAĭijNçT'săy■ăRŃçŽĎăžžæĹěăĹEăTĭçăIJřçzt'æŁd'ăĂĆăfRăyĹéČĭăĹEă

èğcăEşşæŮzæąĹ

ăžŮăIJñèťĭăyŁèőĭĭijNă;ăèęAăőŽăžL'ăyĂăyĹéąűçžgPythonăNĚĭijNă;IJăyžăyĂăyĹăđ'gėZEăŘĹăĹEăĭajĂç  
ăIJĹçzšăyĂăy■ăRŃçŽĎçŽăăĭTĭéĜNçzšăyĂçŽyăRŃçŽĎăŚĭăR■çĭ'zéŮťĭijNă;EăŸřèęAăĹăăŮžçTĭăĹěăřĭ

```
foo-package/
  spam/
    blah.py
```

```
bar-package/  
    spam/  
        grok.py
```

āĲĲē£ŽŽäȳłçŻōā;TēĠŇĲĲŇēČ;æĲĲłçĲĲāĲēšāŖŇçŽĎāŚ;āŖ■çł'žēŮt' spamāĀČāĲĲläzzä;TäȳĀäȳłçŻōā;Tēē  
èŌł' æĻŚäžŋçĲĲŇçĲĲŇĲĲŇāēČæĎĲĲāŖEfoo-packageāŚŇbar-  
packageēČ;āĻāāĻŖpythonæĲāĲĲēŭŖā;ĎāžŭāŖĲēŖTāŖĲāĲēēĲĲŽāŖŚçTšāžĀāžĻ

```
>>> import sys  
>>> sys.path.extend(['foo-package', 'bar-package'])  
>>> import spam.blah  
>>> import spam.grok  
>>>
```

äȳđ'äȳłäȳ■āŖŇçŽĎāŇēČŻōā;TēēčŋāŖĻāžŭāĻŖäȳĀēŭĲĲŇā;āāŖŖäžēāŖĲāĲēspam.blahāŚŇspam.grokĲĲŇā

## èŌĲēōž

āĲĲē£ŽēĠŇāŭēä;ĲçŽĎæĲžāĻŭēčŋçġŖäȳžāĀĲāŇēāŚ;āŖ■çł'žēŮt'āĀĲçŽĎäȳĀäȳłçĻ'žā;AāĀČāžŌæĲŇē  
āŇēāŚ;āŖ■çł'žēŮt'çŽĎāĲēšēŤŌæŸŖçāŭāĲĲēāŭčžççŻōā;Täȳ■æšæĲĲ\_\_init\_\_.pyæŮĠäžŭæĲä;ĲäȳžāĲēšā  
äȳłäȳłäȳŇā■ŖĲĲž

```
>>> import spam  
>>> spam.__path__  
_NamespacePath(['foo-package/spam', 'bar-package/spam'])  
>>>
```

āĲĲāŭōžä;■āŇēČŽĎā■ŖçžĎäžŭæŮŭĲĲŇçŻōā;T\_\_path\_\_āŖEēčŋçŤĲāĻŖ(äȳŇāēČ,  
ā;ŠāŖĲāĲēspam.grokæĻŮēĀĲēspam.blahçŽĎæŮŭāĀž).

āŇēāŚ;āŖ■çł'žēŮt'çŽĎäȳĀäȳłēĠēēAçĻ'žçČžæŸŖäžžä;TäžžēČ;āŖŖäžēçŤĲēĠāŭšçŽĎäžççāAæĲēæĻ'Ŗās

```
my-package/  
    spam/  
        custom.py
```

æēČæĎĲĲā;āāŖEä;äçŽĎäžççāAçŻōā;TāŚŇāĲēŭäžŮāŇēäȳĀēŭæŭžāĻāāĻŖsys.pathĲĲŇē£ŽāŖEæŮāçĲĲāĲĲā

```
>>> import spam.custom  
>>> import spam.grok  
>>> import spam.blah  
>>>
```

äȳĀäȳłāŇēæŸŖāŖēēčŋä;ĲäȳžäȳĀäȳłāŇēāŚ;āŖ■çł'žēŮt'çŽĎäȳžēēAæŮžæšTæŸŖæčĀæšēāĲē\_\_file\_\_ā

```
>>> spam.__file__  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>
```

```
AttributeError: 'module' object has no attribute '__file__'
>>> spam
<module 'spam' (namespace)>
>>>
```

æŽt'ad'ŽčŽDāNĚāŚ;āŘ■čl'zéŮt'āŁæAřāŘřžčæščIJŇ [PEP 420](#).

## 12.6 10.6 éĜ■æŮřāŁæ;::æłāłŮ

### éŮóécŸ

ä;ăæČšéĜ■æŮřāŁæ;::ăüşčžŘāŁæ;::čŽDæłāłŮŮijNāZăăyžă;ăăržăĚŮæžŘčăAèŁZèqNăžEăŁôæŤžăĂĆ

### èĝčăEşæŮžæąŁ

ä;ŁçŤlīmp.reload()æłééĜ■æŮřāŁæ;::ăĚŁāŁ■ăŁæ;::čŽDæłāłŮŮăĂĆăy;ăyłă;Nă■ŘiijŽ

```
>>> import spam
>>> import imp
>>> imp.reload(spam)
<module 'spam' from './spam.py'>
>>>
```

### èółèőž

éĜ■æŮřāŁæ;::æłāłŮŮăIJłāijĂăŘŚăŚNērČērŤèŁĜćlNăy■ăyyăyyă;ŁæIJL'çŤlăĂĆă;EăIJłçŤšăžĝčŎřăćČă

reload()æŞşéŽD'ăžEăłāłŮŮăžŤăśČă■ŮăĚyçŽDăEăĚăžzīijNăžŮéĂŽèŁĜéĜ■æŮřāŁ'ĝèqNăłāłŮŮçŽDæžŘ

ăr;çôăăĆă■d'iijNreload()æşşæIJL'æŽt'æŮřăČŘăĂlfrom module import  
nameăĂlèŁZăăüă;ŁçŤlīmpimportërăŘčăřijăĚčçŽDăőŽăžŁ'ăĂĆăy;ăyłă;Nă■ŘiijŽ

```
# spam.py
def bar():
    print('bar')

def grok():
    print('grok')
```

çŎřăIJłăŘřāŁăžd'ăžŚăijŘăijŽèřlīijŽ

```
>>> import spam
>>> from spam import grok
>>> spam.bar()
bar
>>> grok()
grok
```

```
grok
>>>
```

äy■éÄÄGŽPythonä£öæŤžspam.pyçŽDæžŘčãAñijŇãEgrok()ãĜ;æŤræŤžæĹŘè£ŽæüñijŽ

```
def grok():
    print('New grok')
```

çŎřãIJlãŽďãĹřãžd'ãžŠãijŘãijŽèřñijŇéĜ■æŮřãĹæ;;æĹããĹUñijŇãřĹerŤãŷŇè£ŽãŷĹãóđetñijŽ

```
>>> import imp
>>> imp.reload(spam)
<module 'spam' from './spam.py'>
>>> spam.bar()
bar
>>> grok() # Notice old output
grok
>>> spam.grok() # Notice new output
New grok
>>>
```

ãIJlè£ŽãŷĹã;Ňã■Řãŷ■ñijŇã;ãçIJŇãĹřæIJL'2ãŷĹçĹĹæIJñçŽDgrok()ãĜ;æŤrècñãĹæ;;ãÄĆéÄŽãŷŷæĹèert'ñ  
ãŽãæ■d'ñijŇãIJlçŤšãžĝçŎřãçCãŷ■ãŘrèÇ;éIJÄèæAéAçãĚ■éĜ■æŮřãĹæ;;æĹããĹUãÄĆãIJlãžd'ãžŠçŎřãçC

## 12.7 10.7 è£ŘèãŇçŽóã;ŤæĹÚãŎŇçijl'æŮĜãžú

### éŮóécŸ

æĆĹæIJL'ãŷÄãŷĹãüšæĹŘèŤŤãŷžãŇĚãŘñãd'ŽãŷĹæŮĜãžüçŽDãžŤçŤĹñijŇãóCãüšè£IJãŷ■ãE■æŸřãŷãŷĹç

### èĝçãEşæŮzæãĹ

ãçĆãđIJã;ãçŽDãžŤçŤĹçĹŇãžŘãüšçžŘæIJL'ãd'ŽãŷĹæŮĜãžüñijŇã;ããŘřãžèæĹĹã;ãçŽDãžŤçŤĹçĹŇãžŘæŤ;  
ãŷ;ãŷĹã;Ňã■ŘñijŇã;ããŘřãžèãČŘè£ŽæãüãĹŽãžççŽóã;ŤñijŽ

```
myapplication/
  spam.py
  bar.py
  grok.py
  __main__.py
```

ãçĆãđIJ\_\_main\_\_.pyã■ŸãIJlñijŇã;ããŘřãžèçõÄã■ŤãIJřãIJléãüçžĝçŽóã;Ťè£ŘèãŇPythoneğççéĜĹãŽĹñijŽ

```
bash % python3 myapplication
```

èĝççéĜĹãŽĹãřEæĹ'ĝèãŇ\_\_main\_\_.pyæŮĜãžüã;IJãŷžãŷççĹŇãžŘãÄĆ

ãçĆãđIJã;ããŘEã;ãçŽDãžççãAæĹ'ŞãŇĚæĹŘřãŷæŮĜãžüñijŇè£Žçĝ■æĹæIJřãŘŇæãüãžšéÄĆçŤĹñijŇãŷ;ã

```
bash % ls
spam.py bar.py grok.py __main__.py
bash % zip -r myapp.zip *.py
bash % python3 myapp.zip
... output from __main__.py ...
```

## èõìèõž

ãĹŽăžăyĂăyĭçŽôă;ŢăĹŪzipăŪĞăžŭăžŭăŭăĹă\_\_main\_\_.pyăŪĞăžŭăĭăăŕĖăyĂăyĭăŽt'ăd'ğçŽĐPyth  
çŢăžŏçŽôă;ŢăŠŅzipăŪĞăžŭăyŌă■ăăyăŪĞăžŭăIJĹ'ăyĂçĆăy■ăŔŅĭjŅă;ăăŔŕèĈ;èĤŸéIJăèĖAăćđă

```
#!/usr/bin/env python3 /usr/local/bin/myapp.zip
```

## 12.8 10.8 èŕzăŔŪă;■ăžŌăŅĖăy■çŽĐăŢŕă■ŏăŪĞăžŭ

### éŬŏécŸ

ă;ăçŽĐăŅĖăy■ăŅĖăŔŅăžçăAéIJăèĖAăŌžèŕzăŔŪçŽĐăŢŕă■ŏăŪĞăžŭăĂĆă;ăéIJăèĖAăŕ;ăŔŕèĈ;ăIJçŦ

## èğĉăĖşăŪzăăĹ

ăAĞèŏ;ă;ăçŽĐăŅĖăy■çŽĐăŪĞăžŭçzĐçzĞăĹŔăĖĆăyŅĭjŽ

```
mypackage/  
  __init__.py  
  somedata.dat  
  spam.py
```

çŌŕăIJăăAĞèŏ;spam.pyăŪĞăžŭéIJăèĖAăŕzăŔŪsomedata.dataăŪĞăžŭăy■çŽĐăĖĖăŏžăĂĆă;ăăŔŕăžèçŦăĹă

```
# spam.py  
import pkgutil  
data = pkgutil.get_data(__package__, 'somedata.dat')
```

çŢăă■d'ăžğçŢşçŽĐăŔŸéĞŔăŸŕăŅĖăŔŅèŕăŕăŪĞăžŭçŽĐăŌşăğŅăĖĖăŏžçŽĐă■ŪèĹĆă■ŪçŋăyşăĂĆ

## èõìèõž

èĖAăŕzăŔŪăŢŕă■ŏăŪĞăžŭĭjŅă;ăăŔŕèĈ;ăĭjŽăĂ;ăŔŖăžŏçĭjŪăĖŽă;ĤçŦăĖĖç;ŏçŽĐĹ/  
ŌăĹşèĈ;çŽĐăžçăAĭĭjŅăĖĆopen()ăĂĆă;ĖăŸŕèĤçğ■ăŪzăşŢăžşăIJĹ'ăyĂăžŽéŪŏécŸăĂĆ  
éĖŪăĖĹĭjŅăyĂăyĭăŅĖăŕzèğĉéĖĹăŽĭçŽĐă;şăĹ'■ăŭă;IJçŽôă;ŢăĞăžŏŖăşăIJĹ'ăŌğăĹŭăĬăĂĆăŽăă  
çŋăžŅŅĭjŅăŅĖăĂžăyăŏĹ'èĉĖă;IJăyž.zipăĹŪ.eggăŪĞăžŭĭjŅăŖŽăžŽăŪĞăžŭăžŭăy■ăĆŔăIJăŪĞăžŭ

```
pkgutil.get_data('TrawYrayAaylerzaRUwTraw'wU'GazucZDenYczgaueaEuijNay'cTlcoaN'wYra  
get_data('cZDcnayAaylaRCwTrawYraN'wRnaN'wR'cZDa'U'cneyssa'Acj;aaRraze'Zt'wO'w;fcTlaN'w
```

## 12.9 10.9 arEaU'Gazuad'zaLa'aE'alsys.path

### eUoe'cY

ajaaU'asTarijaE'aj;cZDPythonazcc'aAaZaayza'wCwL'AaIJlcZDcZoa;Tay'aIJsys.patheGN'aAcj;aaCs

### egcaEsaU'zaal

aeIJL'ayd'cg'aaycTlcZDaeU'zaijRarEaU'rcZoa;TawzaLa'aLrsys.pathaAc'cnayAcg'uijNaj;aaRraze'w;fcTlaN'

```
bash % env PYTHONPATH=/some/dir:/other/dir python3  
Python 3.3.0 (default, Oct 4 2012, 10:17:33)  
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin  
Type "help", "copyright", "credits" or "license" for more  
->information.  
>>> import sys  
>>> sys.path  
['', '/some/dir', '/other/dir', ...]  
>>>
```

aIJleG'aoZazL'azTcTlcI'NaZRay'uijNefZ'wau'cZDcO'racCaRYeGRaRraIJlcI'NaZRarRaL'aeU'eo'c;oaLU'  
cnnaZ'Ncg'aeU'zawTawYraL'ZazayAayl.pthaeU'GazuuijNarEcZoa;TalaU'ay;agzae'uijN'aCR'efZ'wau'ijZ

```
# myapplication.pth  
/some/dir  
/other/dir
```

efZayl.pthaeU'Gazu'eIJae'AwT;aiJla'sRaylPythoncZDsite-  
packagescZoa;TijN'eAZayya;azO'usr/local/lib/python3.3/site-packages aeLU'eA' ~/lo-  
cal/lib/python3.3/sitepackages'aAc;S'egceGLaZlaRraL'aeU'uijN.pthaeU'Gazu'eGN'aLU'ay;agzae'ecZD'aY'aIJ

### eo'leoz

arT'etu'et'zaLZaIJraL;aeU'GazuuijNaj;aaRreC;aijZa'Ag'aRSazO'aEZayAaylazcc'aAwL'NaL'lerC'eL'Csys.pat

```
import sys  
sys.path.insert(0, '/some/dir')  
sys.path.insert(0, '/other/dir')
```

eZ;cD'uefZeC;aaIJau'ea;IJa'liijNa'wCwYraIJla'wdeutay'awdaayze'DEaijsiijNaZTar;eGR'eAfa'w;fcTlaN'

```
import sys
from os.path import abspath, join, dirname
sys.path.insert(0, join(abspath(dirname(__file__)), 'src'))
```

èĚŽāŕĚsrcçŽŏā;TæûzāŁāāĹŕpathéĜŇĭjŇāŠŇæL'ġèāŇæŔŠāĔĔæ■ēēld'çŽĎžčçăĀāĪĴāŔŇăŷĀăŷłçŽŏā;  
site-packagesçŽŏā;TæŸŕçŋŋăŷL'æŰzāŇĔŇāŠŇæĴāĪŰāŏL'èçĔçŽĎçŽŏā;TăĀĆăĕĆăđĪă;ăæL'ŇāĴāŏL'èç  
packagesçŽŏā;TăĀĆèŽ;çĎŭçŦĴăžŎēĔ■ç;ŏpathçŽĎ.pthæŰĜăžŭăĔĔēāzæŦ;ç;ŏāĪĴsite-  
packageséĜŇĭjŇă;ĒāŏĆēĔ■ç;ŏçŽĎèŭŕă;ĎāŔŕăžæŸŕçşçzşşăŷĴăžză;Tă;ăăŷŇæĪJŽçŽĎçŽŏā;TăĀĆăŽăæ■đ

## 12.10 10.10 éĀŽèĚĜā■ŰçŋăŷşăŔ■ăŕĭjăĔĔæĴāĪŰ

### éŰŏéćŸ

ă;ăæČşăŕĭjăĔĔăŷĀăŷĴæĴāĪŰĭjŇă;ĒæŸŕæĴāĪŰçŽĎăŔ■ă■ŰāĪĴă■ŰçŋăŷşéĜŇăĀĆă;ăæČşăŕză■Űçŋăŷ

### èġčăĒşæŰzæĴĴ

ă;ĔçŦĴimportlib.import\_module()ăĜ;æŦŕæĴæL'ŇāĴĴăŕĭjăĔĔăŔ■ă■Űăŷză■ŰçŋăŷşçzŽăĜžçŽĎăŷĀăŷĴæ

```
>>> import importlib
>>> math = importlib.import_module('math')
>>> math.sin(2)
0.9092974268256817
>>> mod = importlib.import_module('urllib.request')
>>> u = mod.urlopen('http://www.python.org')
>>>
```

import\_moduleăŔĴæŸŕçŏĀă■ŦăĪJŕæL'ġèāŇāŠŇimportçŽŷăŔŇçŽĎæ■ēēld'ĭjŇă;ĒæŸŕèĔŦăŽđçŦşæĴŔç  
ăĕĆăđĪă;ăæ■čăĪĴă;ĔçŦĴçŽĎăŇĔĭjŇimport\_module()ăžşăŔŕçŦĴăžŎçŽŷăŕzăŕĭjăĔĔăĀĆă;ĒæŸŕĭjŇă;ăē

```
import importlib
# Same as 'from . import b'
b = importlib.import_module('.b', __package__)
```

### èŏĴèŏž

ă;ĔçŦĴimport\_module()æL'ŇāĴĴăŕĭjăĔĔæĴāĪŰçŽĎéŰŏéćŸéĀŽăŷŷăĜžçŎŕăĪĴăžæşŔçġ■æŰzăĭJŔçĭjŰă  
ăĪĴăŰġçŽĎžčçăĀĭjŇæĪJL'æŰŭă;ăăĭjŽçĪJŇāĴŕçŦĴăžŎăŕĭjăĔĔçŽĎăĒăžzăĜ;æŦŕ\_\_import\_\_()ăĀĆăŕ;  
éĀŽăŷŷæŽŦ'ăŏžæŸşă;ĔçŦĴăĀĆ  
èĜĴăŏžăžL'ăŕĭjăĔĔèĔĜçĴŇçŽĎénŸçžġăŏđă;ŇèġĀ10.11ăŕŔèĴĆ

## 12.11 10.11 éĀŽè£GéŠ'ā■Řè£IJćÍNāŁăè;ǰæÍaǰİŮ

### éŮóécŸ

äǰăæČšèĜłăőŽăzŁ'PythonçŽĐimportèí■āŘëiǰNăǰŁăŮăőČèČǰăzŌè£IJćÍNăIJžăŽÍăŷŁéÍcéĀŘæŸŌçŽĐă

### èĝčăEşæŮzæǰĹ

éĕŮăĒŁèĕAæŘŘăĜžæİĕçŽĐăŸřăŌŁ'ăĒİéŮóécŸăĀĆæIJñèŁĆèŏłèŏžçŽĐăĀİæČşăĕĆăđIJæşqæIJŁ'ăŷĀăžşăřsæŸřèř'iiǰNăĹSăznçŽĐăŷzèĕAçŽŏçŽĐăŸřăŭsăĒĕăĹEăđŘPythonçŽĐimportèí■āŘëæIJžăĹŭăĀĆăĕĆăđIJăǰăçREèĝčăžEăIJñèŁĆăĒĒĕĹăŌşçREiǰNăǰăăřsèČǰăđ'şăŷžăĒŭăžŮăžzăǰŤçŽŏçŽĐăĀNèĜłăőŽăzŁ'İăIJŁ'ăžEăĕ£ŽăžŽiǰNèŏŁ'ăĹSăznççžĝçz■āŘSăŁ'■ĕřăĀĆ

æIJñèŁĆăăŷăĤCăŸřèŏçèŏăřiǰăĒĕĕř■āŘĕçŽĐăŁ'ĹăşŤăĹşĕČǰăĀĆæIJŁ'ăǰĹăđ'Žçĝ■æŮzæşŤăŘřăžĕăAŹăŷ■ĕ£ĜăŷžăžEăiǰŤçđ'žçŽĐăŮzăǰĕiǰNăĹSăznăiǰĀăĜNăĒĹăđĎĒăăăŷNéÍcé£ŽăŷŤPythonăžçčăAçzŞăđĎiǰž

```
testcode/  
    spam.py  
    fib.py  
    grok/  
        __init__.py  
        blah.py
```

è£ŽăžŽăŮĜăžŭçŽĐăĒĒăŏžăžŭăŷ■ĕĜ■ĕĕAiiǰNăŷ■ĕ£ĜăĹSăznăIJĹăřŘăŷŤăŮĜăžŭăŷ■æŤǰăĒĒăžEăřSéĜè£ŽăăŭăǰăăŘřăžĕăŤNĕřŤăŏČăznăžŭăşĕçIJNăǰŞăŏČăznĕĕăřiǰăĒĕăŮŭçŽĐĕŞăĜžăĀĆăǰNăĕĆriǰŽ

```
# spam.py  
print("I'm spam")  
  
def hello(name):  
    print('Hello %s' % name)  
  
# fib.py  
print("I'm fib")  
  
def fib(n):  
    if n < 2:  
        return 1  
    else:  
        return fib(n-1) + fib(n-2)  
  
# grok/__init__.py  
print("I'm grok.__init__")  
  
# grok/blah.py  
print("I'm grok.blah")
```

è£ŽĒĜNçŽĐçŽŏçŽĐăŸřăĒAĕŏŷĕ£ŽăžŽăŮĜăžŭăǰIJăŷžăĹaǰİŮĕĕñè£IJćÍNăŏŏĕŮăĀĆăžşĕŏŷăIJĀçŏĀă■ŤçŽĐăŮzăiǰŘăřsæŸřăĒĒăŏČăznăŘSăŷČăĹŤăŷĀăŷŤwebæIJ■ăĹăăŽÍăŷŁéÍcăĀĆăIJŁtestcode



```
bash % cd testcode
bash % python3 -m http.server 15000
Serving HTTP on 0.0.0.0 port 15000 ...
```

æIJ■āŁāŻİèƒRèqÑèŧüæİēāŔŌāE■āŔŕāŁİäYÄäYİā■ŧçNñçŽĐPythonèġćéĠāŻİāĂĆ  
çaŏāİä;āāŔŕäzēä;ƒçŦİ urllib èŏƒéŬŏāĹŕèƒIJçĹNæŨĠāzūāĂĆä;NæĆİijŽ

```
>>> from urllib.request import urlopen
>>> u = urlopen('http://localhost:15000/fib.py')
>>> data = u.read().decode('utf-8')
>>> print(data)
# fib.py
print("I'm fib")

def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n-1) + fib(n-2)
>>>
```

äzŌèƒŽäYİæIJ■āŁāŻİāŁäè;;æžŔäzççāAæŸŕæŌëäYNæİēæIJñèĹĆçŽĐā\$žçāĂāĂĆ  
äYžāZæZŒäzçæĹNāĹİçŽĐéĂŽèƒĠ urlopen() æİēæŦüéZEæžŔæŨĠāzūūijŦ  
æĹSäzñéĂŽèƒĠèĠāŏŽäZĹimportèŕ■āŔēæİēāIJāŔŌāŔŕèĠāŁİäYŏæĹSäzñāAŽāĹŕāĂĆ

āŁäè;;èƒIJçĹNæİāİŨçŽĐçñäYĂçġ■æŨzæŧŦæŸŕāĹZāzžäYÄäYİæŸçd'žçŽĐāŁäè;;āĠæŦŕæİēāŏNæĹŔ

```
import imp
import urllib.request
import sys

def load_module(url):
    u = urllib.request.urlopen(url)
    source = u.read().decode('utf-8')
    mod = sys.modules.setdefault(url, imp.new_module(url))
    code = compile(source, url, 'exec')
    mod.__file__ = url
    mod.__package__ = ''
    exec(code, mod.__dict__)
    return mod
```

èƒŽäYİāĠæŦŕäijŽäYŦè;;æžŔäzççāAīijŦāzūā;ƒçŦİ compile()
årEāĒūçijŨèŕSāĹŕäYÄäYİäzççāAāržèšāY■īijŦ çĐūāŔŌāIJİäYÄäYİæŸŕāĹZāzžçŽĐæİāİŨāržèšaçŽĐā■ŨāĒYä

```
>>> fib = load_module('http://localhost:15000/fib.py')
I'm fib
>>> fib.fib(10)
89
>>> spam = load_module('http://localhost:15000/spam.py')
I'm spam
>>> spam.hello('Guido')
```

```

Hello Guido
>>> fib
<module 'http://localhost:15000/fib.py' from 'http://
↳localhost:15000/fib.py'>
>>> spam
<module 'http://localhost:15000/spam.py' from 'http://
↳localhost:15000/spam.py'>
>>>

```

æ■čāēĆä;äæL'ÄëĜAīijŊārŷāžŎčōĀā■TçŽDæÍaāIŮēŁŽäyŁæŸřēāŊā;ŮéĀŽçŽDăĀĆ  
äy■ēŁĜăōČāžŮæšāēIJL'āŋŊăĒăĹŕéĀŽăyŷçŽDimportēŕ■āŔēăy■īijŊăēĆăđIJēēAæŤŕăŊAæŽt'énŸçžġçŽDçz  
äyĀăyŁæŽt'ēĒūçŽDăAŽæşŤæŸŕăĹŽăžžăyĀăyŁēĜăăōŽăžL'āŕijăĒăăŽĹăĀĆçŋăyĀçġ■æŮzæşŤæŸŕăĹŽăž

```

# urlimport.py
import sys
import importlib.abc
import imp
from urllib.request import urlopen
from urllib.error import HTTPError, URLError
from html.parser import HTMLParser

# Debugging
import logging
log = logging.getLogger(__name__)

# Get links from a given URL
def _get_links(url):
    class LinkParser(HTMLParser):
        def handle_starttag(self, tag, attrs):
            if tag == 'a':
                attrs = dict(attrs)
                links.add(attrs.get('href').rstrip('/'))
    links = set()
    try:
        log.debug('Getting links from %s' % url)
        u = urlopen(url)
        parser = LinkParser()
        parser.feed(u.read().decode('utf-8'))
    except Exception as e:
        log.debug('Could not get links. %s', e)
    log.debug('links: %r', links)
    return links

class UrlMetaFinder(importlib.abc.MetaPathFinder):
    def __init__(self, baseurl):
        self._baseurl = baseurl
        self._links = { }
        self._loaders = { baseurl : UrlModuleLoader(baseurl) }

    def find_module(self, fullname, path=None):

```

```

log.debug('find_module: fullname=%r, path=%r', fullname,
→path)
if path is None:
    baseurl = self._baseurl
else:
    if not path[0].startswith(self._baseurl):
        return None
    baseurl = path[0]
parts = fullname.split('.')
basename = parts[-1]
log.debug('find_module: baseurl=%r, basename=%r', baseurl,
→basename)

    # Check link cache
    if basename not in self._links:
        self._links[baseurl] = _get_links(baseurl)

    # Check if it's a package
    if basename in self._links[baseurl]:
        log.debug('find_module: trying package %r', fullname)
        fullurl = self._baseurl + '/' + basename
        # Attempt to load the package (which accesses __init__.
→py)

        loader = UrlPackageLoader(fullurl)
        try:
            loader.load_module(fullname)
            self._links[fullurl] = _get_links(fullurl)
            self._loaders[fullurl] = UrlModuleLoader(fullurl)
            log.debug('find_module: package %r loaded',
→fullname)
        except ImportError as e:
            log.debug('find_module: package failed. %s', e)
            loader = None
        return loader

    # A normal module
    filename = basename + '.py'
    if filename in self._links[baseurl]:
        log.debug('find_module: module %r found', fullname)
        return self._loaders[baseurl]
    else:
        log.debug('find_module: module %r not found', fullname)
        return None

def invalidate_caches(self):
    log.debug('invalidating link cache')
    self._links.clear()

# Module Loader for a URL
class UrlModuleLoader(importlib.abc.SourceLoader):
    def __init__(self, baseurl):

```

```

        self._baseurl = baseurl
        self._source_cache = {}

    def module_repr(self, module):
        return '<urlmodule %r from %r>' % (module.__name__, module.__
↪__file__)

    # Required method
    def load_module(self, fullname):
        code = self.get_code(fullname)
        mod = sys.modules.setdefault(fullname, imp.new_
↪module(fullname))
        mod.__file__ = self.get_filename(fullname)
        mod.__loader__ = self
        mod.__package__ = fullname.rpartition('.')[0]
        exec(code, mod.__dict__)
        return mod

    # Optional extensions
    def get_code(self, fullname):
        src = self.get_source(fullname)
        return compile(src, self.get_filename(fullname), 'exec')

    def get_data(self, path):
        pass

    def get_filename(self, fullname):
        return self._baseurl + '/' + fullname.split('.')[-1] + '.py'

    def get_source(self, fullname):
        filename = self.get_filename(fullname)
        log.debug('loader: reading %r', filename)
        if filename in self._source_cache:
            log.debug('loader: cached %r', filename)
            return self._source_cache[filename]
        try:
            u = urlopen(filename)
            source = u.read().decode('utf-8')
            log.debug('loader: %r loaded', filename)
            self._source_cache[filename] = source
            return source
        except (HTTPError, URLError) as e:
            log.debug('loader: %r failed. %s', filename, e)
            raise ImportError("Can't load %s" % filename)

    def is_package(self, fullname):
        return False

    # Package loader for a URL
    class UrlPackageLoader(UrlModuleLoader):

```

```

def load_module(self, fullname):
    mod = super().load_module(fullname)
    mod.__path__ = [ self._baseurl ]
    mod.__package__ = fullname

def get_filename(self, fullname):
    return self._baseurl + '/' + '__init__.py'

def is_package(self, fullname):
    return True

# Utility functions for installing/uninstalling the loader
_installed_meta_cache = { }
def install_meta(address):
    if address not in _installed_meta_cache:
        finder = UrlMetaFinder(address)
        _installed_meta_cache[address] = finder
        sys.meta_path.append(finder)
        log.debug('%r installed on sys.meta_path', finder)

def remove_meta(address):
    if address in _installed_meta_cache:
        finder = _installed_meta_cache.pop(address)
        sys.meta_path.remove(finder)
        log.debug('%r removed from sys.meta_path', finder)

```

äyÑéÍcæYřäyÄäyŁäzd'äžŠäijŽerİijNäijTçd'žäžEäæCä;Tä;ŁçTİäL■éÍçŽDäzčçäAüijŽ

```

>>> # importing currently fails
>>> import fib
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> # Load the importer and retry (it works)
>>> import urlimport
>>> urlimport.install_meta('http://localhost:15000')
>>> import fib
I'm fib
>>> import spam
I'm spam
>>> import grok.blah
I'm grok.__init__
I'm grok.blah
>>> grok.blah.__file__
'http://localhost:15000/grok/blah.py'
>>>

```

èŁŽäyŁçL'žæŁçŽDæŰžæäLäijŽäŁL'èçEäyÄäyŁçL'žäŁñçŽDæšæL'çäŽİ  
UrlMetaFinder                      äŁđä;NüijN                      ä;IJäyž                      sys.meta\_path  
äy■æIJÄäRÖçŽDäŁđä;ŠäÄĆ      ä;ŠäŁäİŰèçnärijäEëæŰüüijNäijŽä;İæ■ō      sys.meta\_path

äy■çŽĐæšæL;ăŽlăōŽă;■ælaaiUăĂĆ      âIJlêŽăylă;Nă■Răy■iijNUrlMetaFinder  
 aōdă;NæYræIJAăRŌăyĂăylæšæL;ăŽlăŪzæăLiiijNă;ŞălaaiUăIJlăzză;TăyĂăylæŽŏéĂŽăIJræŪzéČ;æL;ăy  
 ä;IJăyžăyÿègAçŽĐăōđçŎræŪzæăLiiijNUrlMetaFinder  
 çşzăNĚèčĚăIJlăyĂăylçTlăLŭæNĜăōŽçŽĐURLăylăĂĆ âIJlăEĚĚĆiijNæšæL;ăŽlăĂŽèĚGăLŞăRŪæNĜăō  
 ârijaĚĚçŽĐæŪăăĂŽiijNălaaiUăR■aijZëuşăuşæIJLçŽĐéŞ;æŎă;IJărzærTăĂĆăĉĆăđIJæL;ăLrăžEăyĂăylă  
 äyĂăylă■TçNňçŽĐ UrlModuleLoader çşzèčñçTlălăžŎĚĚIJçlNăIJžăŽlăylăLăLăè;ăžRăžççăAăžŭăLŽăž  
 èĚŽéGŇçijŞă■YéŞ;æŎăçŽĐăyĂăylăŎşăŽăæYréAăĚă■ăy■ăĚĚĉAçŽĐHTTPèrŭăśĆéG■ăđ■ârijaĚĚăĂĆ  
 èĜlăōŽăZăLărijaĚĚçŽĐçňăžNçg■æŪzæşTæYrçijŪăEŽăyĂăylăŚlă■RçZl'æŎăĥNăĚăLr  
 sys.path      âRŸéGRăy■ăŎžiiijN      èrĚăLăNăşRăžŽçŽŏă;TăŚ;ăR■ælaaijRăĂĆ      âIJl  
 urlimport.py äy■æŭăăLăăĉCăyNçŽĐçşzăŞNæTŕæNăĜ;æTŕiijŽ

```

# urlimport.py
# ... include previous code above ...
# Path finder class for a URL
class UrlPathFinder(importlib.abc.PathEntryFinder):
    def __init__(self, baseurl):
        self._links = None
        self._loader = UrlModuleLoader(baseurl)
        self._baseurl = baseurl

    def find_loader(self, fullname):
        log.debug('find_loader: %r', fullname)
        parts = fullname.split('.')
        basename = parts[-1]
        # Check link cache
        if self._links is None:
            self._links = [] # See discussion
            self._links = _get_links(self._baseurl)

        # Check if it's a package
        if basename in self._links:
            log.debug('find_loader: trying package %r', fullname)
            fullurl = self._baseurl + '/' + basename
            # Attempt to load the package (which accesses __init__.
            ↪py)
            loader = UrlPackageLoader(fullurl)
            try:
                loader.load_module(fullname)
                log.debug('find_loader: package %r loaded', ↪
            ↪fullname)
            except ImportError as e:
                log.debug('find_loader: %r is a namespace package', ↪
            ↪fullname)
                loader = None
            return (loader, [fullurl])

        # A normal module
        filename = basename + '.py'
        if filename in self._links:
  
```

```

        log.debug('find_loader: module %r found', fullname)
        return (self._loader, [])
    else:
        log.debug('find_loader: module %r not found', fullname)
        return (None, [])

    def invalidate_caches(self):
        log.debug('invalidating link cache')
        self._links = None

# Check path to see if it looks like a URL
_url_path_cache = {}
def handle_url(path):
    if path.startswith(('http://', 'https://')):
        log.debug('Handle path? %s. [Yes]', path)
        if path in _url_path_cache:
            finder = _url_path_cache[path]
        else:
            finder = UrlPathFinder(path)
            _url_path_cache[path] = finder
        return finder
    else:
        log.debug('Handle path? %s. [No]', path)

def install_path_hook():
    sys.path_hooks.append(handle_url)
    sys.path_importer_cache.clear()
    log.debug('Installing handle_url')

def remove_path_hook():
    sys.path_hooks.remove(handle_url)
    sys.path_importer_cache.clear()
    log.debug('Removing handle_url')

```

òēAä;ŁçŦİēfZäyİēŭră;ĐæşæL;ăZİiijŇă;ăăRİēIJĂēēAăIJÍ sys.path  
 äy■ăLăăĖĖURLēŞ;æŐēăĂĆă;ŇăēĆījŽ

```

>>> # Initial import fails
>>> import fib
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Install the path hook
>>> import urlimport
>>> urlimport.install_path_hook()

>>> # Imports still fail (not on path)
>>> import fib
Traceback (most recent call last):

```

äĖſeTõçCzârſæYř handle\_url() åĴ;æTřrijNăoČcěcnæũzâŁăăLřăžE sys.  
 path\_hooks âRÝeGRäy■āĀĆ â;Ş sys.path çŽDăodă;Şēcñad'DçŘEæUűrijNăijZerÇçTí  
 sys.path\_hooks äy■čŽDăGĵ;æTřrāĀĆ æčĆædIJăzză;TăyĂăylăG;æTřreŁTăŽďăžEăyĂăylăšěæL'¿ăŽlăržesă  
 sys.path ăodă;ŞăŁăè;;æłaiUăĀĆ  
 èŦIJćÍNăłaiUăŁăè;;euşăEűázŮčŽDăŁăè;;ă;ŁçTłæŰzæşTăGăăžŌæYřăyĂăêuűčŽDăĀĆă;NăeĆriiž

èóíèőž

```

    aIjleřęçzEeöleöžázNál'riijNæIJL'ćĆžęęAąijžęřĆçŻDæYřriijNPythonçŻDælaaIŮāĀAāNĖāSŇarıjāĖĕæIJ
    aŋsą;ęçzŘéIŇäyřarŇçŻDPythonçIŇāžŘāSŸāžšāčLārSęĆ;çş;éĀŽāōCāžnāĀĆ
    æĹSāIJleĖŽéGŇæŌleŇRäyĀāžŽāĀijçŻDāŌžęřçŻDæŮGæaçāSŇāžęçsriijNāNĖæNŇ    im-
    portlib module āSŇ PEP 302. æŮGæaçāĖĖāōžāIJleĖŽéGŇäyāijŽęćnéĖāđ'āāŘŘāĹriijNäyāēĖGæĹSāIJleĖŽ
    éęŮāĖĹriijNāęĆæđIJā;āāĆšāĹZāžžäyĀäyĭæŮřçŻDælaaIŮāržęsąriijNā;ęçĹĹ    imp.
    new_module() āĖ;æŮriijŽ

```



```
>>> import imp
>>> m = imp.new_module('spam')
>>> m
<module 'spam'>
>>> m.__name__
'spam'
>>>
```

æÍááÍŮáŕžèšæÉŽžáÿÿæIJL'äÿÄäZæIJšæIJZásðæÄgüijNáNĚæNň \_\_\_\_\_file\_\_\_\_  
üijLè£RèaÑæÍááÍŮáLäè;ìèr■áRëçŽDæŮGäzŭäR■üijL' äšNĚ \_\_\_\_\_package\_\_\_\_ (äNĚäR■)äÄĆ

äĚŮæñüijNæÍááÍŮäijŽècñègčéĜLäZÍçijš■ÿèŭæIēāÄĆæÍááÍŮçijš■ÿäRŕäzèäIJL■Ůäÿÿ  
sys.modules äÿ■ècñæL;äLŕäÄĆ äZäÿzæIJL'äZÈè£Žäÿlçijš■ÿæIJZäLüüijNéÄŽäÿÿäRŕäzèäŕEçijš■ÿäš

```
>>> import sys
>>> import imp
>>> m = sys.modules.setdefault('spam', imp.new_module('spam'))
>>> m
<module 'spam'>
>>>
```

äĚĆædIJçzŽäóŽæÍááÍŮäŭšçzR■ÿäIJléCčázLäŕšäijŽçŽt' æŌèèŌŭä; ŮäŭšçzRècñäLZäzžè£ĜçŽDæÍááÍŮŕ

```
>>> import math
>>> m = sys.modules.setdefault('math', imp.new_module('math'))
>>> m
<module 'math' from '/usr/local/lib/python3.3/lib-dynload/math.so'>
>>> m.sin(2)
0.9092974268256817
>>> m.cos(2)
-0.4161468365471424
>>>
```

çTšäzŌäLZäzžæÍááÍŮä;LçóÄ■TüijNä;LäóžæÿšçijŮäEŽçóÄ■TäĜ;æTŕæŕTäeĆçññäÿÄéČlälEçŽD  
load\_module() äĜ;æTŕäÄĆ è£ŽäÿæŮžæäLçŽDäÿÄäÿlçijžçĆzæÿŕä;LéŽ;äd' DçRĚäd' ■æIĆæČĚäĚtæŕT  
äÿžäZĚäd' DçRĚäÿÄäÿlänĚüijNä;äèeÄeĜ■æŮŕäóðçŌŕæŽóeÄŽimportèr■áRëçŽDäžTäsĆeÄzè; SüijLæŕTäeĆa  
æL'gèaÑéCčázZæŮGäzŭüijNèöç;òèŭŕä;Dç■L'üijL'äÄCè£Žäÿläd' ■æIĆæÄgäŕsæÿŕäÿžäZÄäZLæIJAäè;çŽt' æŌ

æL'l'äsTimportèr■áRëä;LçóÄ■TüijNä;EæÿŕäijŽæIJL'ä;Läd'ŽçgžäLlæš■ä;IJäÄĆ  
æIJÄénÿäsCäÿLüijNäŕijäĚæš■ä;IJècñäÿÄäÿlä;■äžŌsys.meta\_pathäLŮèäÿäÿçŽDäÄIJäĚČèŭŕä;DäÄlæšèæ  
äĚĆædIJä;äè;šäĜžäóççŽDäÄijüijNäijŽçIJNäLŕäÿNéIcè£ZæäüüijŽ

```
>>> from pprint import pprint
>>> pprint(sys.meta_path)
[<class '_frozen_importlib.BuiltinImporter'>,
<class '_frozen_importlib.FrozenImporter'>,
<class '_frozen_importlib.PathFinder'>]
>>>
```

ä;šæL'gèaÑäÿÄäÿlèr■áRëæŕTäeĆ import fib æŮüüijNègčéĜLäZlāijŽéA■äŌĚsys.mata\_pathäÿ■çŽD  
èŕÇçTlāóCäzñçŽD find\_module() æŮžæšTäóžä;■æ■ççäóçŽDæÍááÍŮäLäè;äZlāÄĆ

āŖŕāzēēĀŽēŁĠāōđēĬŊāĭēçĬĬŊçĬĬŊĭĭĴ

```
>>> class Finder:
...     def find_module(self, fullname, path):
...         print('Looking for', fullname, path)
...         return None
...
>>> import sys
>>> sys.meta_path.insert(0, Finder()) # Insert as first entry
>>> import math
Looking for math None
>>> import types
Looking for types None
>>> import threading
Looking for threading None
Looking for time None
Looking for traceback None
Looking for linecache None
Looking for tokenize None
Looking for token None
>>>
```

æŝĭæĎŖçĬĬŊ find\_module() æŰzæŝŦæŶŕæĀŌæăăĭĬĭæŕŔăŷĀăŷĭăŕĭĵăĒēăŕŝēçŋēğăăŖŝçŽĎăĀĆ  
ēŁŽăŷĭæŰzæŝŦăŷçŽĎpathăŖçæŦŕçŽĎăĭĬçŦĭæŶŕăđŦĎçŖĒăŊĒăĀĆ  
ăđŦŽăŷĭăŊĒēçŋăŕĭĵăĒēŕĭĵŊăŕŝæŶŕăŷĀăŷĭăŕŕăĬĬăŊĒēçŽĎ\_\_path\_\_  
ăŝđæĀğăŷĭæĬĭăĬŕçŽĎēŭŕăĭĎăĬŰēăĭăĀĆ ēçĀæĬĭăĬŕăŊĒēçŽĎăŖçžĎăžăŭăŕŝēçĀæçĀæŝēēŁŽăžŽēŭŕăĭĎăĀ  
æŕŦăçĀæŝĭæĎŖăŕžăžŌ xml.etree ăŝŊ xml.etree.ElementTree  
çŽĎēŭŕăĭĎēĒçĭŕĭĴ

```
>>> import xml.etree.ElementTree
Looking for xml None
Looking for xml.etree ['/usr/local/lib/python3.3/xml']
Looking for xml.etree.ElementTree ['/usr/local/lib/python3.3/xml/
↳etree']
Looking for warnings None
Looking for contextlib None
Looking for xml.etree.ElementPath ['/usr/local/lib/python3.3/xml/
↳etree']
Looking for _elementtree None
Looking for copy None
Looking for org None
Looking for pyexpat None
Looking for ElementC14N None
>>>
```

ăĬĬŊ sys.meta\_path äŷĬæŝēæĬĭăŽĭçŽĎăĭçĭăăĬĭĒēçĀĭĭĵŊăŕĒăăŌčăžŌēŶŝăđŦŕçğžăĬŕéŶŝăŕçĭĭĵŊ

```
>>> del sys.meta_path[0]
>>> sys.meta_path.append(Finder())
>>> import urllib.request
>>> import datetime
```

çŒřăIJlăjăçIJNăy■ăLřăzăză;TēçSăGzăzEiijNăZăăyZăřijăĖĖĖĖĖsys.meta\_pathăy■çŽĐăĖŭăzŬăăđă;Şăđ'Đă  
ĖĖŽăŬăăZăiijNăjăăRlăIJL'ăIJlăřijăĖĖăy■ă■ŬăIJlălăăIŬçŽĐăŬăăZăL'■Ėç;çIJNăLřăăŒĖĖĖĖĖăRŠiijŽ

```
>>> import fib
Looking for fib None
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>> import xml.superfast
Looking for xml.superfast ['/usr/local/lib/python3.3/xml']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named 'xml.superfast'
>>>
```

ăjăăzNăL'■ăăŒĖĖĖĖGăyĂăylă■TēŬăIJlçşşălăăIŬçŽĐăşşăL;ăŽlriijNĖĖŽăylăŬř  
UrlMetaFinder çşçŽĐăĖşĖŦăăĈăăyĂăyl UrlMetaFinderăăđăĖNĖĖĖăŭăăLăăLř  
sys.meta\_path çŽĐăIJnăř;riijNă;IJăyZăIJăăRŎăyĂăylăşşăL;ăŽlăŬăăLăăĈăă  
ăĖĈăđIJĖĖĖĖĖĖĖĈçŽĐălăăIŬăR■ăy■Ėç;ăăŒăZă;■riijNăřsăijZĖĖĖĖĖŽăylăşşăL;ăŽlăđ'DçRĖăŎLăăĈăă  
ăđ'DçRĖăăNĖçŽĐăŬăăZăĖIJăĖĖăAşlăĖĐRiijNăIJlpathăRĈăăTřăy■ăNĖGăăŒççŽĐăăIjĖIJăĖĖăĖĖĖĖĖăăşĖriij  
ăĖĈăđIJăy■ăŬřriijNĖřăă■RălăăIŬăĖĖĖăZă;ŞăşđăZăŎăĖŭăzŬăşşăL;ăŽlăăŭăĖĖĖă;çTĖăŎLăăĈăă

ăřăzăZăŎăNĖçŽĐăĖŭăzŬăđ'DçRĖăăRřăIJl UrlPackageLoader  
çşzăy■ĖĖĖăL;ăLřăăĈăăĖĖŽăylçşzăy■ăijZăřijăĖĖăNĖăR■riijNĖăNăăŬřăŎăăLăăĖ;ăřăzăŦçŽĐ  
\_\_init\_\_.pyăăŬăăZăăăĈăăăăŒăZăşăijZĖĖç;çăălăăIŬçŽĐăă\_\_path\_\_ăă  
ăşđăăĖřriijNĖĖŽăyăă■ăă;ĖĖĖăĖăriijNăăZăăyZăIJlăăĖ;ăNĖçŽĐă■RălăăIŬăŬăĖĖĖŽăylăăIjăijZĖĖĖăijăçZăă  
find\_module()ĖřĈçŦlăăĈăăşşăZăŎăŭăřă;ĐçŽĐăřijăĖĖĖŦăă■RăăŬřĖĖŽăZăăĖăĖĖçççŽĐăyăăylăăL'ăăŦriijNăă  
ăĖŦăăZăăĖç;çşĖĖăŞriijNăsys.pathăăŬřăyăăylăPythonăşşăL;ălăăIŬçŽĐçZăă;ŦăŦŬăălriijNă;ăăĖĖĖriijZăă

```
>>> from pprint import pprint
>>> import sys
>>> pprint(sys.path)
['',
 '/usr/local/lib/python3.3.zip',
 '/usr/local/lib/python3.3',
 '/usr/local/lib/python3.3/plat-darwin',
 '/usr/local/lib/python3.3/lib-dynload',
 '/usr/local/lib/...3.3/site-packages']
>>>
```

ăIJl sys.pathăy■çŽĐăřRăyăăylăăđă;ŞĖç;ăijZĖĖĖĖĖăđăŬçŽĐçZăăŒăăLřăyăăylăşşăL;ăŽlăřăZăşăyăă  
ăjăăRřăZăăĖĖĖĖĖĖĖĖĖĖĖ sys.path\_importer\_cache  
ăŎçIJNăyNĖĖŽăZăZăşşăL;ăŽlriijZăă

```
>>> pprint(sys.path_importer_cache)
{'.': FileFinder('.'),
 '/usr/local/lib/python3.3': FileFinder('/usr/local/lib/python3.3'),
 '/usr/local/lib/python3.3/': FileFinder('/usr/local/lib/python3.3/
→'),
 '/usr/local/lib/python3.3/collections': FileFinder('...python3.3/
→collections'),
```

```

'/usr/local/lib/python3.3/encodings': FileFinder('...python3.3/
↳ encodings'),
'/usr/local/lib/python3.3/lib-dynload': FileFinder('...python3.3/
↳ lib-dynload'),
'/usr/local/lib/python3.3/plat-darwin': FileFinder('...python3.3/
↳ plat-darwin'),
'/usr/local/lib/python3.3/site-packages': FileFinder('...python3.3/
↳ site-packages'),
'/usr/local/lib/python3.3.zip': None}
>>>

```

```

sys.path_importer_cache ærT sys.path äijZæZt'ad'gçCzïijN
åZäyZäoCäijZäyZæL'ÄæIJL'ècñäLæ; äzççäAçZDçZöa;Tëořa;TäoČäznçZDæšæL; åZlāĀĆ
èfZāNĒæNñāNĒçZDā■RçZöa;TïijNèfZāZZeĀZāyYāIJ sys.path
äy■æYřäy■ā■YāIJlçZDāĀĆ

```

```

èçAæL'gëaÑ import fib iijNäijZéazāZRæčĀæšë sys.path äy■çZDçZöa;TāĀĆ
årzāZŌæfRäytlçZöa;TïijNāR■çğřāĀIJfībāĀIäijZècñäijäçzZçZyāZTçZD sys.
path_importer_cache äy■çZDæšæL; åZlāĀĆ èfZäyLāRřäzèèŌ'ä;āāLZāzZeĜlāuŝçZDæšæL; åZlāZūā

```

```

>>> class Finder:
...     def find_loader(self, name):
...         print('Looking for', name)
...         return (None, [])
...
>>> import sys
>>> # Add a "debug" entry to the importer cache
>>> sys.path_importer_cache['debug'] = Finder()
>>> # Add a "debug" directory to sys.path
>>> sys.path.insert(0, 'debug')
>>> import threading
Looking for threading
Looking for time
Looking for traceback
Looking for linecache
Looking for tokenize
Looking for token
>>>

```

```

āIJlèfZéĜNïijNā;āāRřäzëäyZāR■ā■ŪāĀIJdebugāĀIāLZāzZäyĀäyLāŪřçZDçijŞā■Yāōdā;ŞāZūārĒāōCèŌ;
sys.path äyLçZDçñnāyĀäyLāĀĆ āIJæL'ĀæIJL'æŌëäyNæIëçZDārijaĒëäy■ïijNā;āāijZçIJNāLřā;äçZDæšæL;
äy■èfĜïijNçTšāZŌāōCèfTāZd (None, [])ïijNéCčāZLād' DçREèfZçI'NäijZçzğçz■ād' DçREäyNāyĀäyLāōdā;Şā

```

```

sys.path_importer_cache çZDā;fçTlècñäyĀäyLā■YāĆlāIJl sys.path_hooks
äy■çZDāG;æTřāLŪëalæŌgāLūāĀĆ èrTërTäyNèIççZDā;Nā■RïijNāōCäijZæyĒéZd' çijŞā■YāZūçZ
sys.path_hooks æūZāLāäyĀäyLāŪřçZDëŭřā;DæčĀæšëāĜ;æTř

```

```

>>> sys.path_importer_cache.clear()
>>> def check_path(path):
...     print('Checking', path)
...     raise ImportError()

```

```

...
>>> sys.path_hooks.insert(0, check_path)
>>> import fib
Checked debug
Checking .
Checking /usr/local/lib/python3.3.zip
Checking /usr/local/lib/python3.3
Checking /usr/local/lib/python3.3/plat-darwin
Checking /usr/local/lib/python3.3/lib-dynload
Checking /Users/beazley/.local/lib/python3.3/site-packages
Checking /usr/local/lib/python3.3/site-packages
Looking for fib
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'
>>>

```

æ■çÇä;äæL'ÀëĜAīijNcheck\_path()      åĜ;æTřecñæfRäył      sys.path  
äy■çŽDăođä;ŞerÇçTlāĀĆ      äy■éa;īijNçTsāžŌæLZăĜzāžE      ImportError      āijĆāyīijN  
āTřeeČ;äy■āijZāRŚçTšāžEīijLāzĒāžĒārEæčĀæšëè;ñçĝžĀłrsys.path\_hooksçŽDäyNäyĀäylāĜ;æTřīijL'āĀĆ  
çšëéAşžāžEæĀŌæăsys.pathæYřæĀŌæăuëcñād'DçRĒçŽDīijNā;āārseČ;ædDāzzāyĀäylēĜlāōŽāzL'èurā;

```

>>> def check_url(path):
...     if path.startswith('http://'):
...         return Finder()
...     else:
...         raise ImportError()
...
>>> sys.path.append('http://localhost:15000')
>>> sys.path_hooks[0] = check_url
>>> import fib
Looking for fib # Finder output!
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named 'fib'

>>> # Notice installation of Finder in sys.path_importer_cache
>>> sys.path_importer_cache['http://localhost:15000']
<__main__.Finder object at 0x10064c850>
>>>

```

èŁŽārsæYřæIJñēŁcæIJĀāRŌēČlāLEçŽDăĒşçTōçCzāĀĆāžNăođäyLiijNäyĀäylçTlālēāIJlsys.pathäy■æ  
ā;ŞăoČāžñëcñçrāŁrçŽDæŪŭāĀZīijNäyĀäylæŪřçŽD      UrlPathFinder  
ăođä;NecñāLZăžzāžūëcñæT;ăĒē      sys.path\_importer\_cache.  
āžNāRŌīijNæL'ĀæIJL'ēIJĀēçAæčĀæšë sys.path çŽDārijāĒëèér■āRēéČ;āijZā;ŁçTlā;ăçŽDēĜlāōŽāzL'æšë  
āşžāžŌëurā;DārijāĒëçŽDăNĒād'DçRĒçŁ■ā;ōæIJL'çCzād'■æiĆīijNāžūāyTēu\$  
find\_loader() æŪzāşTēŁTāZdāĀijæIJL'ăĒşāĀĆ āřzāžŌçōĀā■TāēāāIŪīijNfind\_loader()  
èŁTāZdāyĀäylāĒĒçzD(loader,      None)īijN      āĒŪāy■çŽDload-  
eræYřāyĀäylçTlāžŌārijāĒëēāāIŪçŽDăLæ;ĵāZlāođä;NāĀĆ

```

    áržāžŌäyÄäyġæŽŏéĂŽçŽĐāŇĒījŇfind_loader()    èĤTāŽđäyÄäyġæĚČçzĐ(loader,
path)ījŇāĒŭäy■çŽĐloaderæYřäyÄäyġçTġāžŌārijāĒēāŇĒījLāžŭæLġèāŇ__init__.pyījL'çŽĐāĤæġġāŽġāŏđäġ
pathæYřäyÄäyġāijŽāġġāŇāŇŭāŇĒēçŽĐ    __path__    āśđæĀġçŽĐçŽŏāġTāġŬēāġāĤ
āġŇāēČījŇāēČæđĲĲāšžçāĲURLæYř    http://localhost:15000    āžŭäyTäyÄäyġçTġæġŬæLġèāŇ
import grok ,    éČčāžġ    find_loader()    èĤTāŽđçŽĐpathāřsāijŽæYř    [    āĤYhttp:
//localhost:15000/grokāĤŽ    ]

```

```

    find_loader()    èĤYēēĀēČġād'ĐçŘĒäyÄäyġāŚġāŘ■çġ'žēŮr'āŇĒēāĤĤ
äyÄäyġāŚġāŘ■çġ'žēŮr'āŇĒēäy■æĲĲ'äyÄäyġāĤĤæŝTçŽĐāŇĒēçŽŏāġTāŘ■ījŇāġĒæYřäy■āŇYāĲĲ__init__.pyæŮ
èĤŽæāŭçŽĐēĤījŇfind_loader()    āĤĒēāžèĤTāŽđäyÄäyġæĚČçzĐ(None,    path)ījŇ
pathæYřäyÄäyġçŽŏāġTāġŬēāġījŇçTġāŏČæġēāđĲāžžāŇĒēçŽĐāŏŽāžĤ'æĲĲ__init__.pyæŮĠāžŭçŽĐ__path__
āržāžŌēĤŽçġ■æČĒāĒījŇārijāĒēāĲĲāġŬāijŽççġç■āĤ'■ēāŇāŌžæčĤæŝēsys.pathäy■çŽĐçŽŏāġTāĤĤ
āēČæđĲĲæĤġāĤŕāžĒāŚġāŘ■çġ'žēŮr'āŇĒēījŇāĤĤ'āĲĲ'çŽĐçŽŝæđĲĲēŭŕāġĐēčŇāĤāġĤŕäyĤēġŭæġēāđĲāžžæĲĲā
āĤŝāžŌāŚġāŘ■çġ'žēŮr'āŇĒēçŽĐæŽt'ād'ŽāĤæĲāŕēŕŭāĤĤĤ10.5ārĤēĤĤāĤĤ

```

```

    æĤ'ĤæĲĲ'çŽĐāŇĒēēČġāŇĒēāŖŇāžĒäyÄäyġāĤĤēČĤēŭŕāġĐēŏġçġōījŇāŖŕāžēāĲĲ__path__āśđæĀġäy■çĲĲĲ

```

```

>>> import xml.etree.ElementTree
>>> xml.__path__
['/usr/local/lib/python3.3/xml']
>>> xml.etree.__path__
['/usr/local/lib/python3.3/xml/etree']
>>>

```

```

    āžŇāĤ'■æŘĤāĤījŇ__path__çŽĐēŏġçġŏæYřéĂŽèĤĠ    find_loader()
æŮžæŝTēĤTāŽđāĲīæŌġāĤçŽĐāĤĤäy■èĤĠījŇ__path__æŌēäyŇāġēāžŝēčŇsys.path_hooksäy■çŽĐāĠġæT
āŽāæ■đ'ījŇāġĒāŇĒēçŽĐāŇĤçžĐāžŭēčŇāĤæġġāŖŌījŇāġ■āžŌ__path__äy■çŽĐāŏđäġŝāijŽēčŇ
handle_url()    āĠġæTŕæčĤāæŝēāĤĤ    èĤŽāijŽārijēĠt'æŮřçŽĐ    UrlPathFinder
āŏđäġŇēčŇāĤŽāžžāžŭäyTēčŇāĤāāĒēāĤĤŕ    sys.path_importer_cacheäy■āĤĤ

```

```

    èĤYæĲĲ'äyġēŽġçČžāřsæYř    handle_url()    āĠġæTŕāžēāĤĤāŏČēŭŝāĤĤēČĤāġĤçTġçŽĐ
__get_links()    āĠġæTŕāžŇēŮr'çŽĐāžđ'āžŝāĤĤ    āēČæđĲĲāġāçŽĐæŝēæĤġāŽġāŏđçŌŕēĲĲāēēĲāġĤçTġāĤĤŕāĒŭ
æĲĲāŖŕēČġèĤŽāžŽæġāġŮāijŽāĲĲæŝēæĤġāŽġæŝ■āĲĲæĲŝēŮr'èĤŽēāŇæŽt'ād'ŽçŽĐārijāĒēāĤĤ
āŏČāŖŕāžēārijēĠt'    handle_url()    āŝŇāĒŭāžŮæŝēæĤġāŽġēČĤāĤĤēŽŭāĒēäyĤçġ■ēĤŝāġŝāġçŌŕçĤŬæĤāĤ
äyžāžĒēġçēĠēĤçġ■āŖŕēČġæĤġījŇāŏđçŌŕäy■æĲĲ'äyÄäyġēčŇāĤŽāžžçŽĐæŝēæĤġāŽġçijŝāŇYījĤŕæŕäyĤ
āŏČāŖŕāžēēĲāĤĤāĤŽāžžēĠāđ'■æŝēæĤġāŽġçŽĐēŮŏēYāĤĤ
āŖēāđ'ŮījŇāyŇēĤççŽĐāžčçāĲçĤĤĠāŕŕāŖŕāžēçāŏāĤĤæŝēæĤġāŽġäy■āijŽāĲĲāĤġāŇāŇŮēŝġæŌēēŽĒāĤĤçŽĐ

```

```

# Check link cache
if self._links is None:
    self._links = [] # See discussion
    self._links = _get_links(self._baseurl)

```

```

    æĲĲāŖŌījŇāŝēæĤġāŽġçŽĐ    invalidate_caches()
æŮžæŝTæYřäyÄäyġāŭēāĒŭæŮžæŝTījŇçTġāġēäyĒēČĤĲāĤēēČġijŝāŇYāĤĤ
èĤŽäyġæŮžæŝTāĤ■çTġāĤŬēŕČçTġ    importlib.invalidate_caches()
çŽĐæŮŭāĤŽēčŇēġēāŖŝāĤĤ    āēČæđĲĲāġāæČŝēŏĤURLārijāĒēēĤēēĠ■æŮŕēŕžāŖŮēŝġæŌēāĤŬēāġçŽĐēŕĲāŖŕā

```

```

    āŕžæŕTäyŇäyđ'çġ■æŮžæāĤījĤĤāĤŏæTžsys.meta_pathæĤŮāġĤçTġäyÄäyġēŭŕāġĐēŝĤ'āŇĲījĤāĤĤ
āġĤçTġsys.meta_pathçŽĐārijāĒēēĤēēāŖŕāžēæŇĤçĒēĠāŭŝçŽĐēĲĲāēēĲāġĤŝād'ĐçŘĒæġāġŮāĤĤ
āġŇāēČījŇāŏČāžŇāŖŕāžēāžŌæTŕæ■ŏāžŝäy■ārijāĒēæĤŬāžēäy■āŖŇāžŌäyĤēĤŇāġāġŮ/āŇĒēāđ'ĐçŘĒæŮžäy

```

æIJǻRÖijŇāzžèöőäjäèŁśĆzæUúéÚťčIJNçIJN PEP 302 äžěáŘLim-  
portlibčŽDæŮĞæąčãĂĆ

éŮőécŸ

èġčăẸșæŮźæąŁ

èŁZävleUóécYàRràzëä;ŁçTl10.11ârRèŁĆäy■aRŃæauçŽDariiǎĖĕēŠl'ǎ■ŘæIJzǎLúæleǎođčŎřāĀĆäyNélc

```
# postimport.py
import importlib
import sys
```



```

from collections import defaultdict

_post_import_hooks = defaultdict(list)

class PostImportFinder:
    def __init__(self):
        self._skip = set()

    def find_module(self, fullname, path=None):
        if fullname in self._skip:
            return None
        self._skip.add(fullname)
        return PostImportLoader(self)

class PostImportLoader:
    def __init__(self, finder):
        self._finder = finder

    def load_module(self, fullname):
        importlib.import_module(fullname)
        module = sys.modules[fullname]
        for func in _post_import_hooks[fullname]:
            func(module)
        self._finder._skip.remove(fullname)
        return module

def when_imported(fullname):
    def decorate(func):
        if fullname in sys.modules:
            func(sys.modules[fullname])
        else:
            _post_import_hooks[fullname].append(func)
        return func
    return decorate

sys.meta_path.insert(0, PostImportFinder())

```

è£ŽæüüijŇä;ääřsäŘřäzë;£çŤÍ when\_imported() èčĚéěřăŽĺăžĚüijŇä;ŇăęĆüjŽ

```

>>> from postimport import when_imported
>>> @when_imported('threading')
... def warn_threads(mod):
...     print('Threads? Are you crazy?')
...
>>>
>>> import threading
Threads? Are you crazy?
>>>

```

ä;IJäyžäyÄäyŁæŽť äôđéŽĚčŽĎä;Ňă■ŘüijŇä;ääŘřëČ;æČřăIJĺăůřă■ŸăIJĺčŽĎăóŽăzŁ'äyŁéÍćæůzăŁăèčĚé



```

from functools import wraps
from postimport import when_imported

def logged(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print('Calling', func.__name__, args, kwargs)
        return func(*args, **kwargs)
    return wrapper

# Example
@when_imported('math')
def add_logging(mod):
    mod.cos = logged(mod.cos)
    mod.sin = logged(mod.sin)

```

## ẽõlẽõž

æIJñèŁĆæŁĂæIJřä; İetŮäžŎ10.11ârRèŁĆäy■èõšèfřèfGçŽĎárijãĚěéŠl'ã■ŘijŇázúčl■ä;IJăfôæŤžãĀĆ

@when\_imported ěĚěěřãŽlčŽĎä;IJçŤlæŸřæšlãĚŇãIJlãrijãĚěæŮűècñæŁĀæt'zçŽĎad'ĎçŘĚãŽlãĠ;ã  
 èřèèĉĚěěřãŽlæĉĀæššsys.modulesæĬæššçIJŇælaaiŮæŸřãŘçIJšçŽĎãűçzŘècñãŁæ;ĵăžĚãĀĆ  
 æĉĈæđIJæŸřçŽĎërĬijŇërëad'ĎçŘĚãŽlècñçŇŇã■šërĈçŤlãĀĆäy■ĎŮijŇãd'ĎçŘĚãŽlècñæűzãŁãĀĽr  
 \_post\_import\_hooks æ■ŮãĚÿäy■çŽĎäyĀäyĽãŁŮèãĽäy■ãŎžãĀĆ  
 \_post\_import\_hooks çŽĎä;IJçŤlãřsæŸřæŤűéZEæŁĀæIJL'çŽĎäyžæřRäyĽæĽãĽŮæšlãĚŇçŽĎad'ĎçŘĚ  
 äyĀäyĽæĽãĽŮãŘřäžæšlãĚŇãd'ŽäyĽad'ĎçŘĚãŽlãĀĆ

èèAèõl'æĽãĽŮãřijãĚěãŘŎèğèãŘŚæűzãŁăçŽĎãĽlã;IJijŇPostImporter  
 çšžècñèðç;õäyžsys.meta\_pathçŇŇäyĀäyĽãĚĈçt'ããĀĆãóĈäijŽæ■ŤèŎŮæŁĀæIJL'æĽãĽŮãřijãĚěæš■ä;IJãĀĆ

æIJñèŁĆäy■çŽĎPostImporter çŽĎä;IJçŤlãžúäy■æŸřãŁæ;ĵæĽãĽŮijŇĚĀŇæŸřèĠäyèãrijãĚ  
 åóđéŽĚçŽĎárijãĚěècñãğŤæt'ççžŽä;■ăžŎsys.meta\_pathäy■çŽĎãűzãŮæššæŁ;ãŽlãĀĆ  
 PostImporterLoader çšžäy■çŽĎ imp.import\_module()  
 åĠ;æŤřècñæĀšã;šçŽĎërĈçŤlãĀĆ äyžăžĚæĀĽãĚ■éŽũãĚěæŮăçžĽã;ĽçŎřijŇPostImporter  
 æĽlãŇãžĚäyĀäyĽæŁĀæIJL'ècñãŁæ;ĵèfGçŽĎæĽãĽŮæZEăŘĽãĀĆ  
 æĉĈæđIJäyĀäyĽæĽãĽŮãŘ■ã■ŸãIJlãřsãijŽçŽt'æŎèècñãŁççŤæŎĽ'ãĀĆ

ã;šäyĀäyĽæĽãĽŮècñ imp.import\_module() æŁæ;ĵãŘŎijŇ  
 æŁĀæIJL'ãĽlã\_post\_import\_hooksècñæšlãĚŇçŽĎad'ĎçŘĚãŽlècñèřĈçŤlãijŇã;ĽçŤlæŮřãŁæ;ĵæĽãĽŮã;IJäyž

æIJL'äyĀçĆžéIJăèèAæšlãĎŘçŽĎæŸřæIJŇæIJžäy■éĀĆçŤlãžŎèĈçăžžéĀŽèĽĠ imp.  
 reload() ècñæŸçãijŘãŁæ;ĵçŽĎæĽãĽŮãĀĆ äžšãřsæŸřèřt'ijŇãèĈæđIJă;ããŁæ;ĵäyĀäyĽăžŇãĽ■ăűšècñãŁă  
 âŘëad'ŮijŇŇëèAæŸřã;ăăžŎsys.modulesäy■ãĽăéŽd'æĽãĽŮçĎũãŘŎãĚ■éĠæŮřãrijãĚěëijŇãd'ĎçŘĚãŽlãŘĽã

æŽt'ad'ŽăĚšăžŎãrijãĚěãŘŎéŠl'ã■ŘăĽæAæřèřũãŘĆèĀĆ PEP 369.

## 12.13 10.13 aóL'ècĚċġAæIJL'çŽĐăÑĚ

### éŮóécŸ

ä;ăæĈşëeAăóL'ècĚăŸĂăŸlçñăŸL'æŮzăÑĚiijNă;EæŸræşææIJL'æİĈéŽŔăŕEăóĈăóL'ècĚăĹŕçşçzçşPythonæĹŮèĂĚiijNă;ăăŔŕeĈ;æĈşëeAăóL'ècĚăŸĂăŸlă;ŽèĠăũă;ĚçŤlçŽĐăÑĚiijNèĂNăŸ■æŸŕçşçzçşăŸĹéİcæL'Ăă

### èġcăEşæŮzæaĹ

PythonæIJL'ăŸĂăŸlçŤlæĹăóL'ècĚçŽă;ŤiijNéĂŽăŸŸçşzăiijăĂĪ~/.local/lib/python3.3/site-packagesăĂİăĂĈ èeAăijzăĹăIJlèĚŽăŸlçŽă;ŤăŸ■ăóL'ècĚăÑĚiijNăŔŕă;ĚçŤlăóL'ècĚéĂĹéqzăĂĪ-userăĂİăĂ

```
python3 setup.py install --user
```

æĹŮèĂĚ

```
pip install --user packagename
```

ăIJlsys.pathăŸ■çŤlæĹŮçŽĐăĂĪsite-packagesăĂİçŽă;Ťă;■ăžŮçşçzçşçŽĐăĂĪsite-packagesăĂİçŽă;ŤăzNăL'■ăĂĈ âŽăæ■d'iijNă;ăăóL'ècĚăIJlèĠéİççŽĐăÑĚăŕşæŕŤçşçzçşăũăóL'ècĚçŽĐăÑiijĹăŕ;çăăžăŸ■æĂzæŸŕèĚæăũiijNèeAăŔŮăEşăžŮçñăŸL'æŮzăÑĚçăçŔEăŽlġiijNăŕŤăeĈdistributeæĹŮp

### èóİèőž

éĂŽăŸŸăÑĚăiijŽećăóL'ècĚăĹŕçşçzçşçŽĐsite-packagesçŽă;ŤăŸ■ăŮžġiijNăũŕă;ĐçşzăiijăĂĪ/usr/local/lib/packagesăĂİăĂĈ äŸ■èĚġiijNèĚæăũăĂŽéİĂèeAæIJL'çăçŔEăŤŸæİĈéŽŔăžăŸŤă;ĚçŤlşudoăŤ;ăzd'ăĂĈăŕşçđŮă;ăæIJL'èĚæăũçŽĐăİĈéŽŔăŮzæL'ġeăNăŤ;ăzd'iijNă;ĚçŤlşudoăŮzăóL'ècĚăŸĂăŸlæŮŕçŽĐiijNăŔŕeĈ

ăóL'ècĚăÑĚăĹŕçŤlæĹŮçŽă;ŤăŸ■ăĂŽăŸŸæŸŕăŸĂăŸlæIJL'æŤĹçŽĐæŮzæaĹiijNăóĈăĂeðŸă;ăăĹŽăžă

ăŔeăd'ŮiijNă;ăèĚŸăŔŕăžăăĹŽăžăŸĂăŸlèŽŽæNşçŮŕăcĈiijNèĚŽăŸlæĹŤăžăăIJlăŸNăŸĂèĹĈăiijŽèðşăĹŕă

## 12.14 10.14 aĹŽăžžæŮŕçŽĐPythonçŮŕăcĈ

### éŮóécŸ

ä;ăæĈşăĹŽăžžăŸĂăŸlæŮŕçŽĐPythonçŮŕăcĈiijNçŤlæİăóL'ècĚăĹăăĪŮăŤNăÑĚăĂĈăŸ■èĚġiijNă;ăăŸ■æĈşăóL'ècĚăŸĂăŸlæŮŕçŽĐPythonăĚNéŽEiijNăžşăŸ■æĈşăŕçşçzçşçşPythonçŮŕăcĈăžġçŤş

### èġcăEşæŮzæaĹ

ä;ăăŔŕăžăă;ĚçŤl pyvenv âŤ;ăzd'ăĹŽăžžăŸĂăŸlæŮŕçŽĐăĂĪèŽŽæNşăĂİçŮŕăcĈăĂĈèĚŽăŸlăŤ;ăzd'èćăăóL'ècĚăIJlPythonèġcĈĠăŽlăŔNăŸĂçŽă;ŤiijNăĹŮWindowsăŸĹéİççŽĐScriptsçŽă;ŤăŸ

```
bash % pyvenv Spam
bash %
```

äijäçžŽ pyvenv äŚ;äzd'çŽĐäŘ■ä■ÜæÝřäĚëëÄècñäĹŽäzzçŽĐçŽōä;ŤäŘ■äÄĆä;ŞècñäĹŽäzzaŘÖiijŇS

```
bash % cd Spam
bash % ls
bin include lib pyvenv.cfg
bash %
```

äĹĹbinçŽōä;Ťäy■iijŇä;äaijŽæĹ;äĹräyÄäyĹäŘřäzëä;£çŤĹçŽĐPythonèğçéĠäŽĹiijŽ

```
bash % Spam/bin/python3
Python 3.3.0 (default, Oct 6 2012, 15:45:22)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more_
↵information.
>>> from pprint import pprint
>>> import sys
>>> pprint(sys.path)
['',
 '/usr/local/lib/python3.3.zip',
 '/usr/local/lib/python3.3',
 '/usr/local/lib/python3.3/plat-darwin',
 '/usr/local/lib/python3.3/lib-dynload',
 '/Users/beazley/Spam/lib/python3.3/site-packages']
>>>
```

èĴZäyĹèğçéĠäŽĹçŽĐçĹ;ççCzârşæÝřäzÜçŽĐsite-packagesçŽōä;Ťècñèö;ç;ōäyžæŮräĹŽäzzçŽĐçŎřäčČ  
äëČædĪJä;äëÄÄöĹ'èçĚçñnäyĹæŮzâŇĚiijŇäöČäznäijŽècñäöĹ'èçĚäĪĹéČçéĠŇiijŇèÄŇäy■æÝřéÄŽäyçşžçz  
packagesçŽōä;ŤäÄČ

## èöĹèöž

äĹŽäzzèŽŽæŇşçŎřäčČèÄŽäyÿæÝřäyžäžĚäöĹ'èçĚäŞŇçōaçĤĚçñnäyĹæŮzâŇĚäÄČ  
æ■čäëČä;ääĪĹä;Ňä■Räy■çĪJŇäĹřçŽĐéČçæüiijŇsys.path  
äŘŸéĠRäŇĚäŖñæĹèèĠäžŎçşžçşPythonçŽĐçŽōä;ŤiijŇ äÄŇ site-  
packagesçŽōä;ŤäüşçžŘècñéĠ■äöžä;■äĹräyÄäyĹæŮřçŽĐçŽōä;ŤäÄČ

æĪĹäžĚäyÄäyĹæŮřçŽĐèŽŽæŇşçŎřäčČiijŇäyŇäyÄæ■čârşæÝřäöĹ'èçĚäyÄäyĹäŇĚçōaçĤĚäŽĹiijŇærŤä  
ä;ĚäöĹ'èçĚèĴZæäüçŽĐäüëäĚüäŞŇäŇĚçŽĐæŮüäÄŽiijŇä;äéĪJäëëÄçäöäĤĹä;ää;£çŤĹçŽĐæÝřèŽŽæŇşçŎřäčČ  
äöČäijŽârĚäŇĚäöĹ'èçĚäĹräŮräĹŽäzzçŽĐsite-packagesçŽōä;Ťäy■äŎžäÄČ

är;çōäyÄäyĹèŽŽæŇşçŎřäčČçĪJŇäyĹäŎžæÝřPythonäöĹ'èçĚçŽĐäyÄäyĹäđ'■äĹüiijŇ  
äy■èĴĠäöČäöđéŽĚäyĹäŖĹäŇĚäŖnäžĚârŞéĠRäĠäĠäyĹæŮĠäzüäŞŇäyÄäžçñëârŮéŞ;æŎëäÄČ  
æĹÄæĪĹæäĠäĠĚäžŞäĠ;æŮĠäzüäŞŇäŖräĹ'ğëäŇèğçéĠäŽĹéČ;æĹèèĠäŎŞæĹèçŽĐPythonäöĹ'èçĚäÄČ  
äžäæ■đ'iijŇäĹŽäzzèĴZæäüçŽĐçŎřäčČæÝřä;ĹäöžæÝŞçŽĐiijŇäzüäyŤäĠäžäŎäy■äijŽæŮĹäŮæĪJäžĹèĤä

ézŸèöđ'æČĚäĤäyŇiijŇèŽŽæŇşçŎřäčČæÝřçĹ'ççŽĐiijŇäy■äŇĚäŖnäžä;ŤéçĹäđ'ÜçŽĐçñnäyĹæŮzâş  
äŘřäzëä;£çŤĹäĪJ—system-site-packagesäĹĹäĹ'ëäzæĹëäĹŽäzzèŽŽæŇşçŎřäčČçĪJŇä;ŇäëçĪijŽ

```
bash % pyvenv --system-site-packages Spam
bash %
```

èùšăđ'ŽăĚšăžŎ pyvenv âŠŇěŽŽæŇşçŎŕăcĈçŽĎăŋæAŕăŔăŕăžěăŔĈèĂĈ [PEP 405](#).

## 12.15 10.15 âĹĖăŔŖŖăŇĚ

### éŬŏécŸ

ăĵăăŭşçžŔĉĵĴăĚŽăžĖăŷĂăŷłæIJL'çŦĭçŽĎăžŖĵĴŇæĈşăŕĖăŕăŔăŕăžěăŔĈèĂĈ

### èğĉăĖşæŰžæąĹ

ăĖĈăđIJăĵăæĈşăĹĖăŔŖŖăŇĚăĵçŽĎăžçĉăAĵĵŇçŇăŷĂăžŭăžŇăŕŖŖăŷŖŕçžŽăŕăŔăŕăžěăŔĈèĂĈăĵłăŦŕăŷĂçŽĎăŔŖŖăŇĚăĵĵĴŇăŷĂăŷłăĖŷăđŇçŽĎăĜĵăŦŕăžŖşăŇĚăĵĴçşžăĵĵĵăŷŇéĭĉèĚæăŭĵĴ

```
projectname/
  README.txt
  Doc/
    documentation.txt
  projectname/
    __init__.py
    foo.py
    bar.py
    utils/
      __init__.py
      spam.py
      grok.py
  examples/
    helloworld.py
  ...
```

èĖAèŏŦ'ăĵçŽĎăŇĚăŔŖăŕăžěăŔŖŖăŷŖŕçžŽăŕăŔăŕăžěăŔĈèĂĈăĵłăŦŕăŷĂçŽĎăŔŖŖăŇĚăĵĵĴŇăŷĂăŷłăŦŕăžŖşăŇĚăĵĴçşžăĵĵĵăŷŇéĭĉèĚæăŭĵĴ `setup.py` ĵĴŇçşžăĵĵĵăŷŇéĭĉèĚæăŭĵĴ

```
# setup.py
from distutils.core import setup

setup(name='projectname',
      version='1.0',
      author='Your Name',
      author_email='you@youraddress.com',
      url='http://www.you.com/projectname',
      packages=['projectname', 'projectname.utils'],
)
```

ăŷŇăŷĂæŭĵĴŇăŕŖŖăŷŖŕăŕăžěăŔŖŖăŷŖŕçžŽăŕăŔăŕăžěăŔĈèĂĈăĵłăŦŕăžŖşăŇĚăĵĴçşžăĵĵĵăŷŇéĭĉèĚæăŭĵĴ `MANIFEST.in` æŰĜăžŭĵĴŇăĹŰăĜžăĹĂæIJL'ăĴĴăĵçŽĎăŇĚăĵĵĴŇăŷĂăŷłăŦŕăžŖşăŇĚăĵĴçşžăĵĵĵăŷŇéĭĉèĚæăŭĵĴ

çaõafI setup.py aŠÑ MANIFEST.in æŨGäzúaŤ; aIŤlă;ăçŽĐaŇĚçŽĐæIJĂéąűçžğçŽoă;Ťăy■ăĂĆ  
äyĂăŮëă;ăăușçzRăAžZăĚëfZăžŽiijŇă;ăărsăRřrăžăČRăyŇeIcèfŽæaũăL'ğëăŇăŠ;ăzd' æIěăLŽăžžăyĂăyIæzĚ

ãĉĈaijZăLZăzžyÄäylæŨGăzüærTăeCăĂİprojectname-1.0.zipâĂİ æŁŨ  
âĂİprojectname-1.0.tar.gzâĂİ, âĖüă;ŞăĹ İetŪăžÖă;ăçŽDşşçğşşāšăRřăĂĆăeCăđIJăYăĂĽGă■čăÿÿtjŃ  
ēfZăylæŨGăzüârşăRřăžeăRŚēĂAçżZăĽnăžžă;£çŢlæŁŨêĂĖăYŁăijăëGş Python Package In-  
dex.

[illegible]

áržāžŌæūL'āRŁāLřCæL'ľāsTčŽDžččāAæL'šāNěäyŌāLEāRSārsæŽt'ad'■æiCčCžāžEāĀC  
 čňň15čňāāržāEšžāžŌCæL'ľāsTčŽDžččāZæŪžéiCčšēēřEæIJL'äyĀāžZēřčžEēōšēgčijNčL'žāLńāYřāIJ15.2ārRēl'

æIŋçnǽæYráĚsǻžŌǎIǰĭ;ŚçzIǰǻžTčTlǎSŋǎLĚǻyČǎijRǻžTčTlǻy■ǻ;ǻTčTlčZǰǎRǰčǻ■ǻyžécYǎĀčǻyžécYǎ

### 13.1 11.1 äJäyžaóæŁuçńräyŎHTTPæJ■åŁäžd'äžŠ

ä:äëIJÄëeAëÄŽëfGHHTPa■RéoóäzëäóëæŁuçnrćŽDæŨzàiJRéoóëÉŮoád'Žçg■æIJ■nŁaŁaŹĆä;NäeĆiiŃäy

## èġċăĖşăŮzăăĹ

ărzăžŎçŏĂă■ŦçŽĐăžŊăĈĖăĭèĕrt'ĭĭjŊéĂŽăÿÿăĵĚçŦĭ urllib.  
request æĹăăĭŮăřśăđ' şăžĖăĂĈăĹŊăĖĈĭĭjŊăŔŖŖĂăĤăÿĂăÿĹçŏĂă■ŦçŽĐHTTP  
GETĕrŭăśĈăĹŖĕĚĬJçĹŊçŽĐăĬ■ăĹăÿĹĭĭjŊăŔŖăžĕĕĚăăŭăĂŽĭĭjŽ

```
from urllib import request, parse

# Base URL being accessed
url = 'http://httpbin.org/get'

# Dictionary of query parameters (if any)
parms = {
    'name1' : 'value1',
    'name2' : 'value2'
}

# Encode the query string
querystring = parse.urlencode(parms)

# Make a GET request and read the response
u = request.urlopen(url+'?' + querystring)
resp = u.read()
```

ăĖĈăđĬăĵăĕĬĂĕĖĂăĵĚçŦĭPOSTăŮzăşŦăĬĭĕrŭăśĈăÿzăĴşăÿ■ăŔŖŖĂăăşĕĕĕŕĈăŔĈăŦĭĭjŊăŔŖăžĕăŕĖăŔ  
urlopen() âĢĵăŦĭĭjŊăŕśăĈŖĕĚăăŭĭĭjŽ

```
from urllib import request, parse

# Base URL being accessed
url = 'http://httpbin.org/post'

# Dictionary of query parameters (if any)
parms = {
    'name1' : 'value1',
    'name2' : 'value2'
}

# Encode the query string
querystring = parse.urlencode(parms)

# Make a POST request and read the response
u = request.urlopen(url, querystring.encode('ascii'))
resp = u.read()
```

ăĖĈăđĬăĵăĕĬĂăĖĂăĬăŔŖŖăĢççŽĐĕrŭăśĈăÿ■ăŔŖăĴăÿĂăžŽĕĢăŏŽăžĹçŽĐHTTPăđ't'ĭĭjŊăĴŊăĖĈăĹ  
user-agent â■ŮăŏĴăŔŖăžĕăĹŽăžăÿĂăÿĹăŊĖăŔŖăăŮăŏĴăĂĭĵçŽĐă■ŮăĖĭĭjŊăžŭăĹŽăžăÿĂăÿĹRequestă  
urlopen() ĭĭjŊăĖĈăÿŊĭĭjŽ

```
from urllib import request, parse
...
```

```

# Extra headers
headers = {
    'User-agent' : 'none/ofyourbusiness',
    'Spam' : 'Eggs'
}

req = request.Request(url, querystring.encode('ascii'),
↳headers=headers)

# Make a request and read the response
u = request.urlopen(req)
resp = u.read()

```

æĈædIJéIJÀèèAäzd'äzŠçŽĐæIJ■āLæŕTäyLélcçŽĐäĭNā■RéĈĭèèAād'■æĬĈiijNāzšèöyāžTërëāŌzçIJNç  
 requests äžŠiijLhttps://pypi.python.org/pypi/requestsiijL'āĀĈäĭNāèĈiijNäyNéĬcèŁZäyŁçd'žäĭNéĜĜçTĭreques

```

import requests

# Base URL being accessed
url = 'http://httpbin.org/post'

# Dictionary of query parameters (if any)
parms = {
    'name1' : 'value1',
    'name2' : 'value2'
}

# Extra headers
headers = {
    'User-agent' : 'none/ofyourbusiness',
    'Spam' : 'Eggs'
}

resp = requests.post(url, data=parms, headers=headers)

# Decoded text returned by the request
text = resp.text

```

äĖšäžŌrequestsäžŠiijNäyÄäyĭāĀijäĭUäyÄæRRçŽĐçL'zæĀğārsæŸŕāōĈèĈĭäzèād'Žçğ■æŰžāijRāzŌèŕūa  
 resp.text äyèçžZæĹSäžñçŽĐæŸŕäžèUnicodeèğççāAçŽĐāŠ■āžTæŰĜæIJñāĀĈäĭEæŸŕiijNāèĈædIJāŌžè  
 resp.content iijNāŕsāijZäĭŰāLŕāŌšāğNçŽĐäžNèŁZāLŰæTŕæ■ōāĀĈāRēäyÄæŰzélĈiijNāèĈædIJèōŁéŰ  
 resp.json iijNéĈcäzLāŕsāijZäĭŰāLŕJSONæāijāijRçŽĐāŠ■āžTāĖĖāōzāĀĈ

äyNéĬcèŁZäyŁçd'žäĭNāĹſçTĭ requests äžŠāŕSètūäyÄäyĭHEAD-  
 èŕūæšĈiijNāžūāzŌāŠ■āžTäy■æŕŔāŕŰāĜžäyÄäžZHTTPād'tæTŕæ■ōçŽĐā■ŰæōiijZ

```

import requests

resp = requests.head('http://www.python.org/index.html')

```

```
status = resp.status_code
last_modified = resp.headers['last-modified']
content_type = resp.headers['content-type']
content_length = resp.headers['content-length']
```

äýÑéÍcæÝřäýÄäýläl'çTlrequestséÄŽèŁĞăšžæIJñèóđ'érAçŽžâ;TPypicŽDä;Nă■ŘiijŽ

```
import requests

resp = requests.get('http://pypi.python.org/pypi?action=login',
                    auth=('user', 'password'))
```

äýÑéÍcæÝřäýÄäýläl'çTlrequestsârEHTTP cookiesäzÖäýÄäýlêrûæšĆaijæÄŠăLřăRëäýÄäýlçŽDä;Nă■

```
import requests

# First request
resp1 = requests.get(url)
...

# Second requests with cookies received on first requests
resp2 = requests.get(url, cookies=resp1.cookies)
```

æIJăĀŔŌă;EăžúéíđæIJÄäý■éĜ■èēAçŽDäýÄäýlă;Nă■RăYřçTlrequestsäýLăijăăEĚăőžiiJŽ

```
import requests
url = 'http://httpbin.org/post'
files = { 'file': ('data.csv', open('data.csv', 'rb')) }

r = requests.post(url, files=files)
```

## èóìèőž

ăržăžŎçIJšçŽDă;ŁçőĂă■THTTTPăóçæŁuçnrăžčçăAiiJŇçTlăEĚç;őçŽD urllib  
æłăăiŮëÄŽăýýăřšèùšăđ'šăžEăĀĆă;EăYřiiJŇăēĆăđIJă;ăèēAăĂŽçŽDăý■ăžĚăžĚăŔăYřçőĂă■TçŽDGETæŁ  
requestsăđ'ğăY;èžñæLŇçŽDăŮăĂŽăžEăĀĆ

ă;ŇăēĆriJŇăēĆăđIJă;ăăEșăőŽăiŽăŇĂă;ŁçTlăăĞăĜEçŽDçłŇăžŔăžŠèĂŇăý■èĂĈèZŠăČŔ  
requests èŁŽăăüçŽDçñăýL'æŮžăžšiiJŇéĆăžŁăžšèőýăřšăý■ă;Ůăý■ă;ŁçTlăžTăšĆçŽD  
http.client æłăăiŮăĬăăđčŎřëĜłăűšçŽDăžčçăAăĀĆăřTăŮžèŕ'iiJŇăýŇéÍcçŽDăžčçăAăšTçđ'žăžEăēĆă

```
from http.client import HTTPConnection
from urllib import parse

c = HTTPConnection('www.python.org', 80)
c.request('HEAD', '/index.html')
resp = c.getresponse()

print('Status', resp.status)
```



```
for name, value in resp.getheaders():
    print(name, value)
```

aŔNæauåIJrijNæCædIJafĖĖazcijŮaFZæul'arŁläzččŘĚãĀAèöd'ērAãĀAcookiesäzëaRŁäĚüazŮäyĀäz  
 urllibårsäŸ;ă;ŮcL'zålŋåLŋåL■åŠNåTråŮëãĀCærTæŮžert'ijNäyNélcèfZäyłcd'žă;ŮåodçŮŖåIJPython

```
import urllib.request

auth = urllib.request.HTTPBasicAuthHandler()
auth.add_password('pypi', 'http://pypi.python.org', 'username',
    ↪ 'password')
opener = urllib.request.build_opener(auth)

r = urllib.request.Request('http://pypi.python.org/pypi?
    ↪ :action=login')
u = opener.open(r)
resp = u.read()

# From here. You can access more pages using opener
...
```

ăġęŻ;èřťġġŃæL'ĂæIJL'čŽĐěřŽăžZæŞ■ă;IJăIJĲ  
 ăžŞăy■éČ;ăRŲă;ŲćőĂă■ŲčŽĐăđ'ŽăĂĆ

requests

aUJaijAaRSefGcInäy■etNerTHTTpaócæLüçnřazčcāAāyŷāyŷæYřaŁŁāzd'ažžæšōāyğçŽDriiŃaŽāāyžæL  
 //httpbin.orgiijL'āĀĆēfZāyłçnŽčČžaijŽæŌēāTūāRŠāGžčŽDērūāsčĪijŃçĐūāRŌāžēJSONçŽDā;ćaijRāřEçŽy

```
>>> import requests
>>> r = requests.get('http://httpbin.org/get?name=Dave&n=37',
...     headers = { 'User-agent': 'goaway/1.0' })
>>> resp = r.json
>>> resp['headers']
{'User-Agent': 'goaway/1.0', 'Content-Length': '', 'Content-Type': '
↪',
'Accept-Encoding': 'gzip, deflate, compress', 'Connection':
'keep-alive', 'Host': 'httpbin.org', 'Accept': '*//*'}
>>> resp['args']
{'name': 'Dave', 'n': '37'}
>>>
```

aIJlëeAaRÑäyÄäyİçIJ\$■ççŽĐñŽçĆžèfŽeaÑāzd'āžŠaL'■rijÑāĖĹaIJ
 httpbin.org  
 èfŽæuüçŽĐç;ŠćñŽäyĹaAŽāōđétÑāyŷāyŷæYřāRřāRŮčŽĐāĹđæšTāĀĆārd'āĖŷæYřā;ŠæĹSāžñēīcārž3æñaçŽ.

request requests ælǣaIŪčŽǣaŪĠæačīijLhttp://docs.python-requests.org)èr' léGRå; LéŋYīijLāleçŽ; èrt' ærTāIJełēZçš■çš■čŽǣyÆełČčŽDçrĠāzĒäy■æL' ÄæRĠä; ŽčŽǣāzā; Tāfæ

```
from socketserver import StreamRequestHandler, TCPServer

class EchoHandler(StreamRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
```

```

# self.rfile is a file-like object for reading
for line in self.rfile:
    # self.wfile is a file-like object for writing
    self.wfile.write(line)

if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()

```

## èõlèõž

socketserver aRfrazèèòl' æLŠaznāĹLāōžæYŞçŽDāLŽāžžçóĀā■TçŽDTCpæIJ■āLāāZlāĀĆ  
 äjEæYřijNā;æIJĀèçAæşlæĐRçŽDæYřijNézYèød' æČĚāEřāyNefŽçg■æIJ■āLāāZlāYřā■TçžĚčlNçŽDřijNāy  
 æČædIJā;æČşad' DçŘĚād' ŽāylāōçæLūçnrřijNāRfrazēāLlāgNāNŪāyĀāyĹ  
 ForkingTCPServer æLŪèĀĚæYř ThreadingTCPServer āřzèsāāĀĆä;NāçCřijŽ

```

from socketserver import ThreadingTCPServer

if __name__ == '__main__':
    serv = ThreadingTCPServer(('', 20000), EchoHandler)
    serv.serve_forever()

```

ä;ĚçTlforkæLŪçžĚčlNæIJ■āLāāZlāIJLāylæ;IJāIJléŪōécYāřsæYřāōČāžnāijŽāyžæfRāylāōçæLūçnrèĚdæ  
 çTšāžŌāōçæLūçnrèĚdæŌēæTřæYřæşæIJL'éŽŘāLūçŽDřijNāZāæ■d' āyĀāylæAūæĐRçŽDžSāōçāRfrazēāRŇ

æČædIJā;æNĚāfČèĚŽāyléŪōécYřijNā;āāRfrazēāLŽāžžāyĀāyléçDāĚLāLĚēĚ■ad' gārRçŽDāūēā;IJçžĚç  
 ä;āāĚLāLŽāžžāyĀāylæŽōéĀŽçŽDēlDçžĚčlNæIJ■āLāāZlřijNçDūāRŌāIJāyĀāylçžĚčlNæşāy■ā;ĚçTl  
 serve\_forever() æŪzæşTælēāRřāLlāōČāžnāĀĆ

```

if __name__ == '__main__':
    from threading import Thread
    NWORKERS = 16
    serv = TCPServer(('', 20000), EchoHandler)
    for n in range(NWORKERS):
        t = Thread(target=serv.serve_forever)
        t.daemon = True
        t.start()
    serv.serve_forever()

```

āyĀēLāælēççřijNāyĀāyĹ TCPServer āIJlāōdā;NāNŪçŽDæŪūāĀŽāijŽçzSāōŽāžúæĚĀæt'zçŽyāžTçŽ  
 socket āĀĆ āy■æĚřijNāIJLæŪūāĀŽā;æČşēĀŽēĚĚōç;ç;ōæşŘāžŽēĀLéqāāŌžèřČæTř' āžTāyNçŽD  
 socket' řijNāRfrazèèç;ç;ōāŘČæTř bind\_and\_activate=False āĀĆæČāyNřijŽ

```

if __name__ == '__main__':
    serv = TCPServer(('', 20000), EchoHandler, bind_and_
    ↪activate=False)
    # Set up various socket options
    serv.socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,
    ↪True)

```

```
# Bind and activate
serv.server_bind()
serv.server_activate()
serv.serve_forever()
```

äYlÉlčŽD socket éĀLéazæYřäYÄäylÉldäyYæŽóéA■čŽDéĚ■;óéazijNăŏČăĖAèöyæIJ■āLāZlÉĜ■  
čTšăŽŎèAècncZŘäyY;ŁçTlĀLřijNăŏČècnăTĭ;č;ŏĀLřçšZăRŸéĜRäy■ijNăRřazèçZřæŎèĀIJl  
TCPServer äYlÉlčèö;č;ŏĀĀČ āIJlăŏdă;NăNŮāIJ■āLāZlčŽDæUŮăĀŽăŎzèö;č;ŏăŏČčŽDăĀijijNăçCăyN

```
if __name__ == '__main__':
    TCPServer.allow_reuse_address = True
    serv = TCPServer(('', 20000), EchoHandler)
    serv.serve_forever()
```

aIJläYLeÍcđ'žä;Näy■iijNæŁŚāznæijTčd'žāžEäyđ'čg■äy■aRŇčŽĐad'DčŘEāZlāšžčsziijŁ  
 BaseRequestHandler      āŠŇ      StreamRequestHandler      iijLāĀĆ  
 StreamRequestHandler æŽt'aŁacAæt'zcĆziiNēČ;eĀŽefGēō;č;ōāEūāzŪčŽĐčszāRŸēGRæİcæŤræNā

```
import socket

class EchoHandler(StreamRequestHandler):
    # Optional settings (defaults shown)
    timeout = 5 # Timeout on all socket_
    ↪operations
    rbufsize = -1 # Read buffer size
    wbufsize = 0 # Write buffer size
    disable_nagle_algorithm = False # Sets TCP_NODELAY socket_
    ↪option
    def handle(self):
        print('Got connection from', self.client_address)
        try:
            for line in self.rfile:
                # self.wfile is a file-like object for writing
                self.wfile.write(line)
        except socket.timeout:
            print('Timed out!')
```

æIJÅaRŌijN̄eƒȲeIJÅēeĀæʃl̄æDR̄čZ̄D̄æȲr̄aũl̄ad'ḡeC̄l̄aLEPython̄cZ̄D̄enȲāsC̄ç;Š̄çzIJĀlālŪijL̄ærT̄āeC̄l̄  
RPC̄ç■L̄t̄ijL̄éC̄j;æȲr̄az̄z̄çn̄N̄aĪJl̄ socketserver āL̄š̄eC̄j;az̄N̄äȲL̄āAC̄  
az̄š̄ārs̄āeȲr̄ēr̄t̄ijN̄çZ̄t̄ æŌ̄ēāj;ƒçT̄l̄ socket āz̄š̄āĪēāōd̄çŌ̄r̄æIJĀāL̄āāZ̄l̄az̄š̄āz̄ūäȳæȲr̄āL̄éZ̄j;ā̄AC̄  
äyN̄éĪc̄āeȲr̄äȲĀäȳl̄āj;ƒçT̄l̄ socket çZ̄t̄ æŌ̄ēçijŪçĪN̄āōd̄çŌ̄r̄çZ̄D̄äȲĀäȳl̄æIJĀāL̄āāZ̄l̄çōĀā■T̄ä;N̄ā■R̄ijJ̄Z̄

```
from socket import socket, AF_INET, SOCK_STREAM

def echo_handler(address, client_sock):
    print('Got connection from {}'.format(address))
    while True:
        msg = client_sock.recv(8192)
        if not msg:
            break
```

```

        client_sock.sendall(msg)
    client_sock.close()

def echo_server(address, backlog=5):
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(address)
    sock.listen(backlog)
    while True:
        client_sock, client_addr = sock.accept()
        echo_handler(client_addr, client_sock)

if __name__ == '__main__':
    echo_server('', 20000)

```

## 13.3 11.3 UDPæI■åŁaŻÍ

### éŮóécŸ

äjäæČšáodčŎřäyÄäyİaşžžŎUDPæI■åŁaŻÍæİäyŎáoćæŁúçńréĂŽăŁaãĂĆ

### èğcăEşæŮzæąŁ

èùšTCPäyĂæăüijŃUDPæI■åŁaŻÍläžšăŔřäžééĂŽèŁĠăŁçŦÍ socketserver  
 äžŞăİŁăőzæŸŞçŽĐěcňăĹŽăžžăĂĆ äĹŃăĉĈijŃäyŃéĹćæŸřäyÄäyİčőĂăŦçŽĐæŮúéŮŕ æI■åŁaŻÍliijŽ

```

from socketserver import BaseRequestHandler, UDPServer
import time

class TimeHandler(BaseRequestHandler):
    def handle(self):
        print('Got connection from', self.client_address)
        # Get message and client socket
        msg, sock = self.request
        resp = time.ctime()
        sock.sendto(resp.encode('ascii'), self.client_address)

if __name__ == '__main__':
    serv = UDPServer('', 20000), TimeHandler)
    serv.serve_forever()

```

èùšăžŃăĹăäyĂæăüijŃăăĹăőŽăžŦăyÄäyİăodčŎř handle()  
 çĹžăőŁăŮžæşŦçŽĐçşzĭijŃäyžăőćæŁúçńréĹđæŎěæI■åŁaãĂĆ èŁŽäyİčşžçŽĐ  
 request áşđæĂğæŸřäyÄäyİăŃĚăŔňăžĚæŦŕæ■őæĹăăŞŃăžŦăşĆsock-  
 etăŕžèşăçŽĐăĚČçžĐăĂĆclient\_address âŃĚăŔňăžĚăőćæŁúçńŕăĹŕăĹăăĂĆ

æĹŠăžŃăİăæŦŃerŦäyŃéŁŽäyİæI■åŁaŻÍliijŃéçŮăĹĹèŁēŔëăŃăőĈijŃçĐăăŔŎăĹŞăijĂăŔëăđ ŮäyÄäyH

```
>>> from socket import socket, AF_INET, SOCK_DGRAM
>>> s = socket(AF_INET, SOCK_DGRAM)
>>> s.sendto(b'', ('localhost', 20000))
0
>>> s.recvfrom(8192)
(b'Wed Aug 15 20:35:08 2012', ('127.0.0.1', 20000))
>>>
```

## ëóíëóž

äyÄäyłaËyādNçŽDUDPæIJ■āLāāZÍæŎœTūāLrē;çŽDæTṛæ■ōæLē(æūLæAṛ)āŠNāóœLūçnrāIJṛāIĀā  
 āóČēçAçzŽāóœLūçnrāZḍāRŠäyÄäyłaTṛæ■ōæLēāĀČārzāžŎæTṛæ■ōæLēçŽDāijāéĀAīijN  
 ā;āāžTēřēā;ççTīsocketçŽD sendto() āŠN recvfrom() æŬzæsTāĀČ  
 āṛ;çōāijāçzççŽD send() āŠN recv() āžšāRřāzēē;āLṛāRŇæāūçŽDæTṼædIJīijN  
 ā;EæYṛāL■ēlççŽDäyḍ'äyłaŬzæsTṛāzāžŎUDPēfḍæŎēēĀNēlĀæŽt'æŽóéA■āĀČ

çTṣāžŎæšæIJL'āžTṣāČçŽDēfḍæŎēīijNUPDæIJ■āLāāZÍçŽyārzāžŎTCPæIJ■āLāāZÍæIēēōšāóḍçŎřēṭūāI  
 äy■ēfGīijNUPDāḍ'IçTṣæYṛäy■āRřēlāçŽDīijLāZāyžēĀŽāfæšæIJL'āžžçñNēfḍæŎēīijNæūLæAṛāRřēČ;äy  
 āŽāæ■d'ēIJĀēçAçTṣā;āēGīāūsæIēāEšāóŽēřæĀŎæāūāḍ'DçRĒäyçāḍ'sæūLæAṛçŽDæČēāEṭāĀČēfZäyIāūsçz  
 äy■ēfGēĀŽäyæIēērt'īijNāçCædIJāRřēlāæĀgārzāžŎā;āçlNāžRā;LéG■ēçAīijNā;āēIJĀēçAāĀšāL'āžŎāžRāI  
 UDPēĀŽäyçēççTīāIJlēCçāžZārzāžŎāRřēlāāijāē;çēçAæšCäy■æYṛā;LēnYçŽDāIJzāRĹāĀČā;NāçCīijNāIJlā  
 æŬāēIJĀēfTāZḍæAçāḍ'■äyçāḍ'sçŽDæTṛæ■ōāNēīijLçlNāžRāRlēIJĀçōĀā■TçŽDāf;çTēāóCāžçççç■āRŠāI

UDPSever çšzæYṛā■TçžççlNçŽDīijNāžšārsæYṛērt'äyĀæñāāRlēČ;äyžäyÄäyIāóœLūçnrēfḍæŎēæIJ■  
 āóḍēŽĒä;ççTīäy■īijNēfZäyIæŬāēōžæYṛārzāžŎUDPēfYæYṛTCPēČ;äy■æYṛāžĀāžLāḍ'gēŬōēçYāĀČ  
 āçCædIJā;āæČšēçAāžūāRŠæš■ā;IJīijNāRřāzēāōḍā;NāNŬäyÄäyI ForkingUDPSever  
 æLŬ ThreadingUDPSever āřzēšāijŽ

```
from socketserver import ThreadingUDPServer

if __name__ == '__main__':
    serv = ThreadingUDPServer(('',20000), TimeHandler)
    serv.serve_forever()
```

çŽt'æŎēā;ççTī socket æIēāóḍçŎřäyÄäyIUDPæIJ■āLāāZÍāžšäy■ēŽ;īijNāyNēlçæYṛäyÄäyIā;Nā■RīijŽ

```
from socket import socket, AF_INET, SOCK_DGRAM
import time

def time_server(address):
    sock = socket(AF_INET, SOCK_DGRAM)
    sock.bind(address)
    while True:
        msg, addr = sock.recvfrom(8192)
        print('Got message from', addr)
        resp = time.ctime()
        sock.sendto(resp.encode('ascii'), addr)
```

```
if __name__ == '__main__':
    time_server(('', 20000))
```

## 13.4 11.4 éĀŽèĚĠCIDRāIJrāiĀçTšæĹŘárzážTčŽĎIPāIJrāiĀéŽĚ

### éŮóéčŸ

äĵäæIJLäyĀäyĪCIDRçĵŠçzIJāIJrāiĀæfTæČâĀIJ123.45.67.89/27āĀiĵNäĵäæČšārĒāĒūèĵñæ■cæĹŘāóČa  
iĵĹæfTæČiĵNāĀIJ123.45.67.64āĀĪ, āĀIJ123.45.67.65āĀĪ, āĀç, āĀIJ123.45.67.95āĀĪiĵL'

### èğčāĒşæŮžæāĹ

āŘrāžēäĵčTĪ ipaddress æĹāiŮāĹĹāóžæŸŞçŽĎāóđçŎřèfŽæăŭçŽĎèóaçŏŮāĀČāĹNāçČiĵŽ

```
>>> import ipaddress
>>> net = ipaddress.ip_network('123.45.67.64/27')
>>> net
IPv4Network('123.45.67.64/27')
>>> for a in net:
...     print(a)
...
123.45.67.64
123.45.67.65
123.45.67.66
123.45.67.67
123.45.67.68
...
123.45.67.95
>>>

>>> net6 = ipaddress.ip_network('12:3456:78:90ab:cd:ef01:23:30/125')
>>> net6
IPv6Network('12:3456:78:90ab:cd:ef01:23:30/125')
>>> for a in net6:
...     print(a)
...
12:3456:78:90ab:cd:ef01:23:30
12:3456:78:90ab:cd:ef01:23:31
12:3456:78:90ab:cd:ef01:23:32
12:3456:78:90ab:cd:ef01:23:33
12:3456:78:90ab:cd:ef01:23:34
12:3456:78:90ab:cd:ef01:23:35
12:3456:78:90ab:cd:ef01:23:36
12:3456:78:90ab:cd:ef01:23:37
>>>
```

Network äžšāĒĀèőyāČŘæTřçzĎäyĀæăŭçŽĎçt'cāiĵTāRŮāĀiĵiĵNäĹNāçČiĵŽ

```
>>> net.num_addresses
32
>>> net[0]
IPv4Address('123.45.67.64')
>>> net[1]
IPv4Address('123.45.67.65')
>>> net[-1]
IPv4Address('123.45.67.95')
>>> net[-2]
IPv4Address('123.45.67.94')
>>>
```

āRēād' ŪīijNā;āēfYāRrāzēæL'gēāNç;ŚçzIJæLŖāŚYæčĀæšēāzNçszçŽDæS■ā;IJijŽ

```
>>> a = ipaddress.ip_address('123.45.67.69')
>>> a in net
True
>>> b = ipaddress.ip_address('123.45.67.123')
>>> b in net
False
>>>
```

äyĀäyHPāIJrāiĀāSŇç;ŚçzIJāIJrāiĀēČ;éĀŽēfGāyĀäyHPæŌēāRčælēæNĜāōŽīijNā;NāēĆīijŽ

```
>>> inet = ipaddress.ip_interface('123.45.67.73/27')
>>> inet.network
IPv4Network('123.45.67.64/27')
>>> inet.ip
IPv4Address('123.45.67.73')
>>>
```

## èõléõž

ipaddress ælāāiŪæIJL'ā;Lād'ŽçszāRrāzēæāçd'žIPāIJrāiĀāĀAç;ŚçzIJāSŇæŌēāRčāĀĆ  
 ā;Šā;āēIJĀēæAæS■ā;IJç;ŚçzIJāIJrāiĀīijLærTāēCègčædRāĀAæL'Sā■rāĀAēfNērAç■L'īijL'çŽDæŪūāĀŽāijŽā  
 èæAæslæDRçŽDæYīijNīipaddress ælāāiŪēūšāEūāzŪāyĀāzZāSŇç;ŚçzIJçŽyāĒşçŽDælāāiŪærTāēĆ  
 socket āžŠāžd'ēZEā;LārSāĀĆ æL'ĀāzēīijNā;āāy■ēČ;ā;fçTĪ IPv4Address  
 çŽDāōdā;NālēāzçæŽfäyĀäyĪāIJrāiĀā■ŪçņēāyīijNā;āēçŪāĒLā;ŪæY;āijRçŽDā;fçTĪ  
 str() è;ñæ■cāōČāĀĆä;NāēĆīijŽ

```
>>> a = ipaddress.ip_address('127.0.0.1')
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.connect((a, 8080))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't convert 'IPv4Address' object to str implicitly
```



```
>>> s.connect((str(a), 8080))
>>>
```

# æẏt'ad'žčẏyăĔșăĔĖăőőĩĩjÑěŕuáŔĆĉĤĈ An Introduction to the ipaddress Module

## 13.5 11.5 áĹžǻzǻžǻŸĂăŸłçőĂă■ȚçŽǾRESTæŎěǻŘč

éŮőécŸ

ä;äăČsä;ĚčŤlāvÄäyļcōÅā■ȚÇŽDREStæŎěāŘćéĂžēfĠç;ŚczIjēłIJćÍNæŎǵáLúăŁŮëóĲēŮöā;ăčŽďăžŤ

èğčǎEşæŮźæǻŁ

ædĐāzžāyĀäylRESTēĀēāijčŽĐæŌēāRcæIJĀçōĀā■TçŽĐæŪzæşTæYřāLZāzžāyĀäylāşzāžŌWSGIæā  
3333iijL'cŽĐā;ĹārRcŽĐāzSīijNāyNéIcæYřāyĀäylā;Nā■RīijŽ

```
# resty.py

import cgi

def notfound_404(envIRON, start_response):
    start_response('404 Not Found', [ ('Content-type', 'text/plain
↪') ])
    return [b'Not Found']

class PathDispatcher:
    def __init__(self):
        self.pathmap = { }

    def __call__(self, environ, start_response):
        path = environ['PATH_INFO']
        params = cgi.FieldStorage(environ['wsgi.input'],
                                   environ=environ)
        method = environ['REQUEST_METHOD'].lower()
        environ['params'] = { key: params.getvalue(key) for key in_
↪params }
        handler = self.pathmap.get((method,path), notfound_404)
        return handler(environ, start_response)

    def register(self, method, path, function):
        self.pathmap[method.lower(), path] = function
        return function
```

äyžāẸä;ŁçTlēfZävlerČāẸāZlīijŃā;āāRlēIJĀēēAçijŪāEŻäy■āRŃçŽDād'DçRĒāZlīijŃārśāČRäyNeİcēŁ

```
import time

_hello_resp = '''\
```

```

<html>
  <head>
    <title>Hello {name}</title>
  </head>
  <body>
    <h1>Hello {name}!</h1>
  </body>
</html>'''

def hello_world(environ, start_response):
    start_response('200 OK', [ ('Content-type', 'text/html') ])
    params = environ['params']
    resp = _hello_resp.format(name=params.get('name'))
    yield resp.encode('utf-8')

_localtime_resp = '''\
<?xml version="1.0"?>
<time>
  <year>{t.tm_year}</year>
  <month>{t.tm_mon}</month>
  <day>{t.tm_mday}</day>
  <hour>{t.tm_hour}</hour>
  <minute>{t.tm_min}</minute>
  <second>{t.tm_sec}</second>
</time>'''

def localtime(environ, start_response):
    start_response('200 OK', [ ('Content-type', 'application/xml') ]
    ↪)
    resp = _localtime_resp.format(t=time.localtime())
    yield resp.encode('utf-8')

if __name__ == '__main__':
    from resty import PathDispatcher
    from wsgiref.simple_server import make_server

    # Create the dispatcher and register functions
    dispatcher = PathDispatcher()
    dispatcher.register('GET', '/hello', hello_world)
    dispatcher.register('GET', '/localtime', localtime)

    # Launch a basic server
    httpd = make_server('', 8080, dispatcher)
    print('Serving on port 8080...')
    httpd.serve_forever()

```

ẽeAætNërTäyNèfZäyIæI■āLāāZlíjNā;āāRřazëā;fçTlāyÄäyIætRëğLāZlæLŮ urllib  
 åŠNăőCăzd'ăzŠăĂCă;NăeĆrijŽ

```

>>> u = urlopen('http://localhost:8080/hello?name=Guido')
>>> print(u.read().decode('utf-8'))
<html>
  <head>
    <title>Hello Guido</title>
  </head>
  <body>
    <h1>Hello Guido!</h1>
  </body>
</html>

>>> u = urlopen('http://localhost:8080/localtime')
>>> print(u.read().decode('utf-8'))
<?xml version="1.0"?>
<time>
  <year>2012</year>
  <month>11</month>
  <day>24</day>
  <hour>14</hour>
  <minute>49</minute>
  <second>17</second>
</time>
>>>

```

## ëõlëõž

ãIJłijŮãEŽRESTæŌëãRčæŮüijÑéĂžäyyéČ;æÝřæIJ■ãŁažžŌæŽóéĂŽčŽDHTTPèřúæśCãĂČă;EæÝřè  
 èŁŽäžŽæŤřæ■öäžëãRĎčĝ■æăĜăĖæăijăijRčijŮčăAĭijÑæřŤăĕČXMLăĂAJSONæŁŮCSVăĂČ  
 år;čōačłNăžRčIJNăyŁăŌžă;ŁčŏĂă■ŤijNă;EæÝřăžèèŁŽčĝ■æŮžăijRæŘRă;ŽčŽDAPIăržăžŌă;Łăđ'ŽăžŤčŤł

ă;NăĕČijNěŤŁæIJšèŁŘèaNčŽĎčłNăžRăRřèČ;ăijŽă;ŁčŤłăyĂăyĤREST  
 APIæłăŏđčŌřčŽŚæŌĝæŁŮĕŁæŮ■ăĂČăđ'ĝæŤřæ■ŏăžŤčŤłčłNăžRăRřăžëă;ŁčŤłRESTæłăĕđDăžžăyĂăyŁæ'  
 RESTèŁŸĕČ;čŤłæłăĕŌĝăŁŮčăňăžüèŏ;ăđ'ĜæřŤăĕČæIJžăŽłăžžăĂĂăijăæĎšăŽłăĂĂăüëăŌČæŁŮčAřæşăăĂČ  
 æŽ'ĕĜ■ĕĖAčŽĎæÝřijNREST APIăüşčžRĕčňăđ'ĝĕĜRăŏčæŁŮčňřčijŮčłNčŌřăčČæŁ'ĂæŤřæŤăijNăřŤăĕČ-  
 Javascript, Android, iOS■Ł'ăĂČăŽăæ■đ'ijNăłŁ'čŤłĕŁŽčĝ■æŌëãRčăRřăžèèŏł'ă;ăăijĂăRŚăĜžæŽŤ'ăŁăăđ'■æ

ăyžăEăŏđčŌřăyĂăyŁčŏĂă■ŤčŽĎRESTæŌëãRčrijNă;ăăRłĕIJĂĕŏł'ă;ăčŽĎčłNăžRăžččăAæžăĕŮşPython  
 WSGIĕčňăăĜăĖEăžŞæŤřæŤăijNăRŤæŮŮăžşĕčňčžłăđ'ĝĕČłăŁĖčňňăyŁ'æŮžwebăĕEăĎŮæŤřæŤăăĂČ  
 ăŽăæ■đ'ijNăĕČăđIJă;ăčŽĎăžččăAĕAŤă;łĕŁŽăyŁæăĜăĖĭijNăIJłăRŌĕłčŽĎă;ŁčŤłĕŁĜčłNăy■ăřśăijŽæŽŤ'ăŁ

ãIJłWSGIăy■ijNă;ăăRřăžëăČRăyNĕłĕĕŁæăüčžĕăŏŽčŽĎæŮžăijRăžëăyĂăyŁăRřĕřČčŤłăržĕśăă;čăijRăĕł

```

import cgi

def wsgi_app(environ, start_response):
    pass

```

environ      áśđæĂĝæÝřăyĂăyŁă■ŮăĖyijNăNĕăRňăžĖăžŌwebæIJ■ăŁăăŽłăĕČA-  
 pachełăRČĕĕĂČInternet      RFC      3875]æŘRă;ŽčŽDCGIæŌëãRčăy■ĕŌŮăRŮčŽĎăĂijăĂČ  
 ĕĖAărĖĕŁŽăžŽăy■ăRŤčŽĎăĂijăRŘăRŮăĜžæłĕĭijNă;ăăRřăžëăČRĕŁŽăžŁĕŁŽæăŮăĖŽijŽ

```
def wsgi_app(environ, start_response):
    method = environ['REQUEST_METHOD']
    path = environ['PATH_INFO']
    # Parse the query parameters
    params = cgi.FieldStorage(environ['wsgi.input'],
    ↪environ=environ)
```

```
    æĽSāznāsTçd'žāžEäyÄāzZāyÿëgAçŽDāĀijāĀCenviron['REQUEST_METHOD']
    äzçēāĭērūæsČçśzādNāeČGETāĀAPOSTāĀAHEADç■ĽāĀC    environ['PATH_INFO']
    ēāĭçd'žēcnērūæsČetĎāzRçŽDēurāĭĎāĀC    ēřČĤĬ    cgi.FieldStorage()
    āRřāzēāzŎērūæsCāy■æRŘāRŮæšēērčāRČæTřāzūārEāōCāznæTĭāĖēäyÄäyĭçśzā■ŮāĖyārřzēšāy■āzēāĭŁāRŎ
    start_response āRČæTřæYřāyÄäyĭāyžāžEāĽĭāgNāNŮāyÄäyĭērūæsCārřzēšāēĀNāŁĖēāzēcnērČçTĭ
    çññāyÄäyĭāRČæTřæYřēŁTāŽdçŽDHTTPçĽūæĀĀāĀijĭjNçññāzNāyĭāRČæTřæYřāyÄäyĭ(āR■,āĀij)āĖČçzDā
```

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
```

```
    äyžāžEēŁTāŽdæTřæ■ōĭjNāyÄäyĭWSGIçĭNāzRāŁĖēāzēŁTāŽdāyÄäyĭā■ŮēŁČā■ŮçņēäyšāzRāĽŮāĀČāF
```

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
    resp = []
    resp.append(b'Hello World\n')
    resp.append(b'Goodbye!\n')
    return resp
```

```
    æĽŮēĀĖĭjNāĭæŁYāRřāzēāĭŁçTĬ yield ĭjŽ
```

```
def wsgi_app(environ, start_response):
    pass
    start_response('200 OK', [('Content-type', 'text/plain')])
    yield b'Hello World\n'
    yield b'Goodbye!\n'
```

```
    ēŁŽēGŇēēĀĭjžērČçŽDāyĀçČzæYřæĪĀāRŎēŁTāŽdçŽDāŁĖēāzæYřā■ŮēŁČā■ŮçņēäyšāĀČāēČæĎĪēŁ
    āĭšçĎŮĭjNāzūæšāēĪĽēēĀæsČāĭæŁTāŽdçŽDāyĀāōZæYřæŮĜæĪNĭjNāĭāāRřāzēāĭŁēĭzæĭçŽDçĭjŮāĖŽāy
    āřĭçōāWSGIçĭNāzRéĀŽāyÿēcnāōŽāzĽæĽRāyÄäyĭāĜĭæTřĭjNāy■ēŁĜāĭāāzšāRřāzēāĭŁçTĬçśzāōđāĭNāĭ
    __call__() æŮzæšTāĀČāĭNāēČĭjŽ
```

```
class WSGIApplication:
    def __init__(self):
        ...
    def __call__(self, environ, start_response)
        ...
```

```
    æĽSāznāūšçzRāĪĭāyĽēĭčāĭŁçTĭēŁŽçg■æĽĀæĪřāĽZāžž    PathDispatcher    çśzāĀČ
    ēŁŽāyĭāĽĒāRŠāZĭāzĖāzĖĀRĭæYřçōāçRĖāyÄäyĭā■ŮāĖyĭjNārĖ(æŮzæšTĭ,ēurāĭĎ)ārřæYřāārĎāĽřād'ĎçRĖāZĭ
```

ā;ŠāyĀāyġēfūāēšCālŕāēlēāUūiijNāōČčŽDæŪzæšTāŠNēūfā;DēcñāRŔāRŪāGžæġēiijNčDūāRŔōēcñāLēāRŠāġ  
āRēād'ŪiijNāzā;TæšēērcāRŸēGRāijZēcñēgčædŔāRŔōæT;ālŕāyĀāyġāUāĒyāyūiijNāzē  
environ['params'] ā;čāijRāYāCġāĀC āRŔōēīcēfZāyġāēēīd'ād'ġāyŷēgAīijNāēL'Āāzēāzžēōōā;āāIJāLē  
ā;fçTġāLēāRŠāZġcŽDæŪūāĀZiijNā;āāRġēIJĀçōĀāTçŽDāLZāzžāyĀāyġāōđā;NīijNčDūāRŔōēĀZēfGāōČāš  
çijŪāEŽēfZāzŽāG;æTŕāžTēēēēŪĒçžgçōĀāTāžEīijNāRġēAā;āēAġā;ġ  
start\_response() āG;æTŕçŽDçijŪāEŽēgDāLZiijNāzūāyTæIJĀāRŔōēfTāZđāUēLČāUçņēāyšāšāRŕā

ā;ŠçijŪāEŽēfZçgāG;æTŕçŽDæŪūāĀZēfŸēIJĀæšġæDŔçŽDāyĀçČzāŕsæŸŕāŕzāžŌāUçņēāyšāēġāēīfç  
æšāāzžæDēæDŔāEŽēCççgāLŕād'DæūūāRġLçġĀ print() āG;æTŕ āĀAXM-  
LāŠNād'gēGRæāijāijRāNŪæŠā;IJçŽDāzççāAāĀC æġSāznāyLēīcā;fçTġāzEāyL'āijTāRūāNĒāRŕñçŽDēcDāē  
ēfZçgāæŪzāijRçŽDāRŕāzēēōl'æġSāznā;LāōzæŸŠçŽDāIJāzēāRŔōāfōæTzē;ŠāGžæāijāijR(āRġēIJĀēēAāēōæ

æIJĀāRŔōiijNā;fçTġWSGġēfŸæIJL'āyĀāyġā;LēGēēAçŽDēCġāLēāŕsæŸŕæšāæIJL'āzĀāzġLāIJŕæŪzæŸŕē  
āZāāyžæāGāGĒāŕzāžŌāIJāLāāZġāŠNāæĒāēdūæŸŕāyçñNçŽDīijNā;āāRŕāzēāŕEā;āçŽDçġNāzRæT;āēēāzžā  
æġSāznā;fçTġāyNēīcçŽDāzççāAāēTŕNērTæNērTæIJñēLČāzççāAīijZ

```
if __name__ == '__main__':  
    from wsgiref.simple_server import make_server  
  
    # Create the dispatcher and register functions  
    dispatcher = PathDispatcher()  
    pass  
  
    # Launch a basic server  
    httpd = make_server('', 8080, dispatcher)  
    print('Serving on port 8080...')  
    httpd.serve_forever()
```

āyLēīcāzççāAāLZāzžāžEāyĀāyġçōĀāTçŽDæIJāLāāZġīijNčDūāRŔōā;āāŕsāRŕāzēæġæTŕNērTāyNā;āçŽD  
æIJĀāRŔōiijNā;Šā;āāGĒād'GēfZāyĀāēæL'āsTā;āçŽDçġNāzRçŽDæŪūāĀZiijNā;āāRŕāzēāfōæTzēfZāyġāz

WSGġæIJñēznæŸŕāyĀāyġā;LāŕRçŽDæāGāGĒāĀCāZāæd'āōČāzūæšāæIJL'æRŔā;ZāyĀāzŽēnŸçžgçŽD  
ēfZāzŽā;āēġāūsāōđçŔŕēŭāēīāzšāyēēZ;āĀCāyēēfGāēCādIJā;æēČšēēAæZt'ād'ZçŽDæTŕæNāiijNāRŕāzēē  
WebOb æġŪēĀĒ Paste

## 13.6 11.6 ēĀŽēġXML-RPCāōđçŔŕçōĀāTçŽDēIJçġNērČçTġ

### ēŪōēćŸ

ā;āæČšæL;ālŕāyĀāyġçōĀāTçŽDæŪzāijRāŔzæL'gēāNēēfRēāNāIJġēIJçġNæIJzāZġāyLēīcçŽDPythonçġ

### ēgčāEšæŪzæāġ

āōđçŔŕāyĀāyġēfIJçġNæŪzæšTēŕČçTġçŽDæIJĀçōĀāTæŪzāijRæŸŕā;fçTġXML-  
RPCāĀCāyNēīcæġSāznāēijTçd'žāyĀāyNāyĀāyġāōđçŔŕāzEēTō-  
āĀijāYāCġāLšēČ;çŽDçōĀāTæIJāLāāZġīijZ

```

from xmlrpc.server import SimpleXMLRPCServer

class KeyValueServer:
    _rpc_methods_ = ['get', 'set', 'delete', 'exists', 'keys']
    def __init__(self, address):
        self._data = {}
        self._serv = SimpleXMLRPCServer(address, allow_none=True)
        for name in self._rpc_methods_:
            self._serv.register_function(getattr(self, name))

    def get(self, name):
        return self._data[name]

    def set(self, name, value):
        self._data[name] = value

    def delete(self, name):
        del self._data[name]

    def exists(self, name):
        return name in self._data

    def keys(self):
        return list(self._data)

    def serve_forever(self):
        self._serv.serve_forever()

# Example
if __name__ == '__main__':
    kvserv = KeyValueServer('0.0.0.0', 15000)
    kvserv.serve_forever()

```

äyÑéÍæĹŚāznāzŌäyÄäyġāōcæĹuçñræIJžāŽĹäyĹéÍæĹëèőĹéŰőæIJ■āĹāžĹġijŽ

```

>>> from xmlrpc.client import ServerProxy
>>> s = ServerProxy('http://localhost:15000', allow_none=True)
>>> s.set('foo', 'bar')
>>> s.set('spam', [1, 2, 3])
>>> s.keys()
['spam', 'foo']
>>> s.get('foo')
'bar'
>>> s.get('spam')
[1, 2, 3]
>>> s.delete('spam')
>>> s.exists('spam')
False
>>>

```

## ëõlëõž

XML-RPC āŕŕāzēēōl' æĹŚāznā;ĹāōžæŸŞçŽDæđDēĀāyĀāyĭçōĀā■TçŽDēfIJçĹNērČçTĹæIJ■āĹāāĀCā; ēĀžēfGāōČçŽDæŪzæşT register\_function() æĹæşĹāĒNāG;æTŕiijNçDŭāRŌā;ĤçTĹæŪzæşT serve\_forever() āŕŕāĹĹāōČāĀC āIJĹyĹēĹcæĹŚāznārĒēfŽāžŽæ■ēēd' æT;āIJĹyĀēĭuāĒZāĹŕāyĀāyĭçşž

```
from xmlrpc.server import SimpleXMLRPCServer
def add(x, y):
    return x+y

serv = SimpleXMLRPCServer(('', 15000))
serv.register_function(add)
serv.serve_forever()
```

XML-RPCæŽt' ēIJšāGžæĹççŽDāG;æTŕāŕĹēČ;ēĀČçTĹāžŌēČĹāĹĒæTŕæ■ōçşžādNiiĴNærTæČā■Ūçņēāyş; āŕzāžŌāĒūāžŪçşşādNārşā;ŪēIJĀēçĀāĀŽāžZēčĹād' ŪçŽDāĹşērĹāžĒāĀC ā;NāēČiijNāçCæđIJā;āæČşēĀŽēfG XML-RPC āijāēĀŞāyĀāyĭŕzēşāōđā;NiiĴNāōđēZĒāyĹāŕĹæIJĹ'āžŪçŽD

```
>>> class Point:
...     def __init__(self, x, y):
...         self.x = x
...         self.y = y
...
>>> p = Point(2, 3)
>>> s.set('foo', p)
>>> s.get('foo')
{'x': 2, 'y': 3}
>>>
```

çşzāijijçŽDŕiijNārzāžŌāžNēfZāĹūæTŕæ■ōçŽDād' DçŖĒāžşēuşā;āæČşēşççŽDāy■ād' ĹāyĀæāũŕiijŽ

```
>>> s.set('foo', b'Hello World')
>>> s.get('foo')
<xmlrpc.client.Binary object at 0x10131d410>

>>> _.data
b'Hello World'
>>>
```

āyĀēĹŕæĹēēōšŕiijNā;āāy■āžTērēārĒ XML-RPC æIJ■āĹāžēāĒŕāĒSAPIçŽDæŪzāijŕæŽt' ēIJšāGžæĹēāĀC āŕzāžŌēfZçğ■æČĒāĒŕiijNēĀŽāyāĹĒāyČāijŕāžTçTĹçĹNāžŕāijŽæŸŕāyĀāyĭæŽt' āē;çŽDēĀĹ'æNĹ'āĀC

XML-RPCçŽDāyĀāyĭçijçČzæŸŕāōČçŽDæĀğēČ;āĀC SimpleXMLRPCServer çŽDāōđçŌŕæŸŕā■TçžĤçĹNçŽDŕiijN æĹĀžēāōČāy■ēĀČāŖĹāžŌād' gādNçĹNāžŖiijNār;çōqæĹŚāznāIJĹ1.2ārĹ āŖēād' ŪŕiijNçTşāžŌ XML-RPC āŖĒæĹ'ĀæIJĹ'æTŕæ■ōēČ;āžŖāĹŪāNŪāyžXMLæāijāijŖiijNæĹĀžēāōČāijŽ; ā;ĒæŸŕāōČāžşæIJĹ'āijŸçČzŕiijNēfZçğ■æŪzāijŖçŽDçijŪçāĀŕŕāzēēčnçzĹād' gēČĹāĹĒāĒūāžŪçijŪçĹNēr■ēĹĀ ēĀžēfGā;ĤçTĹēfZçğ■æŪzāijŖiijNāĒūāžŪēr■ēĹĀçŽDāōcæĹūçŕŕçĹNāžŖēČ;ēČ;ēōēēŪōā;āçŽDæIJ■āĹāāĀC

ēŽ;çDŭXML-RPCæIJĹ'ā;Ĺād'ŽçijjççČzŕiijNā;ĒæŸŕāēČæđIJā;āēIJĀēçĀāĒnéĀşæđDāžzāyĀāyĭçōĀā■Tē; æIJĹ'æŪūāĀŽiijNçōĀā■TçŽDæŪzæāĹāŕşāũşçžŖēūşād' şāžĒāĀC

## 13.7 11.7 ĄJlăy■āŔŇçŽĐPythonèġcéĠŁăŽlăzŇéŮt'ăžd'ăžŠ

### éŮóécŸ

ăĵăăJlăy■āŔŇçŽĐæIJžăŽlăyŁéÍcèŁŔèqŇçlĀăd'ŽăyĲPythonèġcéĠŁăŽlăôđăŤŇiĵŇăžŮăyŇæIJŽèČĵăd'šă

### èġcāEşæŮzæąŁ

éĂŽèŁĠăĲçŤl multiprocessing.connection æĴăĲŮăŔŕăžèăŤŁăôžæŸŞçŽĐăôđçŎŕèġcéĠŁăŽlă  
ăyŇéÍcæŸŕăyĂăyĲôĂă■ŤçŽĐăžŤç■ŤæIJ■ăŁăăŽlăŤŇă■ŔiĵŽ

```
from multiprocessing.connection import Listener
import traceback

def echo_client(conn):
    try:
        while True:
            msg = conn.recv()
            conn.send(msg)
    except EOFError:
        print('Connection closed')

def echo_server(address, authkey):
    serv = Listener(address, authkey=authkey)
    while True:
        try:
            client = serv.accept()

            echo_client(client)
        except Exception:
            traceback.print_exc()

echo_server((' ', 25000), authkey=b'peekaboo')
```

çĐŮăŔŎăôcæŁŮçŇŕèŁđæŎcæIJ■ăŁăăŽlăžŮăŔŖŖéĂăăŮŁæĲŕçŽĐçôĂă■Ťçd'žăŤŇiĵŽ

```
>>> from multiprocessing.connection import Client
>>> c = Client(('localhost', 25000), authkey=b'peekaboo')
>>> c.send('hello')
>>> c.recv()
'hello'
>>> c.send(42)
>>> c.recv()
42
>>> c.send([1, 2, 3, 4, 5])
>>> c.recv()
[1, 2, 3, 4, 5]
>>>
```



eùšāžTāsCsocketäy■āRŃçŽDæYřijNæfRäylæúLæAřaijŽāōNæTř'äflā■YřijLæfRäyÄäyléÄŽèfGsend()  
āRēād'ŮřijNæL'ÄæIJL'āržèšāijŽéÄŽèfGpickleāžRāLŮāNŮāĀCāZāæ■d'rijNāžžā;TřāEijāōžpickleçŽDāržèšā

## èóíèőž

çŽōāL'■æIJL'āĹLād'ŽçTlæIēāōđçŎřāRĎçg■æúLæAřaijæē;ŠçŽDāNĒāŠNāG;æTřāžŠřijNæfTāēCZeroMQ  
ā;āēfYæIJL'āRēād'ŮäyĀçg■ēAL'æNl'āršæYřēGlaūsāIJlāžTřāsCsocketāšžçāÄāžNäyLæIēāōđçŎřāyÄäylæúLæ  
ā;EæYřā;āæČšēAçōĀā■TäyĀçČçŽDæŮžæāLřijNéCčāžLēfZæŮūāÄŽ  
multiprocessing.connection āřsæt'čäyLçTlāIJžāžEāĀC  
āžĒāžĒā;fçTlāyÄāžZçōĀā■TçŽDēf■āRēā■šāRřāōđçŎřād'ŽäylēgčēGLāŽlāžNéŮř'çŽDæúLæAřéÄŽāfāĀC

āēČæđIJā;āçŽDēgčēGLāŽlēfRēāNāIJlāRŃäyĀāRřæIJžāŽlāyLēlčřijNéCčāžLā;āāRřāžēā;fçTlāRēād'Ůç  
ēēAæČšā;fçTlāUNIXāššāēŮāŎēā■ŮāIēāLZāžžäyÄäylēfðæŎērijNāRlēIJāçōĀā■TçŽDārEāIJřāIĀæTžāEžäy

```
s = Listener('/tmp/myconn', authkey=b'peekaboo')
```

ēēAæČšā;fçTlāWindowsāŠ;āR■çōāēAŠæIēāLZāžžēfðæŎērijNāRlēIJāČRāyNéíçēfZæāūā;fçTlāyÄäylæ

```
s = Listener(r'\\.\pipe\myconn', authkey=b'peekaboo')
```

äyÄäyléÄŽçTlāGĒāLZæYřijNā;āäy■ēēAā;fçTlā multiprocessing  
æIēāōđçŎřāyÄäylāržād'ŮçŽDāĒāĒsæIJ■āLāāĀC Client() āŠN Listener()  
äy■çŽD authkey āRČæTřçTlæIēēōđ'ērAāRŠēřūēfðæŎēçŽDçžLčřçTlæLūāĀC  
āēČæđIJārEēŠēäy■āržāijŽāžgçTšäyÄäylāijČāyāĀCæ■d'ād'ŮřijNēřēālaIŮæIJĀēĀCāRlçTlæIēāžžčñNéTfē  
āĹNāēČřijNäyd'äylēgčēGLāŽlāžNéŮř'āRřāLlāRŎāřsāijĀāgNāžžčñNēfðæŎēāžūāIJlād'ĎçRĒæšRäyléŮōēčY

āēČæđIJā;āēIJĀēēAāržāžTřāsCēfðæŎēāÄžæŽř'ād'ŽçŽDæŎgāLřijNæfTāēCéIJĀēēAæTřæNĀēūEæŮūā  
ā;āæIJĀāē;ā;fçTlāRēād'ŮçŽDāžšæLŮēĀĒæYřāIJlénYāšCsocketäyLæIēāōđçŎřēfZāžžçL'žæĀgāĀC

## 13.8 11.8 āōđçŎřēIJçlNæŮžæšTērČçTl

### éŮōēčY

ā;āæČšāIJlāyÄäylæúLæAřaijæē;ŠāsČāēC sockets āĀAmultiprocessing  
connections æLŮ ZeroMQ çŽDāšžçāÄāžNäyLāōđçŎřāyÄäylçōĀā■TçŽDēfIJçlNēfGçlNērČçTlřijLRPC

### ēgčāEšæŮžæāĹ

ārEāG;æTřērūāsČāĀāRČæTřāŠNēfTāŽdāĀijā;fçTlāpickleçijŮčāĀāRŎřijNāIJlāy■āRŃçŽDēgčēGLāž  
äyNéíçæYřāyÄäylçōĀā■TçŽDPRCād'ĎçRĒāŽlřijNāRřāžēēčnæTř'ārĹLāLřāyÄäylæIJ■āLāāŽlāy■āŎžrijŽ

```
# rpcserver.py

import pickle
class RPCHandler:
    def __init__(self):
        self._functions = { }
```

```

def register_function(self, func):
    self._functions[func.__name__] = func

def handle_connection(self, connection):
    try:
        while True:
            # Receive a message
            func_name, args, kwargs = pickle.loads(connection.
→recv())

            # Run the RPC and send a response
            try:
                r = self._functions[func_name](*args,**kwargs)
                connection.send(pickle.dumps(r))
            except Exception as e:
                connection.send(pickle.dumps(e))
    except EOFError:
        pass

```

èëAä;£çTíèfZäyIad'DçRĖāZlíjNä;ăéIJĀëëAārĖāōČāLāăĖēāLřāyĀäyIæúLæAřæIJ■āLāāZlāy■ăĂĆă;ăæ  
ä;ĖæŸřä;£çTí multiprocessing āžŞæŸřæIJĀčōĂā■TçŽDāĂĆăyNéÍcæŸřāyĀäyIR-  
PCæIJ■āLāāZlā;Nā■RīijŽ

```

from multiprocessing.connection import Listener
from threading import Thread

def rpc_server(handler, address, authkey):
    sock = Listener(address, authkey=authkey)
    while True:
        client = sock.accept()
        t = Thread(target=handler.handle_connection, args=(client,))
        t.daemon = True
        t.start()

# Some remote functions
def add(x, y):
    return x + y

def sub(x, y):
    return x - y

# Register with a handler
handler = RPCHandler()
handler.register_function(add)
handler.register_function(sub)

# Run the server
rpc_server(handler, ('localhost', 17000), authkey=b'peekaboo')

```

äyžāžĖāžŌäyĀäyIæIJĀčōĂāLūčńřēōĖéŮōæIJ■āLāāZlíjNä;ăéIJĀëëAāLŽāžžāyĀäyIāržāžTçŽDçTlæI





äy■ëfGëGşârSïijNä;äâZTèrëâIJlâŞ■âZTäy■ëfTâZđaijCâyÿâ■ÛçñęäÿšāĀCæĹSázñâIJJSONçŽĐä;Nā■Räy■ā  
ārZāZŌāĒūāZŪçŽĐRPCăôđçŌřä;Nā■RïijNæĹSæŌĹē■Rä;ăçIJNçIJNāIJXML-  
RPCây■ä;ŁçTĹçŽĐ SimpleXMLRPCServer āŠŇ ServerProxy çŽĐăôđçŌřïijN  
āZşârşæYř11.6ārRèĹCây■çŽĐăĒĒăôZāĀC

## 13.9 11.9 çŌĀ■TçŽĐăôçæĹuçnrèôd'èrA

éŬôécŸ

ä;ăæČşâIJlâĹĒâyČaijRçşzçZşÿ■ăôđçŌřäYĀäÿĹçŌĀ■TçŽĐăôçæĹuçnrèĹđæŌëèôd'èrAāĹşèČ;ïijNāRĹā

èğcāEşæŪZæaĹ

ārRäZēāĹ'çTĹ hmac æĹaĹIŬăôđçŌřäYĀäÿĹēĹđæŌëæRæĹNïijNäZŌēĀNăôđçŌřäYĀäÿĹçŌĀ■TèĀNénY

```
import hmac
import os

def client_authenticate(connection, secret_key):
    '''
    Authenticate client to a remote service.
    connection represents a network connection.
    secret_key is a key known only to both client/server.
    '''
    message = connection.recv(32)
    hash = hmac.new(secret_key, message)
    digest = hash.digest()
    connection.send(digest)

def server_authenticate(connection, secret_key):
    '''
    Request client authentication.
    '''
    message = os.urandom(32)
    connection.send(message)
    hash = hmac.new(secret_key, message)
    digest = hash.digest()
    response = connection.recv(len(digest))
    return hmac.compare_digest(digest, response)
```

āşZæIJñāŌşçRĒæYřā;ŞēĹđæŌëāZžçñNāRŌïijNæIJ■āĹaāZĹçZŽăôçæĹuçnrāRŚéĀĀâyĀäÿĹēŽRæIJžçŽĐ  
os.urandom() èĹTâZđâĀijïijĹ'āĀC āôçæĹuçnrāŠNæIJ■āĹaāZĹlāRNæŬūāĹ'çTĹh-  
macāŠNâyĀäÿĹāRĹæIJĹ'ārNæŪžçşēēAŞçŽĐārĒēŞēæĹēèôaçŌŬāĠZâyĀäÿĹāĹārĒāŞĹâyNāĀijāĀCçĐūāRŌā  
æIJ■āĹaāZĹēĀZēĹĠærTē;ČēĹZâyĹāĀijāŠNēĠlāūsēôaçŌŬçŽĐæYřāRęâyĀēĠt'æĹēāEşăôZæŌēāRŬæĹŬæNŠ  
hmac.compare\_digest() āĠ;æTřāĀC ä;ŁçTĹēĹZâyĹāĠ;æTřāRřäZēēAŁāĒ■éA■āĹræŬūēŬt'āĹĒæđRæT  
äÿZāZĒä;ŁçTĹēĹZāZŽāĠ;æTřïijNä;ăēIJĀēēAārĒăôČēZEæĹRāĹrāūsæIJĹ'çŽĐç;ŞçZIJæĹŬæŭĹæAřäZčçāĀây■

```

from socket import socket, AF_INET, SOCK_STREAM

secret_key = b'peekaboo'
def echo_handler(client_sock):
    if not server_authenticate(client_sock, secret_key):
        client_sock.close()
        return
    while True:

        msg = client_sock.recv(8192)
        if not msg:
            break
        client_sock.sendall(msg)

def echo_server(address):
    s = socket(AF_INET, SOCK_STREAM)
    s.bind(address)
    s.listen(5)
    while True:
        c,a = s.accept()
        echo_handler(c)

echo_server(('', 18000))

```

Within a client, you would do this:

```

from socket import socket, AF_INET, SOCK_STREAM

secret_key = b'peekaboo'

s = socket(AF_INET, SOCK_STREAM)
s.connect(('localhost', 18000))
client_authenticate(s, secret_key)
s.send(b'Hello World')
resp = s.recv(1024)

```

## èõìèõž

hmac èòd'èrAçŽDäyÄäyÿyègAä;fçTíIjžæŽræYřâEĚéČlæúLæAřéĂŽăfáčşzçzşăŠNèŁZćlNéŮt' éĂŽ.ä;NăĚĆiijNăĚCăedIJă;ăcijŮăEŽçŽDçşzçzşæúL'ăRĹăLřăyĂăyĹéŽEç;çd'ăy■ăd'ŽăyĹăd'ĐçRĚăŽĹăzNéŮt' çŽDéĂă;ăăRřăžă;fçTíIjNèŁCăŮžăăLăĹăĹăăđăfĹăRĹăIJL'ècŋăĚĂăđôyçŽDèŁZćlNăžNéŮt' æL'■ĚČ;ă;ijă■d' éĂŽăfăăžNăđăyĹiijNăšžăžŮ                      hmac                      çŽDèòd'èrAècŋ                      multiprocessing  
æĹăăĹŮă;fçTíIăĹăđđçŎřă■RĚŁZćlNçŽt' æŎĚçŽDéĂŽăfăăĂĆ

èŁYăIJL'ăyĂçĆzéIJĂèĚĂăijžèrČçŽDăYřéŁđăŎèèòd'èrAăŠNăĹăăřĚăYřăyđ' çăĂăžNăĂĆ  
èòd'èrAăĹăĹăšăžNăŔŎçŽDéĂŽăfăăăúLăAřăYřăžăăYŎăŮĜă;ăăijRăRŠéĂĂçŽDĹiijNăžă;TăžăRĹĚĚĂăĈ

hmacèòd'èrAççŮăşTăšžăžŎăŠĹăyNăĜ;ăTřăĚCMD5ăŠNŠHA-  
ĹiijNăĚşăžŎĚŁZăyĹăIJĹIETF RFC 2104ăy■ăIJL'èrēççZĚăžNçz■ăĂĆ

## 13.10 11.10 aJlč;ŠčzIæIJ■āŁäÿ■āŁăăĚSSL

### ěŮóécŸ

ä;ăæČšăóđčŎřăŸĂăŸłăšžăžŎsocketsčŽĐč;ŠčzIæIJ■āŁäÿ■āŁăăĚSSL■āŁăăĚ

### ěğčăĚşæŮzæąŁ

ssl æłąāłŮěČ;ăŸžăžŤăśČsocketěđăŎěæŮzăŁăSSLčŽĐăŤŕăŇĂăĚČ ssl.  
wrap\_socket() āĠ;æŤŕăŎěăŔŮăŸĂăŸłăŮšă■ŸăIJčŽĐsocketă;IJăŸžăŔČăŤŕăžŮă;ĚčŤİSSLăśČăĚăŇĚđ  
ă;ŇăĚČŋjŇăŸŇéłčăŸŕăŸĂăŸłčŮĂă■ŤčŽĐăžŤč■ŤăIJ■āŁăăĚŤŋjŇěČ;āIJăIJ■āŁăăĚŤŋfăŸžăŁ'ĂăIJŁăđčăŁ

```
from socket import socket, AF_INET, SOCK_STREAM
import ssl

KEYFILE = 'server_key.pem' # Private key of the server
CERTFILE = 'server_cert.pem' # Server certificate (given to client)

def echo_client(s):
    while True:
        data = s.recv(8192)
        if data == b'':
            break
        s.send(data)
    s.close()
    print('Connection closed')

def echo_server(address):
    s = socket(AF_INET, SOCK_STREAM)
    s.bind(address)
    s.listen(1)

    # Wrap with an SSL layer requiring client certs
    s_ssl = ssl.wrap_socket(s,
                            keyfile=KEYFILE,
                            certfile=CERTFILE,
                            server_side=True
                            )

    # Wait for connections
    while True:
        try:
            c, a = s_ssl.accept()
            print('Got connection', c, a)
            echo_client(c)
        except Exception as e:
            print('{:}: {}'.format(e.__class__.__name__, e))

echo_server(('', 20000))
```

äyÑéÍcæĹŚäzñæijŤčd'žäyÄäyĹaóçæĹûçñré£đæŌëæIJ■āŁaāZĹçŽĐäzd'äzŠäĹNā■ŘāĀCāóçæĹûçñräijŽérŭ

```
>>> from socket import socket, AF_INET, SOCK_STREAM
>>> import ssl
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s_ssl = ssl.wrap_socket(s,
                           cert_reqs=ssl.CERT_REQUIRED,
                           ca_certs = 'server_cert.pem')
>>> s_ssl.connect(('localhost', 20000))
>>> s_ssl.send(b'Hello World?')
12
>>> s_ssl.recv(8192)
b'Hello World?'
>>>
```

è£Žçğ■çŽt'æŌëād'ĐçŘĒāžŤāsĆsocketæŮzāijRæIJĹ'äyĹéŮóécŸārsæŸřaóČäy■èČ;āĹĹāë;çŽĐèũ\$æāĠāČ  
äĹNāëČiijNçzĹād'gëČĹāĹĒæIJ■āŁaāZĹäzççāAĹiijĹHTTPāĀXML-  
RPCç■ĹiijĹ'āóđéŽĒäyĹæŸřāšžāžŌ socketserver āžŞçŽĐāĀĆ  
āóçæĹûçñräzççāAāIJläyÄäyĹèĹČénŸāsČäyĹaóđçŌřāĀCæĹŚäzñéIJĀëçAāŘëād'ŮäyĀçğ■çĹ■āĹōäy■āŘNçŽĐ.  
éçŮāĒĹiijNāřžāžŌæIJ■āŁaāZĹèĀNēĹĀiijNāŘřäzëéĀŽè£ĠāČŘäyÑéÍcè£Žæũā;£çŤĹäyÄäyĹmixincşzæĹē

```
import ssl

class SSLMixin:
    '''
    Mixin class that adds support for SSL to existing servers based
    on the socketserver module.
    '''
    def __init__(self, *args,
                 keyfile=None, certfile=None, ca_certs=None,
                 cert_reqs=ssl.CERT_NONE,
                 **kwargs):
        self._keyfile = keyfile
        self._certfile = certfile
        self._ca_certs = ca_certs
        self._cert_reqs = cert_reqs
        super().__init__(*args, **kwargs)

    def get_request(self):
        client, addr = super().get_request()
        client_ssl = ssl.wrap_socket(client,
                                     keyfile = self._keyfile,
                                     certfile = self._certfile,
                                     ca_certs = self._ca_certs,
                                     cert_reqs = self._cert_reqs,
                                     server_side = True)

        return client_ssl, addr
```

äyžāžĒā;£çŤĹè£ŽäyĹmixincşziijNā;āāŘřäzëārĒāóČèũ\$āĒüāžŮæIJ■āŁaāZĹçşzæũāāŘĹāĀCäĹNāëČiijNäy  
RPCæIJ■āŁaāZĹäĹNā■ŘiijŽ



```

# XML-RPC server with SSL

from xmlrpc.server import SimpleXMLRPCServer

class SSLSimpleXMLRPCServer(SSLMixin, SimpleXMLRPCServer):
    pass

Here's the XML-RPC server from Recipe 11.6 modified only slightly,
→to use SSL:

import ssl
from xmlrpc.server import SimpleXMLRPCServer
from sslmixin import SSLMixin

class SSLSimpleXMLRPCServer(SSLMixin, SimpleXMLRPCServer):
    pass

class KeyValueServer:
    _rpc_methods_ = ['get', 'set', 'delete', 'exists', 'keys']
    def __init__(self, *args, **kwargs):
        self._data = {}
        self._serv = SSLSimpleXMLRPCServer(*args, allow_none=True,
→**kwargs)
        for name in self._rpc_methods_:
            self._serv.register_function(getattr(self, name))

    def get(self, name):
        return self._data[name]

    def set(self, name, value):
        self._data[name] = value

    def delete(self, name):
        del self._data[name]

    def exists(self, name):
        return name in self._data

    def keys(self):
        return list(self._data)

    def serve_forever(self):
        self._serv.serve_forever()

if __name__ == '__main__':
    KEYFILE='server_key.pem'      # Private key of the server
    CERTFILE='server_cert.pem'    # Server certificate
    kvserv = KeyValueServer(('', 15000),
                             keyfile=KEYFILE,
                             certfile=CERTFILE)

```

```
kvserve.serve_forever()
```

xmlrpc.client  
https: 15000

```
>>> from xmlrpc.client import ServerProxy
>>> s = ServerProxy('https://localhost:15000', allow_none=True)
>>> s.set('foo', 'bar')
>>> s.set('spam', [1, 2, 3])
>>> s.keys()
['spam', 'foo']
>>> s.get('foo')
'bar'
>>> s.get('spam')
[1, 2, 3]
>>> s.delete('spam')
>>> s.exists('spam')
False
>>>
```

SSL certificate verification  
xmlrpc.client  
https: 15000

```
from xmlrpc.client import SafeTransport, ServerProxy
import ssl

class VerifyCertSafeTransport(SafeTransport):
    def __init__(self, cafile, certfile=None, keyfile=None):
        SafeTransport.__init__(self)
        self._ssl_context = ssl.SSLContext(ssl.PROTOCOL_TLSv1)
        self._ssl_context.load_verify_locations(cafile)
        if certfile:
            self._ssl_context.load_cert_chain(certfile, keyfile)
        self._ssl_context.verify_mode = ssl.CERT_REQUIRED

    def make_connection(self, host):
        # Items in the passed dictionary are passed as keyword
        # arguments to the http.client.HTTPSConnection()
        # constructor.
        # The context argument allows an ssl.SSLContext instance to
        # be passed with information about the SSL configuration
        s = super().make_connection((host, {'context': self._ssl_
        context}))

        return s

# Create the client proxy
s = ServerProxy('https://localhost:15000',
```

```
transport=VerifyCertSafeTransport('server_cert.pem  
↪'),  
allow_none=True)
```

æIJ■āLāZīlārEērAāzēāRŚēĀAçzZāōcæLūçnrīijNāōcæLūçnrælēçəōēōd'āōČčŽDāRĹæŞTæĀgāĀCēfZçg  
āēCādIJæIJ■āLāZīlāCšēēAçəōēōd'āōcæLūçnrīijNāRrāzēārEæIJ■āLāZīlāRrāLlāzččāAāfōāTzāēCāyNīijZ

```
if __name__ == '__main__':  
    KEYFILE='server_key.pem'      # Private key of the server  
    CERTFILE='server_cert.pem'   # Server certificate  
    CA_CERTS='client_cert.pem'   # Certificates of accepted clients  
  
    kvserv = KeyValueServer('', 15000),  
                keyfile=KEYFILE,  
                certfile=CERTFILE,  
                ca_certs=CA_CERTS,  
                cert_reqs=ssl.CERT_REQUIRED,  
            )  
  
    kvserv.serve_forever()
```

äÿžžÈèl'XML-RPCåóœŁũçnráRŚéĀAèfAžžëijŃăfőœŤž ServerProxy  
çŽĎăĹiăġŃăŃŮăžçčăAăçCăÿŃijŽ

```
# Create the client proxy
s = ServerProxy('https://localhost:15000',
               transport=VerifyCertSafeTransport('server_cert.pem',
                                                  'client_cert.pem',
                                                  'client_key.pem'),
               allow_none=True)
```

èóíèőž

ẽrTçjĀăŌzeƒRëaŃŅæIJñèŁĆçŽDăzçcăAèĈ;æŧNèrTă;ăçŽDçşçzçşÉĖ■ç;joèĈ;ăŁZăSŃçRĖèğçSSLăĂĆ  
 ăRrèĈ;æIJAăd'ğçŽDăŃSæLŸăŸrăçCă;TăŸĂă■ěă■ěçŽDèŌăRŪăLăğNéĖ■ç;őkeyăĂĂerĂăžăăSŃăŤăŸăŮă

[illegible]

```
bash % openssl req -new -x509 -days 365 -nodes -out server_cert.pem  
          -keyout server_key.pem
```

Generating a 1024 bit RSA private key

writing new private key to `server_key.pem`





```

        if not msg:
            break
        print('CHILD: RECV {!r}'.format(msg))
        s.send(msg)

def server(address, in_p, out_p, worker_pid):
    in_p.close()
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(address)
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)
        send_handle(out_p, client.fileno(), worker_pid)
        client.close()

if __name__ == '__main__':
    c1, c2 = multiprocessing.Pipe()
    worker_p = multiprocessing.Process(target=worker, args=(c1, c2))
    worker_p.start()

    server_p = multiprocessing.Process(target=server,
                                       args=(',', 15000), c1, c2, worker_p.pid))
    server_p.start()

    c1.close()
    c2.close()

```

[illegible]

èóìèőž

ǎrzǎžŎǎd' gēĆlǎĽEǧlŃǎžŔǎŚŸǎĽēēōśǎĽĽǎy■ǎŔŃēŹǧlŃǎžŤŮŕ' āijǎēĀŠǎŮĜǎžŭǎŔŔēŕŕcņēǎē;ǎĈŔǎēšǎ  
 ā;ĽǎēŸŕiijŃǎēĽĽ'ǎŮŭǎĀŽǎōĈǎēŸŕǎđDǎžžǎyĀǎyŭŕǎŕǎĽŕ'ǎśŤǧșșzǧșșǧșŽđǎ;ĽǎēĽĽ'ǧŤĭǧŽđǎŭēǎĽŭǎĀĈǎ;ŃǎēĈ  
 ā;ǎǎŔǎžēǎēĽĽ'ǎđ'ŽǎyĭPythoneğǧcēĜĽǎŽĽǎōđǎ;ŃiijŃǎŕĽǎēŮĜǎžŭǎŔŔēŕŕcņēǎēāijǎēĀŠǧžŽǎĽŭǎōĈēğǧcēĜĽǎŽǎē

send\_handle()      ħŠŃ      recv\_handle()      ħĜjæŦřăŔlèĈjăd'şçŦlăžŎ  
multiprocessing ěfdæŎěăĂĈ äjfcŦlăŏĈăžňæIěăžçæŽfcŏăéAşçŽĎăjfcŦlĭijLăŔĈèĂĈ11.7èĹĈĭijL'ĭijŃ  
ăjŃăęĈĭijŃăjăăŔŕăžèèŏl'æIJ■ăLăăŽlăŃăŭëăjIJèĂĖăŔĎèĜlăžèă■ŦçŃňçŽĎĭŃăžŔæIěăŔŕăLăĂĈăyŃéIçæŦ

```
# servermp.py
from multiprocessing.connection import Listener
from multiprocessing.reduction import send_handle
import socket

def server(work_address, port):
    # Wait for the worker to connect
    work_serv = Listener(work_address, authkey=b'peekaboo')
    worker = work_serv.accept()
    worker_pid = worker.recv()

    # Now run a TCP/IP server and send clients to worker
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(('', port))
    s.listen(1)
    while True:
        client, addr = s.accept()
        print('SERVER: Got connection from', addr)

        send_handle(worker, client.fileno(), worker_pid)
        client.close()

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: server.py server_address port', file=sys.
↳ stderr)
        raise SystemExit(1)

    server(sys.argv[1], int(sys.argv[2]))
```

èĤŔëăŃëfŽăylæIJ■ăLăăŽlĭijŃăŔlèIJĂèçAæL'ğèăŃ python3 servermp.py /tmp/servconn  
15000 ĭijŃăyŃéIçæŦŕçŽyăžŦçŽĎăŭëăjIJèĂĖăžççăAĭijŽ

```
# workermp.py

from multiprocessing.connection import Client
from multiprocessing.reduction import recv_handle
import os
from socket import socket, AF_INET, SOCK_STREAM

def worker(server_address):
    serv = Client(server_address, authkey=b'peekaboo')
    serv.send(os.getpid())
    while True:
        fd = recv_handle(serv)
```

```

    print('WORKER: GOT FD', fd)
    with socket(AF_INET, SOCK_STREAM, fileno=fd) as client:
        while True:
            msg = client.recv(1024)
            if not msg:
                break
            print('WORKER: RECV {!r}'.format(msg))
            client.send(msg)

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 2:
        print('Usage: worker.py server_address', file=sys.stderr)
        raise SystemExit(1)

    worker(sys.argv[1])

```

python3 workermp.py  
 /tmp/servconn . æTŁædIJeũšä;ŁçTÍPipe()ä;Nä■RæYřăŏNăĚlăyĂæăüçŽĎăĂĆ  
 æŮĜăžŭæRŘèŁřçñçŽĎăijăéĂšăijŽæŭLăRĹăĹrUNIXăşşăēŮăŎēă■ŮçŽĎăĹZăžZăŠNăēŮăŎēă■ŮçŽĎ  
 sendmsg() æŮžæşTăĂĆ äy■ēŁĜēŁŽçģ■æŁĂæIJřăžŭäy■ăyŷëĝAiiJNăyNēĹcæYřă;ŁçTĹăēŮăŎēă■ŮăĹēăijă

```

# server.py
import socket

import struct

def send_fd(sock, fd):
    '''
    Send a single file descriptor.
    '''
    sock.sendmsg([b'x'],
                  [(socket.SOL_SOCKET, socket.SCM_RIGHTS, struct.
→pack('i', fd))])
    ack = sock.recv(2)
    assert ack == b'OK'

def server(work_address, port):
    # Wait for the worker to connect
    work_serv = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    work_serv.bind(work_address)
    work_serv.listen(1)
    worker, addr = work_serv.accept()

    # Now run a TCP/IP server and send clients to worker
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, True)
    s.bind(('', port))
    s.listen(1)
    while True:

```



```

        client, addr = s.accept()
        print('SERVER: Got connection from', addr)
        send_fd(worker, client.fileno())
        client.close()

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: server.py server_address port', file=sys.
→stderr)
        raise SystemExit(1)

    server(sys.argv[1], int(sys.argv[2]))

```

äÿÑéÍæŸřä;ŁçŤíäčŮæŎěă■ŮçŽďăüěä;IJeĂěăóđçŎřijŽ

```

# worker.py
import socket
import struct

def recv_fd(sock):
    '''
    Receive a single file descriptor
    '''
    msg, ancdata, flags, addr = sock.recvmsg(1,
→socket.CMSG_LEN(struct.
    calcsz('i')))

    cmsg_level, cmsg_type, cmsg_data = ancdata[0]
    assert cmsg_level == socket.SOL_SOCKET and cmsg_type == socket.
→SCM_RIGHTS
    sock.sendall(b'OK')

    return struct.unpack('i', cmsg_data)[0]

def worker(server_address):
    serv = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    serv.connect(server_address)
    while True:
        fd = recv_fd(serv)
        print('WORKER: GOT FD', fd)
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM,
→
        fileno=fd) as client:
            while True:
                msg = client.recv(1024)
                if not msg:
                    break
                print('WORKER: RECV {!r}'.format(msg))
                client.send(msg)

```

```

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 2:
        print('Usage: worker.py server_address', file=sys.stderr)
        raise SystemExit(1)

    worker(sys.argv[1])

```

æċCædIJä;æċſåIJlä;ăċŽDċlNăžRăy■ăijăĉĂſæŰĞăžŭæRRĕĕřċñĉijNăžžĕőőă;ăăRCĉĚĚăĚŭăžŰăyĂăžŽ  
 æřTăĉĈ Unix Network Programming by W. Richard Stevens (Prentice  
 Hall, 1990). âIJlWindowsăyŁăijăĉĂſæŰĞăžŭæRRĕĕřċñĉĕŭſUnixæŶřăy■ăyĂăăŭċŽDĉijNăžžĕőőă;ăĉă  
 multiprocessing.reduction äy■ċŽDăžRăžċĉăAĉIJNĉIJNăĚŭăŭă;IJăŎſĉĚĚăĈ

## 13.12 11.12 ċŘĚĕğĉăžNăžŭĕl'ſăLlċŽDIO

### éŰőĉĚ

ă;ăăžŦĕĕăŭſĉzRăRñĕĚĞăſžăžŎăžNăžŭĕl'ſăLlăĚŰăijCă■ĉlOĉŽDăNĚĉijNă;ĚăŶřă;ăĕĚŶăy■ĉĈ;ăőNăĚ  
 æĚŰĕĂĚăŶřăĉCædIJä;ĚĉŦlăőĈĉŽDĕřlăijŽăřză;ăĉŽDċlNăžRăžĉŦſăžĂăžĚă;ſăſ■ăĈ

### ĕğĉăĚſæŰžăæĚl

ăžNăžŭĕl'ſăLl/OăIJñĕřlăyŁăĚĕĕőſăřſæŶřăĚăſžăæIJñl/Oăſ■ă;IJĉijĚăřTăĉCĕřzăſNăĚŽĉijLĕ;ňăNŰăyž  
 äĚNăĉĈĉijNă;ſæŦřă■ăIJăſŦăyĤsocketăyĤĕĉnăŎĉăRŰăăRŎĉijNăőĈăijŽĕ;ňă■ĉăĚŦăyĂăyĤ  
 receive äžNăžŭĉijNĉDŭăRŎĕĉnă;ăăőŽăžĚĤĉŽDăŽĉĕřCăŰžăſŦăĚŰăăĚ;ăĚăĚăĕăd'ĎĉŘĚăĈ  
 ä;IJăyžăyĂăyĤăRĕĈ;ĉŽDĕřŭăğNĉĈĉijNăyĂăyĤăžNăžŭĕl'ſăLlċŽDăæĚăĉđŭăRĕĈ;ăijŽăžĕăyĂăyĤăőĉŦăřăĚă

```

class EventHandler:
    def fileno(self):
        'Return the associated file descriptor'
        raise NotImplemented('must implement')

    def wants_to_receive(self):
        'Return True if receiving is allowed'
        return False

    def handle_receive(self):
        'Perform the receive operation'
        pass

    def wants_to_send(self):
        'Return True if sending is requested'
        return False

    def handle_send(self):
        'Send outgoing data'
        pass

```

efZäyłçşzçŽĐăđăŃăĲăÿzæŔŠăzűěćăŤăăĚçşzăijjăÿŃéİçèĚăăüçŽĐăžŃăzűăłçŎŕăÿ■ijŽ

```
import select

def event_loop(handlers):
    while True:
        wants_recv = [h for h in handlers if h.wants_to_receive()]
        wants_send = [h for h in handlers if h.wants_to_send()]
        can_recv, can_send, _ = select.select(wants_recv, wants_
→ send, [])
        for h in can_recv:
            h.handle_receive()
        for h in can_send:
            h.handle_send()
```

ăžŃăzűăłçŎŕçŽĐăĚşéŤőéĆłăĹăŸŕ select() èŕČçŤĲijŃăŏČăijŽăÿăŮ■è;őèŕçăŮĜăzűăŔŔèĚŕçŋē  
ăĲĲèŕČçŤĲ select() äžŃăĹ■ijŃăŮűéŮŕăłçŎŕăijŽèŕcéŮőăĹăĤăĲçŽĐăđŤĐçŔĚăŽĲăĲăĚşăőŽăŞĲăÿĂă  
çĐűăŔŎăŏČăŕĚçzŞăđĲăĹŮëăĲăŔŔăłçzŽ select() äĤçĐűăŔŎ select()  
èĚŤăŽđăĜĚăđŤĜăŎëăŔŮăĹŮăŔŠéĂăçŽĐăŕžèşăçzĐăĹŔçŽĐăĹŮëăĲăĤ  
çĐűăŔŎçŽÿăžŤçŽĐ handle\_receive() æĹŮ handle\_send()  
æŮžæşŤèćŋèğăŔŠăĤĤ

çijŮăĚŽăžŤçŤĲŃăžŔçŽĐăŮűăĂžijŃEventHandler  
çŽĐăđăŃăĲăÿzæŔŠăzűěćăŤăăĚçşzăijjăÿŃéİçăŸŕăÿđăÿłçŏĂăŤçŽĐăşžăžŎUDPçĲŞçzĲăĲăăĲăçŽĐăđă

```
import socket
import time

class UDPServer(EventHandler):
    def __init__(self, address):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.sock.bind(address)

    def fileno(self):
        return self.sock.fileno()

    def wants_to_receive(self):
        return True

class UDPTimeServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(1)
        self.sock.sendto(time.ctime().encode('ascii'), addr)

class UDPEchoServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(8192)
        self.sock.sendto(msg, addr)

if __name__ == '__main__':
    handlers = [ UDPTimeServer((' ', 14000)), UDPEchoServer((' ',
→ 15000)) ]
```

```
event_loop(handlers)
```

ætÑerTēfZæōtāzččāAīijÑerTçĬÄāzŌāRēād'ŪāyÄāyIPythonèġčéĜLāŽlè£đæŌēāōČīijŽ

```
>>> from socket import *
>>> s = socket(AF_INET, SOCK_DGRAM)
>>> s.sendto(b'', ('localhost', 14000))
0
>>> s.recvfrom(128)
(b'Tue Sep 18 14:29:23 2012', ('127.0.0.1', 14000))
>>> s.sendto(b'Hello', ('localhost', 15000))
5
>>> s.recvfrom(128)
(b'Hello', ('127.0.0.1', 15000))
>>>
```

āōđčŌrāyÄāyITCPæIĬāLāāŽlāijŽæŽt'āLāād'■æIČāyÄçČzīijNāZāāyžæfRāyÄāyIāōčæLūčnréČ;ēēAāLĬ  
āyNēIčæYrāyÄāyITCPāžTç■TāōčæLūčnrā;Nā■ŘīijŽ

```
class TCPServer(EventHandler):
    def __init__(self, address, client_handler, handler_list):
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_
→STREAM)
        self.sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,
→ True)
        self.sock.bind(address)
        self.sock.listen(1)
        self.client_handler = client_handler
        self.handler_list = handler_list

    def fileno(self):
        return self.sock.fileno()

    def wants_to_receive(self):
        return True

    def handle_receive(self):
        client, addr = self.sock.accept()
        # Add the client to the event loop's handler list
        self.handler_list.append(self.client_handler(client, self.
→handler_list))

class TCPClient(EventHandler):
    def __init__(self, sock, handler_list):
        self.sock = sock
        self.handler_list = handler_list
        self.outgoing = bytearray()

    def fileno(self):
        return self.sock.fileno()
```

```

def close(self):
    self.sock.close()
    # Remove myself from the event loop's handler list
    self.handler_list.remove(self)

def wants_to_send(self):
    return True if self.outgoing else False

def handle_send(self):
    nsent = self.sock.send(self.outgoing)
    self.outgoing = self.outgoing[nsent:]

class TCPEchoClient(TCPClient):
    def wants_to_receive(self):
        return True

    def handle_receive(self):
        data = self.sock.recv(8192)
        if not data:
            self.close()
        else:
            self.outgoing.extend(data)

if __name__ == '__main__':
    handlers = []
    handlers.append(TCPServer(('', 16000), TCPEchoClient, handlers))
    event_loop(handlers)

```

TCPä;Nā■ŘčŽDāĚšéTōçĆzæYřazŎad'ĐčŘEāZÍäy■āLŮeāIācđāLāāŠNāLāeZđ'āóçæLūçnrçŽDæŠ■ā;IJā  
 āřzæfRāyĀäyĭēŁđæŎēiijNāyĀäyĭæŮřčŽDad'ĐčŘEāZÍlēcñāLZāzžāzūāLāāLřāLŮeāIāy■āĀCā;ŠēŁđæŎēēcñāĚ  
 āēČādIJā;āēŁRēāNčIñāzRāzūēfTčIĀčTÍTelnetæLŮčšzāijijāuēāĚūēŁđæŎēiijNāōČāijŽāřEā;āāRSéĀAçŽDæŮ

## èóĭeőž

āóđéŽĚäyŁæL'ĀæIJLčŽDāzNāzūēl'sāLÍæāEæđūāŎšçŘEēūšāyŁēIćçŽDā;Nā■ŘčŽyāuōæŮāāGāāĀČāōō  
 ā;EæYřāIJāIJāæāyāŁČčŽDēČIāLEiijNēČ;āijZæIJL'āyĀäyĭē;ōēfrcçŽDā;IçŎræĭēæčĀæšēæt'zāLĭsocketiijNā

āzNāzūēl'sāLÍI/OçŽDāyĀäyĭāRřēČ;āē;ād'ĐæYřāōČēČ;ād'ĐčŘEēIđāyāđ'ğçŽDāzūāRŠēŁđæŎēiijNēĀN  
 āzšāřsæYřēf'ijNselect()ērČčTĭiijLæLŮāĚūāzŮč■L'æTĭLčŽDĭiijL'ēČ;çŽSāRñād'ģēGRčŽDsocketāzūāŠ■  
 āIJā;IçŎrāy■āyĀæñāđ'ĐčŘEāyĀäyĭāzNāzūiijNāzūāy■ēIJĀēçĀāĚūāzŮčŽDāzūāRSæIJzāLūāĀČ

āzNāzūēl'sāLÍI/OçŽDčijžçĆzæYřæšqæIJLčIJšæ■čçŽDāRñæ■ēæIJzāLūāĀČ  
 āēČādIJāzā;TāzNāzūāđ'ĐčŘEāZÍæŮzæšTēYzāāđæLŮæLğēāNāyĀäyĭēĀŮæŮūēōāçōŮiijNāōČāijŽēYzāāđ  
 ērČčTĭēČcāzZāzūāy■æYřāzNāzūēl'sāLÍlēčŎāiijçŽDāzŠāG;æTřāzšāijZæIJL'ēŮōēčYiijNāRñæāūēçĀæYřæš

āřzāžŎēYzāāđæLŮēĀŮæŮūēōāçōŮčŽDēŮōēčYāRřāzēēĀŽēŁGāřEāzNāzūāRSéĀĀäyĭāĚūāzŮā■TçNñç  
 āy■ēŁGriijNāIJāzNāzūā;IçŎrāy■āijTāĚēād'ŽçžŁčIñāŠNād'ŽēŁZčIñæYřærTē;ČæčYæL'NçŽDĭiijN  
 āyNēIćçŽDā;Nā■ŘāijTčđ'žāzEāēČā;Tā;ŁçTĭ  
 æIāāIŮæĭēāōđçŎriijŽ concurrent.futures

```

from concurrent.futures import ThreadPoolExecutor
import os

class ThreadPoolHandler(EventHandler):
    def __init__(self, nworkers):
        if os.name == 'posix':
            self.signal_done_sock, self.done_sock = socket.
↪socketpair()
        else:
            server = socket.socket(socket.AF_INET, socket.SOCK_
↪STREAM)
            server.bind(('127.0.0.1', 0))
            server.listen(1)
            self.signal_done_sock = socket.socket(socket.AF_INET,
                                                    socket.SOCK_
↪STREAM)
            self.signal_done_sock.connect(server.getsockname())
            self.done_sock, _ = server.accept()
            server.close()

        self.pending = []
        self.pool = ThreadPoolExecutor(nworkers)

    def fileno(self):
        return self.done_sock.fileno()

    # Callback that executes when the thread is done
    def _complete(self, callback, r):

        self.pending.append((callback, r.result()))
        self.signal_done_sock.send(b'x')

    # Run a function in a thread pool
    def run(self, func, args=(), kwargs={}, *, callback):
        r = self.pool.submit(func, *args, **kwargs)
        r.add_done_callback(lambda r: self._complete(callback, r))

    def wants_to_receive(self):
        return True

    # Run callback functions of completed work
    def handle_receive(self):
        # Invoke all pending callback functions
        for callback, result in self.pending:
            callback(result)
            self.done_sock.recv(1)
        self.pending = []

```

aJlāzčcāAäy■ijNrun() æŨzæsTēcncŦlāIēārEāũēä;IJæRŘāžd'čzŽāZdērČāĠ;æTṛæšāijNād'ĐčŘEāōN  
 āódéŽĚāũēä;IJècñæRŘāžd'čzŽ ThreadPoolExecutor āōdä;NāĀĆ

äy■ēfGäyÄäyléŽčĆzæYřā■RēřČèõæçõŮčzŠæđIJāŠNāzNāzūā;łçŎřijNāyžāzEëğčāEşāõČřijNæĹŚāznāĹŽāz  
 ā;ŠçžŁčĹNæśāāōNæĹRāũēä;IJāRŎřijNāōČāijŽæĹgèaŃçśzäy■çŽĎ \_complete()  
 æŮzæšTāĀĆēfŽäylæŮzæšTāE■æšRäyĹsocketäyĹāEŽāĚēā■ŮēĹĆāzNāĹ■āijŽèõšæŃČetŭçŽĎāZðerČāG;æT  
 fileno() æŮzæšTēfTāZđāRēād'ŮçŽĎēĆčāyĹsocketāĀĆāZāæ■đ'rijNèfŽäylā■ŮēĹĆècāEŽāĚēæŮřijNā  
 çĎūāRŎhandle\_receive() æŮzæšTēcñæfĀæt'žāzūäyžæĹĀæIJĹāzNāĹ■æRŘāžđ'çŽĎāũēä;IJæĹgèaŃ  
 āĹçŽ;èõřijNēřt'āžEēfŽāzĹād'ŽèfđæĹSèĠāũséČ;æŽTāžEāĀĆ  
 äyNēĹcæYřāyÄäyĹçõĀā■TçŽĎæIJ■āĹāZĹrijNæijTçđ'žāžEāçCā;Tā;ŁçTĹčžŁčĹNæśāāIēāõđçŎřēĀŮæŮūçŽĎē

```
# A really bad Fibonacci implementation
def fib(n):
    if n < 2:
        return 1
    else:
        return fib(n - 1) + fib(n - 2)

class UDPFibServer(UDPServer):
    def handle_receive(self):
        msg, addr = self.sock.recvfrom(128)
        n = int(msg)
        pool.run(fib, (n,), callback=lambda r: self.respond(r,
→addr))

    def respond(self, result, addr):
        self.sock.sendto(str(result).encode('ascii'), addr)

if __name__ == '__main__':
    pool = ThreadPoolHandler(16)
    handlers = [ pool, UDPFibServer(('', 16000))]
    event_loop(handlers)
```

èĹRēāNēfŽäylæIJ■āĹāZĹrijNçĎūāRŎērTçĹĀçTĹāĚūāōČPythončĹNāžRæĹæťNērTāōČřijŽ

```
from socket import *
sock = socket(AF_INET, SOCK_DGRAM)
for x in range(40):
    sock.sendto(str(x).encode('ascii'), ('localhost', 16000))
    resp = sock.recvfrom(8192)
    print(resp[0])
```

ā;āāžTèrēēČ;āIJāy■āRŃçĹŮāRčāy■éG■ād'■çŽĎæĹgèaŃèfŽäylčĹNāžRrijNāzūāyTāy■āijŽā;śāš■āĹrāĚ  
 āũšçžRēYĚērzaōNāžEēfŽäyĀārRēĹČřijNēĆčāzĹā;āāžTèrēā;ŁçTĹēfŽēGŃçŽĎāzçčāĀāRŮřijšāzšèõyāy  
 äy■ēfGřijNāēĆæđIJā;āçRĚēğčāžEāšžæIJnāŎšçRĚřijNā;āāršēČ;çRĚēğçēfŽāžZæāEæđūæĹĀā;ŁçTĹčŽĎæy  
 ā;IJāyžāržāZðerČāG;æTřçijŮčĹNçŽĎæŽēāžçřijNāzNāzūēĹśāĹčijŮčāĀæIJĹæŮūāĀZāijŽā;ŁçTĹāĹrā■RčĹNřij

## 13.13 11.13 āRŚéĀĀäyŎæŎæTūād'gādNæTřçžĎ

éŮōécY

ā;āēēĀēĀŽēfGç;ŚçzIJēfđæŎēāRŚéĀĀāŠNæŎēāRŮēfđçz■æTřæ■ōçŽĎād'gādNæTřçžĎřijNāzūār;éGR

èġčǎẸșæŮžæǻŁ

äyNéIcçZĐäĜ;æTřáLl'çTl memoryviews æIěaRŚéĀAāSŇæŌěaRŮad'gæTřçzĐiiž

```
# zerocopy.py

def send_from(arr, dest):
    view = memoryview(arr).cast('B')
    while len(view):
        nsent = dest.send(view)
        view = view[nsent:]

def recv_into(arr, source):
    view = memoryview(arr).cast('B')
    while len(view):
        nrecv = source.recv_into(view)
        view = view[nrecv:]
```

äyžāẸætNērTçlÍnāẖRijñĖēŨāĖĹāĹZāzzäyÄäyĹēĀŽēfĠsocketēfđāŖĖčŽDāĬ■āĹāāZĹāŠñāóćĹŁuçnrŕ

```
>>> from socket import *
>>> s = socket(AF_INET, SOCK_STREAM)
>>> s.bind(('', 25000))
>>> s.listen(1)
>>> c,a = s.accept()
>>>
```

åIJaóœŁuçnríijLâRęad'ŮäyĂäyłęğćęĠŁăZlăy■íijL'íijŻ

```
>>> from socket import *
>>> c = socket(AF_INET, SOCK_STREAM)
>>> c.connect(('localhost', 25000))
>>>
```

æIɲn̩ɛL̥C̥Ž̌D̥C̥Ž̌oæǣGæY̌r̥ä;æĈĵĕĂŽĕfGĕŁđæŌëäijäe;ŞăyĂăy̌l̥eũĖăd'gæȚr̥çzDăĂCĕfŽçg■æĈĖăĖȚçŽ̌Ď  
array ælaaIŮæLŮ numpy ælaaIŮæIĕăLŽăzzaȚr̥çzĎiijŽ̌

```
# Server
>>> import numpy
>>> a = numpy.arange(0.0, 50000000.0)
>>> send_from(a, c)
>>>

# Client
>>> import numpy
>>> a = numpy.zeros(shape=50000000, dtype=float)
>>> a[0:10]
array([ 0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.])
>>> recv_into(a, c)
>>> a[0:10]
array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.])
```



>>>

## ëóíëőž

āIJlæTṛæ■ōārEéZĖāđNāLĖāyČāijRēōačōŮāŠNāzšēāNēōačōŮčlNāžRāy■iijNēGĥāūsāĖZčlNāžRāĖĭēāōđč  
āy■ēfGriijNēēAæYřā;āčāōāōđæČšēfZæāūāAŽiijNā;āāRřēČ;éIJĀēēAārĖā;āčŽDæTṛæ■ōē;ñæ■céLRāŌšāgN  
ā;āāRřēČ;ēfYēIJĀēēAārĖāTṛæ■ōāLĖāL'sāēLRād'ŽāyĥāŮiijNāZāyžād'gēČlāLĖāSNč;ŠčzIJčZyāĖščŽDāGj

āyĀčg■æŮzæšTæYřā;fčTlæšRčg■æIJāLūāžRāLŮāNŮæTṛæ■ōāĀTāĀTāRřēČ;ārĖāĖŮē;ñæ■céLRāyĀ  
āy■ēfGriijNēfZæāūāIJĀčZLāijŽāLZāžzæTṛæ■ōčŽDāyĀāyĥād'■āLūāĀČ  
ārščōŮā;āāRlæYřéZūččŌčŽDāAžēfZāžZiijNā;āčŽDāžččāAæIJĀčZLēfYæYřāijZæIJL'ād'gēGRčŽDārRādNā

æIJñēLČēĀžēfGā;fčTlāĖĖā■YēgĖāZ;āsTčd'žāžĖāyĀāžZé■TæšTæš■ā;IJāĀČ  
æIJñēl'āyLūiijNāyĀāyĥāĖĖā■YēgĖāZ;ārśæYřāyĀāyĥāūsā■YāIJlæTṛčZDčŽDēēĖčZŮāsČāĀČāy■āzĖāzĖāYřē  
āĖĖā■YēgĖāZ;ēfYēČ;āžēāy■āRŇčŽDæŮzāijRē;ñæ■céLRāy■āRŇčšzādNāĖēāčŌræTṛæ■ōāĀČ  
ēfZāyĥārśæYřāyNēlčēfZāyĥēr■āRēčZDčŽōčŽDiiijZ

```
view = memoryview(arr).cast('B')
```

āōČæŌēāRŮāyĀāyĥæTṛčZD arrāzūārĖāĖŮē;ñæ■cāyžāyĀāyĥæŮāčņēāRŮā■ŮēLČčŽDāĖĖā■YēgĖāZ;āĀČ  
ærTāēČ socket.send() æLŮ send.recv\_into() āĀČ  
āIJlāĖĖēČlriijNēfZāžZæŮzæšTēČ;ād'ščZt'æŌēæš■ā;IJēfZāyĥāĖĖā■YāNžāššāĀČā;NāēČiijNsock.  
send() čZt'æŌēāžŌāĖĖā■Yāy■āRŠčTšæTṛæ■ōēĀNāy■ēIJĀēēAād'■āLūāĀČ send.  
recv\_into() ā;fčTlēfZāyĥāĖĖā■YāNžāššā;IJāyžæŌēāRŮæš■ā;IJčŽDē;ŠāĖēčijŠāĖšāNžāĀČ

āLl'āyNčŽDāyĀāyĥēZ;čČzārśæYřsocketāG;æTṛāRřēČ;ārĥæš■ā;IJēČlāLĖæTṛæ■ōāĀČ  
ēĀžāyāēĖēōšriijNāēLŠāžñā;Ůā;fčTlā;Lād'Žāy■āRŇčŽD send() āŠN recv\_into()  
āĖēāijāē;ŠæTt'āyĥæTṛčZDāĀČ āy■čTlāNēāfČiijNærRæñāæš■ā;IJāRŌiijNēgĖāZ;āijZēĀžēfGāRŠēĀAæLŮ  
æŮřčŽDēgĖāZ;āRŇæāūāžšæYřāĖĖā■YēēĖčZŮāsČāĀČāZāæ■d'iijNēfYæYřæšāæIJL'āžzā;TčŽDād'■āLūāš

ēfZēGNæIJL'āyĥēŮōēčYārśæYřæŌēāRŮēĀĖāfĖēāzāžNāēLčšēēAšæIJL'ād'ŽārŠæTṛæ■ōēēAēčnāRŠēĀ  
āžēā;ĖāōČēČ;ēčDāLĖēĖ■āyĀāyĥæTṛčZDāLŮēĀĖčāōāfĭāōČēČ;ārĖāēŌēāRŮčŽDæTṛæ■ōāT;āĖēāyĀāyĥāūs  
āēČæđIJæšāāLđæšTčšēēAščŽDērĭiijNāRŠēĀAēĀĖārśā;ŮāēLārĖæTṛæ■ōād'gārRāRŠēĀAēfGāēĭiijNčDūā

## 14 čňňā■AžNčňāiijŽāžúāRŠcijŮčlN

āržāžŌāžūāRŠcijŮčlN, PythonæIJL'ād'Žčg■éTfæIJšæTṛæNĀčŽDæŮzæšT,  
āNĖæNñād'ŽčžčlN, ēČčTlā■RēfZčlN, āžēārLāRĐčg■āRĐæāūčŽDāĖšāžŌčTšæLRāZlāG;æTṛčŽDæLĀāū  
ēfZāyĀčňāārĖāijŽčžZāGžāžūāRŠcijŮčlNāRĐčg■æŮzéĭčŽDæLĀāūg,  
āNĖæNñēĀžčTlčŽDād'ŽčžčlNæLĀæIJfāžēārLāžūēāNēōačōŮčŽDāōđčŌræŮzæšT.

āČRčZŘēNāyřārNčŽDčlNāžRāšYæL'ĀčšēēAščŽDēČčæū,  
ād'gāōūæNēāfČāžūāRŠčŽDčlNāžRāæIJL'æ;IJāIJčŽDā■séZl'. āžāæ■d',  
æIJñčāčŽDāyžēēAčZōæāGāžNāyĀæYřčžZāGžæZt'āLāārřāfāēŮŮāŠNæYšērČērTčŽDāžččāA.

Contents:

## 14.1 12.1 aRraŁläyÖaAıJæ■ćçžŁçİÑ

### éUóécŸ

ä;äëAäyžéİJÄëAázüaŖSæL'gëaŃçŽDäzççAaŁLŽázž/éŤÄæŖAçžŁçİÑ

### èğcâEşæŮzæaŁ

threading                      åžŞaŖräžëaıJıa■ŤçŃñçŽDçžŁçİÑäy■æL'gëaŃäzžä;ŤçŽDâıJı  
Python                      äy■aŖräžëèŖČçŤıçŽDâržèşaaĀČä;äaŖräžëaŁŽázžäyÄäyŁ                      Thread  
âržèşaažüaŖEä;äëAæL'gëaŃçŽDâržèşaažè target aŖCæŤŖçŽDâ;çaijŖæŖŖä;ŽçžŽèŖâržèşaaĀČ  
äyŃéİcæŸŖäyÄäyŁçōĀa■ŤçŽDä;Ńa■ŖiijŽ

```
# Code to execute in an independent thread
import time
def countdown(n):
    while n > 0:
        print('T-minus', n)
        n -= 1
        time.sleep(5)

# Create and launch a thread
from threading import Thread
t = Thread(target=countdown, args=(10,))
t.start()
```

ä;Şä;aaŁŽázžäë;äyÄäyŁçžŁçİÑâržèşaaŖÖiijŃèŖâržèşaažüaäy■aijŽçñŃa■şæL'gëaŃiijŃéŽd' éİdä;äèŖČçŤİ  
start() æŮžæşŤiijŁä;Şä;äèŖČçŤİ start() æŮžæşŤæŮiijŃaōČaijŽèŖČçŤİä;äaijæéĀŞèŁZæİççŽDâĜ;æŤ  
POSIX çžŁçİÑæŁŮëĀĒäyÄäyŁ Windows çžŁçİÑiijL'iijŃèŁZäžŽçžŁçİÑârEçŤsæŞ■ä;İçşžçžşæİëaĒİæİČçōaçŁ

```
if t.is_alive():
    print('Still running')
else:
    print('Completed')
```

ä;äazşaŖräžëaŖEäyÄäyŁçžŁçİÑaŁäaĒëaŁŖa;ŞaŁ■çžŁçİÑiijŃázüç■L'ä;ĒaōČçžŁæ■ćiijŽ

```
t.join()
```

PythonèğcéĠŁāŽİçŽŤ' aŁŖæL'ÄæİJL'çžŁçİÑéČ;çžŁæ■cāL'■äž■äİæŃAèŁŖëaŃaĀČâržäžÖéİJÄëAéŤŁæ  
ä;ŃäëĆiijŽ

```
t = Thread(target=countdown, args=(10,), daemon=True)
t.start()
```

ârŖÖaŖŖçžŁçİÑæŮäæşŤç■L'ä;ĒiijŃäy■èŁĠiijŃèŁZäžŽçžŁçİÑaijŽaıJıäyçžŁçİÑçžŁæ■cæŮüèĠİaŁİéŤĀ  
éŽd' äžĒæČäyŁæL'Āçd'žçŽDäyŤ' äyŁæŞ■ä;İiijŃázüæşaaİJL'ad' İad' ŽaŖräžëâržçžŁçİÑaAžçŽDäžŃæČĒāĀČ

```

class CountdownTask:
    def __init__(self):
        self._running = True

    def terminate(self):
        self._running = False

    def run(self, n):
        while self._running and n > 0:
            print('T-minus', n)
            n -= 1
            time.sleep(5)

c = CountdownTask()
t = Thread(target=c.run, args=(10,))
t.start()
c.terminate() # Signal termination
t.join()      # Wait for actual termination (if needed)

```

æĈædIJçžċlNæL'gèaÑäyÄāZāČŘI/OèfZæuċŽDèYzāqđæ\$■ä;IJiijÑéCčázLéĀŽèfGè;őercæİēcZLæ■  
 äĹNā■ŘæCäyNiiž

```

class IOTask:
    def terminate(self):
        self._running = False

    def run(self, sock):
        # sock is a socket
        sock.settimeout(5)          # Set timeout period
        while self._running:
            # Perform a blocking I/O operation w/ timeout
            try:
                data = sock.recv(8192)
                break
            except socket.timeout:
                continue
            # Continued processing
            ...
        # Terminated
        return

```

èóİèőž

çTšazŌăĒlāsĀegcéGŁéTĀiijLGILiijLçŽDăŌşăZăiijŊPython  
 çŽDçžċlNēcnéZŔăLŭăLŕăRÑäyĀæŮăăLzăŔăăĒæøÿăyĀăyĭçžċlNæL'gèaÑèfZæuăyĀăyĭæL'gèaÑæĭqăđN  
 çŽDçžċlNæZt' éĀCçTĭāzŌăd'ĐçŘEI/OăŠNăĚŭăzŮéIJĀèçAăzŭăŔŚæL'gèaÑçŽDèYzāqđæ\$■ä;IJiijLærTæČ

æIJLæŮă;ăaijŽçIJNăLŕăyNè;žèfZçğ■éĀŽèfGçzğæL'f Thread  
 çszæİăăđçŎřçŽDçžċlNiiž

```

from threading import Thread

class CountdownThread(Thread):
    def __init__(self, n):
        super().__init__()
        self.n = n

    def run(self):
        while self.n > 0:

            print('T-minus', self.n)
            self.n -= 1
            time.sleep(5)

c = CountdownThread(5)
c.start()

```

threading ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 threading ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 multiprocessing ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ

```

import multiprocessing
c = CountdownTask(5)
p = multiprocessing.Process(target=c.run)
p.start()

```

CountdownTask  
 CountdownTask

## 14.2 12.2 ăĽd'ăĖŭčŕĕĭNăŸŕăŔĕăuŝĉzŔăŔŕăĹĭ

### éŬŏĕćŸ

ăĭăăüŝĉzŔăŔŕăĹăzĒăŸăĀăŸĭĉŕĕĭNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ

### ĕğĉăĒŕăŰăăĹ

çŕĕĭNăĭZĎăŸăĀăŸĭĕŦŏĉĹ'zăăĒăŸŕăŕăŸĭĉŕĕĭNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 threading ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 event ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 event ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 event ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ  
 event ăŕĭçŏăĕfZăăüăzşăŕŕăzëăüĕăĭJĭĭjNăĭĒĕfZăĭĕăĭŬăĭăçZĎăzĉăăAăĭĭĕŬăzŎ

```

from threading import Thread, Event
import time

# Code to execute in an independent thread
def countdown(n, started_evt):
    print('countdown starting')
    started_evt.set()
    while n > 0:
        print('T-minus', n)
        n -= 1
        time.sleep(5)

# Create the event object that will be used to signal startup
started_evt = Event()

# Launch the thread and pass the startup event
print('Launching countdown')
t = Thread(target=countdown, args=(10,started_evt))
t.start()

# Wait for the thread to start
started_evt.wait()
print('countdown is running')

```

The code above demonstrates how to use the `Event` object to coordinate between a main thread and a worker thread. The worker thread (`countdown`) starts when the `Event` is set, and the main thread waits for the worker thread to start before printing 'countdown is running'.

## Event

The `Event` object is used to signal the start of a thread. It has a `set()` method to set the event and a `wait()` method to wait for the event to be set.

```

event = Event()
event.set()
event.wait()

```

The `Event` object is also used to signal the end of a thread. It has a `clear()` method to clear the event and a `wait()` method to wait for the event to be cleared.

```

event = Event()
event.clear()
event.wait()

```

The `Event` object is also used to signal the completion of a thread. It has a `set()` method to set the event and a `wait()` method to wait for the event to be set.

```

event = Event()
event.set()
event.wait()

```

```

import threading
import time

class PeriodicTimer:
    def __init__(self, interval):
        self._interval = interval
        self._flag = 0
        self._cv = threading.Condition()

```

```

def start(self):
    t = threading.Thread(target=self.run)
    t.daemon = True

    t.start()

def run(self):
    '''
    Run the timer and notify waiting threads after each interval
    '''
    while True:
        time.sleep(self._interval)
        with self._cv:
            self._flag ^= 1
            self._cv.notify_all()

def wait_for_tick(self):
    '''
    Wait for the next tick of the timer
    '''
    with self._cv:
        last_flag = self._flag
        while last_flag == self._flag:
            self._cv.wait()

# Example use of the timer
ptimer = PeriodicTimer(5)
ptimer.start()

# Two threads that synchronize on the timer
def countdown(nticks):
    while nticks > 0:
        ptimer.wait_for_tick()
        print('T-minus', nticks)
        nticks -= 1

def countup(last):
    n = 0
    while n < last:
        ptimer.wait_for_tick()
        print('Counting', n)
        n += 1

threading.Thread(target=countdown, args=(10,)).start()
threading.Thread(target=countup, args=(5,)).start()

```

eventâržesaçŽDäyÄäyléG■èçAçL'žçCžæYřa;ŠaŏČěcnèő;çjőäyžçIJšæUüaijŽaTd'éEŠæL'ÄæIJLç■L'ă;Ě  
Condition âržesaçælēæŽŁäzčāĀČèĀČèŽŚäyÄäyNèŁŽæŏtă;ŁçTłāŁaāRūéGRăŏdčŎřçŽDžžččāAñijŽ

```
# Worker thread
def worker(n, sema):
    # Wait to be signaled
    sema.acquire()

    # Do some work
    print('Working', n)

# Create some threads
sema = threading.Semaphore(0)
nworkers = 10
for n in range(nworkers):
    t = threading.Thread(target=worker, args=(n, sema,))
    t.start()
```

ěĚŘěąNăyĽè;żçŽDăzččăĀăŕĚăijŽăŔŕăĹăyĂăyĽçžĚĹNăśăiijNăĲăŸŕăžŭăşqăĪĴăžĂăžĹăžNăĈĚăŔŚ

```
>>> sema.release()
Working 0
>>> sema.release()
Working 1
>>>
```

çijŨăĚŽăŭĹăŔĹăĹăŕăđ'ğéĠŔçŽĐçžĚĹNéŨŕ'ăŔNă■ēŮőécŸçŽDăzččăĀăijŽēŏĹăĲăçŮŽăy■ăēŋçŦşăŐ

## 14.3 12.3 çžĚĹNéŨŕ'ėĂŽăĚă

### ėŮőécŸ

ăĲăçŽĐĹNăžŔăy■ăĪĴăđ'ŽăyĽçžĚĹNġijNăĲăēĪĂēĲăĀĪĴēĲŽăžŽçžĚĹNăžNéŨŕ'ăŏĹăĚĹăĪŕăžđ'ă■ăăĲăă

### ėğăĲăĚşăŮžăăĹ

ăžŐăyĂăyĽçžĚĹNăŔŚăŔēăyĂăyĽçžĚĹNăŔŚēĂăĲăŦŕă■ŏăĪĂăŏĹăăĚĹçŽĐăŮžăijŔăŔŕēĈĲăŕśăŸŕăĲçŦĴ  
 queue                     ăžŞăy■çŽĐēŸşăĹŮăžĚăĂĈăĹŽăžžăyĂăyĽēĉăăđ'ŽăyĽçžĚĹNăĚśăžŋçŽĐ  
 Queue             ăŕžēşăiijNēĲŽăžŽçžĚĹNéĂŽēĲĠăĲçŦĴ             put()             ăŞŦ             get()  
 æŞ■ăĲăĪăĲăŔŚēŸşăĹŮăy■ăŭžăĹăăĹŮēĂĚăĹăēŽđ'ăĚĈçŦ'ăăĂĈăĲNăēĈġijŽ

```
from queue import Queue
from threading import Thread

# A thread that produces data
def producer(out_q):
    while True:
        # Produce some data
        ...
        out_q.put(data)
```

```

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()
        # Process the data
        ...

# Create the shared queue and launch both threads
q = Queue()
t1 = Thread(target=consumer, args=(q,))
t2 = Thread(target=producer, args=(q,))
t1.start()
t2.start()

```

Queue áržesqáũščžRăÑĚăRñăžEăfĚëeAçŽDěTĀijŇæL'Ăäžěä;ăăRřăžěéĂŽëŁGăóČăIJlăd'ŽăyłçžŁçlŃé  
 ă;Šă;ŁçŤlėYšăLŮăŮüijŇă■RërČçŤšăžgèĂĚăŠŇæŭLèt'žèĂĚçŽDăĚséŮ■ėŮóécŸăRřèČ;ăijŽæIJL'ăyĂäžŽé

```

from queue import Queue
from threading import Thread

# Object that signals shutdown
_sentinel = object()

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        out_q.put(data)

    # Put the sentinel on the queue to indicate completion
    out_q.put(_sentinel)

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()

        # Check for termination
        if data is _sentinel:
            in_q.put(_sentinel)
            break

        # Process the data
        ...

```

æIJnăĶNăy■æIJL'ăyĂăyłçL'žæóŁçŽDăIJræŮžiiJŽæŭLèt'žèĂĚăIJlėfzăLřèŁŽăyłçL'žæóŁăĂijăžŇăŘŮçnŃ



är;çøæÿſåĹŮæŸřæIJĀăÿÿèġAçŽĐçžŁćĹŃéŮťéĂŽăřææIJžăĹŭijNăĲEæŸřăž■çĐŭăŔřăžèèĠăŭséĂŽèŁĠăĹŽ  
ConditionăŔŸéĠŔăĬăŃĚèĉĚăĲăçŽĐăŦřă■őçžſæđĐăĂCăÿŃèĲžèŁŽăÿĹăĲŃă■ŔăĲĲčđ'žăžEăęĆăĲŦăĹŽ

```
import heapq
import threading

class PriorityQueue:
    def __init__(self):
        self._queue = []
        self._count = 0
        self._cv = threading.Condition()
    def put(self, item, priority):
        with self._cv:
            heapq.heappush(self._queue, (-priority, self._count, _
→item))

            self._count += 1
            self._cv.notify()

    def get(self):
        with self._cv:
            while len(self._queue) == 0:
                self._cv.wait()
            return heapq.heappop(self._queue)[-1]
```

ăĲŁçŦĬéŸſåĹŮæĬèēŁZèăŃçžŁćĹŃéŮťéĂŽăřææŸřăÿĂăÿĹă■ŦăŔŖſăĂĂăÿ■çăőăőŽçŽĐèŁĠăĹŽăĂĆéĂŽăÿ  
task\_done()ăŖŇ join()ĲĲŽ

```
from queue import Queue
from threading import Thread

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        out_q.put(data)

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()

        # Process the data
        ...
        # Indicate completion
        in_q.task_done()

# Create the shared queue and launch both threads
q = Queue()
t1 = Thread(target=consumer, args=(q,))
```

```

t2 = Thread(target=producer, args=(q,))
t1.start()
t2.start()

# Wait for all produced items to be consumed
q.join()

```

æĈædĪjāyÄäylçžŁçłŃéĪĖĕĖAłĪlāyÄäylāĀĪæŭĹet'zèĀĖâĀĬçžŁçłŃâd'ĎçŘĖăŏŇçĹ'zâŏŽçŽĎæŢræ■ŏ  
 Event æŢçłĹrāyĀètŭä;ŁçłŢĹijŃĕŁZæăŭâĀĪĴŢšăžgèĀĖâĀĬăřsăŖrăžĕĖĀŽĕŁĜĕŁZäylEventărzèsæĬççŽŚætĹ

```

from queue import Queue
from threading import Thread, Event

# A thread that produces data
def producer(out_q):
    while running:
        # Produce some data
        ...
        # Make an (data, event) pair and hand it to the consumer
        evt = Event()
        out_q.put((data, evt))
        ...
        # Wait for the consumer to process the item
        evt.wait()

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data, evt = in_q.get()
        # Process the data
        ...
        # Indicate completion
        evt.set()

```

## ëŏĹëŏž

âşzăžŎçŏĀă■ŢĕŸşăĹŪçijŪăĖZăd'ŽçžŁçłŃçłŃăžŖăĪĹăd'ŽæŢræĈĖăĖŢăyŃæŸrāyÄäylæfŢĕçĈæŸŎæŽ  
 ä;ŁçłŢĹçžŁçłŃéŸşăĹŪæĪĹ'äyÄäylĕĕAæşĹăĎŖçŽĎĕŪŏĕçŸæŸŕijŃăŖŚĕŸşăĹŪäy■æŭzăĹăæŢræ■ŏéqzæŪŭă

```

from queue import Queue
from threading import Thread
import copy

# A thread that produces data
def producer(out_q):
    while True:
        # Produce some data
        ...

```

```

        out_q.put(copy.deepcopy(data))

# A thread that consumes data
def consumer(in_q):
    while True:
        # Get some data
        data = in_q.get()
        # Process the data
        ...

```

Queue řřžšæŘŘä;ŽäYÄäZŽäIJlä;ŠäL■äYläYÑæŮGä;LæIJLčTlčŽDěŽDäŁäçL'žæÄgäÄĆæfTäeĆäIJ  
 Queue řřžšæŮüæŘŘä;ŽäRréÄLčŽD size äRCæTřæIěéŽRäLŮäRřäžěæüžäŁääLřéYšäLŮäY■čŽDäĚČčt'ä  
 äÄIJæŮLèt'žäÄIčŽDěÄšäžęäfnijÑéČčäZLä;ŁçTlāZžäōŽäd'gärRčŽDěYšäLŮäřšäRřäžěäIJléYšäLŮäũšæžæç  
 get() äŠÑ put() æŮžæšTéČ;æTřæÑÄéIdéYžäąđæŮžäijRäŠÑěō;äōŽěüĚæŮüijNä;NäeĆijŽ

```

import queue
q = queue.Queue()

try:
    data = q.get(block=False)
except queue.Empty:
    ...

try:
    q.put(item, block=False)
except queue.Full:
    ...

try:
    data = q.get(timeout=5.0)
except queue.Empty:
    ...

```

ěŁŽäZžæŠ■ä;IJéČ;äRřäžěçTlæIěéAŁäĚ■ä;ŠæL'gèaÑæšŘäžZçL'žäōŽéYšäLŮäŠ■ä;IJæŮüäRŠçTšæŮäe  
 put() æŮžæšTäŠÑäYÄäYłäZžäōŽäd'gärRčŽDěYšäLŮäYÄètüä;ŁçTlñijÑeŁZæäüä;ŠéYšäLŮäũšæžæŮüäřš

```

def producer(q):
    ...
    try:
        q.put(item, block=False)
    except queue.Full:
        log.warning('queued item %r discarded!', item)

```

äeĆädIJä;äerTäŽ;èol'æŮLèt'žèÄĚčžŁçlNäIJläL'gèaÑäČR q.get()  
 ěŁZæäüçŽDäŠ■ä;IJæŮüijÑeüĚæŮüèGłäLlčZLæ■căžěä;ŁæčÄæšęçZLæ■căäGäŁŮijNä;äāžTēřä;ŁçTl  
 q.get() čŽDäRréÄL'äRCæTř timeout ïijNäeĆäYÑijŽ

```

_running = True

def consumer(q):

```

```

while _running:
    try:
        item = q.get(timeout=5.0)
        # Process item
        ...
    except queue.Empty:
        pass

```

æIJĀāŔŌīījŊæIJL q.qsize() īījŊ q.full() īījŊ q.empty()  
 ç■LāōđçŦĭæŪzæşŦāŔŕāzèèŌūāŔŪāyĀāyĭéYşāLŪçŽĐā;ŞāL■ād'gārŔāŦŦçLūæĀĀāĀĆă;ĒēçĀæşĭæĐŔīījŊæ  
 empty() āLđ'æŪ■āGžèçZāyĭéYşāLŪāyžçŦ'zīījŊā;ĒāŔŊæŪūāŔēād'ŪāyĀāyĭçžççĭŊāŔŕēÇ;āušçzŔāŔŦēçZā

## 14.4 12.4 çŻŽāĒşĕŦŏéČĭāĹĒāŁăéŦĀ

éŬŏéćŸ

äĵäēIJĀēçĀāŕzād'ŽçžççĭŊçĭŊāzŔāy■çŽĐāyt'çŦŊāŦzāŁăéŦĀäzèéĀçāĒ■çndāžL'æĭāzūāĀĆ

èğčāĒşæŪzæāĹ

èçĀāIJĭād'ŽçžççĭŊçĭŊāzŔāy■āōL'āĒĭā;ççŦĭāŔŕāŔŸāŕzèşāīījŊā;äēIJĀēçĀā;ççŦĭ thread-  
 ing āžŞāy■çŽĐ Lock āŕzèşāīījŊāŕşāČŔāyŊēçžèçZāyĭā;Ŋā■ŔèçZæāūīījŽ

```

import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    def __init__(self, initial_value = 0):
        self._value = initial_value
        self._value_lock = threading.Lock()

    def incr(self, delta=1):
        '''
        Increment the counter with locking
        '''
        with self._value_lock:
            self._value += delta

    def decr(self, delta=1):
        '''
        Decrement the counter with locking
        '''
        with self._value_lock:
            self._value -= delta

```

Lock árzèsaǎŠŇ with èr■āRēāĪŪäyĀetūā;ŁçŦĪāRřäzèäŁİērAāžŠæŪēæL'gèaŇĭĭjŇāřsæŸřæfRæñqāRĪæĪ  
with èr■āRēāŇĒāRŋçŽDāžčçāAāĪŪāĀĆwith èr■āRēāĭjŽāĪĪĪēŁŽäyĪāžčçāAāĪŪæL'gèaŇāL'■ēĠāĪĪēŌŭāRŪēŦ

## ěóĪēőž

čžŁçĪŇērČāžçæĪĤèt'ĪäyŁæŸřäy■çāőāőŽçŽĎĭĭjŇāŽāæ■d'ĭĭjŇāĪĪĪād'ŽčžŁçĪŇçĪŇāžRäy■éŦŽērřāĪĪřā;ŁçŦ  
āĪĪäyĀāžŽāĀĪĪēĀAçŽĎāĀĪ Python āžčçāAäy■ĭĭjŇæŸ;āĭjRēŌŭāRŪāŠŇēĠŁæŦ;éŦAæŸřā;ĪäyŸēġAçŽĎāĀ

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    def __init__(self, initial_value = 0):
        self._value = initial_value
        self._value_lock = threading.Lock()

    def incr(self, delta=1):
        '''
        Increment the counter with locking
        '''
        self._value_lock.acquire()
        self._value += delta
        self._value_lock.release()

    def decr(self, delta=1):
        '''
        Decrement the counter with locking
        '''
        self._value_lock.acquire()
        self._value -= delta
        self._value_lock.release()
```

čŽyærŦāžŌēŁŽçġ■æŸ;āĭjRērČçŦĪčŽĎæŪzæšŦĭĭjŇwith èr■āRēæŽŦ'āŁāāĭjŸēŽĒĭĭjŇāžšæŽŦäy■āőzæŸš  
release() æŪzæšŦæĪŪēĀĒçĪŇāžRāĪĪĪēŌŭā;ŪēŦAāžŇāRŌāžġçŦšāĭjČāyŸēŁŽäyĎ'çġ■æČĒāĒĭĭjĪä;ŁçŦĪ  
with èr■āRēāRřäzèäŁİērAāĪĪĪēŁŽäyĎ'çġ■æČĒāĒĭĭjŇāž■ēČ;æ■ççāőēĠŁæŦ;éŦAĭĭjĪāĀĆ  
äyžāžĒēAŁāĒ■āĠçŦŌřæ■zéŦAçŽĎæČĒāĒĭĭjŇā;ŁçŦĪēŦAæĪĪžāĪŪčŽĎçĪŇāžRāžŦērēēő;āőŽäyžærRäyŁçžŁçĪ  
āĪĪĪ threading āžšäy■ēŁŸæRŘā;ŽāžĒāĒŪāžŪčŽĎāRŇæ■ēāŌšēr■ĭĭjŇærŦāēČ RLock  
āŠŇ Semaphore árzèsaǎĀĆā;ĒæŸřæāžæ■őāžēā;ĀçžRēĪŇĭĭjŇēŁŽāžŽāŌšēr■æŸřçŦĪāžŌäyĀāžŽçŁ'zæőŁçŽ  
RLock ĭĭjĪāRřēĠ■āĒēēŦAĭĭjĪāRřäzèēčŇāRŇäyĀäyŁçžŁçĪŇāđ'ŽæñæŌŭāRŪĭĭjŇäyžēēAçŦĪæĪēāőđçŦŌřāšžāž  
SharedCounter çšžĭĭjŽ

```
import threading

class SharedCounter:
    '''
    A counter object that can be shared by multiple threads.
    '''
    _lock = threading.RLock()
```

```

def __init__(self, initial_value = 0):
    self._value = initial_value

def incr(self, delta=1):
    '''
    Increment the counter with locking
    '''
    with SharedCounter._lock:
        self._value += delta

def decr(self, delta=1):
    '''
    Decrement the counter with locking
    '''
    with SharedCounter._lock:
        self.incr(-delta)

```

aIJläyŁèŁ zèŁŻäyŁä; Nā■Räy■rijNæšqæIJL'ärzæfRäyÄäyŁäōđäŁNäy■çŽDāRfāRŸärzèšqāŁäēTÄrijNāRŪē.  
 decr æŪzæšTāĀĆ èŁŽçg■āōđçŌræŪzāijRçŽDäyÄäyŁçL'zçCzæŸrijNæŪæōžèŁŻäyŁçšzæIJL'ād'ŽārSäyŁäōđä.  
 äŁqāRŪēGRärzèšqæŸrāyÄäyŁäžžçñNāIJlāĒšāžñèōqæTŗāŽlāšžçqāÄäyŁçŽDāRNæ■ēāŌšèr■āĀĆāçCæđIJèōqæ.  
 èr■āRēārĒèōqæTŗāŽlāGRlrijNçžŁçlNècñāĒĒèōyæL'gèqNāĀĆwith  
 èr■āRēæL'gèqNçzšæIšāRŌrijNèōqæTŗāŽlāLārijSāĀĆāçCæđIJèōqæTŗāŽlāyž0rijNçžŁçlNārĒècñēŸzāqđrijNçz

```

from threading import Semaphore
import urllib.request

# At most, five threads allowed to run at once
_fetch_url_sema = Semaphore(5)

def fetch_url(url):
    with _fetch_url_sema:
        return urllib.request.urlopen(url)

```

āçCæđIJä;āärzçžŁçlNāRNæ■ēāŌšèr■çŽDāžTāsĆçRĒèōžāŠNāōđçŌræDšāĒr'eüçrijNārřzēāRCèĀĆæš

## 14.5 12.5 éŸšæ■cæ■zéTĀçŽDāŁäēTĀæIJžāLŪ

### éŪōécŸ

äjāæ■cāIJlāĒZäyÄäyŁād'ŽçžŁçlNçlNāžRrijNāĒŪäy■çžŁçlNéIJĀèçAäyÄæñæēŌūāRŪād'ŽäyŁēTÄrijNæ■

### èğčāEšæŪzæqĹ

aIJlād'ŽçžŁçlNçlNāžRäy■rijNæ■zéTĀéŪōécŸāŁād'gäyĀéČlāŁĒæŸřçTšāžŌçžŁçlNāRNæŪūēŌūāRŪā.  
 æŪūāĀŽāRŠçTšēŸzāqđrijNèČčāzŁēŁŻäyŁçžŁçlNāršāRřèC;ēŸzāqđāĒūāžŪçžŁçlNçŽDæL'gèqNrijNāžŌēĀNā.  
 èğčāEšæ■zéTĀéŪōécŸçŽDäyĀçg■æŪzæqĹæŸrāyžçlNāžRäy■çŽDæfRäyÄäyŁēTĀāŁĒēĒāyÄäyŁāTŗāyĀçŽ.  
 æŸřēlđäyŸāōžæŸšāōđçŌrçŽDrijNçd'žäŁNāçCäyNrijŽ

```

import threading
from contextlib import contextmanager

# Thread-local state to store information on locks already acquired
_local = threading.local()

@contextmanager
def acquire(*locks):
    # Sort locks by object identifier
    locks = sorted(locks, key=lambda x: id(x))

    # Make sure lock order of previously acquired locks is not
    ↪violated
    acquired = getattr(_local, 'acquired', [])
    if acquired and max(id(lock) for lock in acquired) >= ↪
    ↪id(locks[0]):
        raise RuntimeError('Lock Order Violation')

    # Acquire all of the locks
    acquired.extend(locks)
    _local.acquired = acquired

    try:
        for lock in locks:
            lock.acquire()

        yield

    finally:
        # Release locks in reverse order of acquisition
        for lock in reversed(locks):
            lock.release()
        del acquired[-len(locks):]

```

æCä;Tä;ŁçTłēŁZäyŁayŁäyNæŮĠçóaçŘĚāZlāŚcījśā;āāRfāzēæŃŁçĚğæ■čāyŷéĀTā;ĎāŁZāzzāyĀäyŁēŤ  
 acquire() āĠ;æTřæĬčTšēŕuēŤĀiijŃ çd'žä;NæĆäyŃiijŽ

```

import threading
x_lock = threading.Lock()
y_lock = threading.Lock()

def thread_1():
    while True:
        with acquire(x_lock, y_lock):
            print('Thread-1')

def thread_2():
    while True:
        with acquire(y_lock, x_lock):
            print('Thread-2')

t1 = threading.Thread(target=thread_1)

```

```

t1.daemon = True
t1.start()

t2 = threading.Thread(target=thread_2)
t2.daemon = True
t2.start()

```

æĈædIJä;äæL'gëaÑefŽæoĵäzčĉăAġijŃä;äaijŽăRŚçŎřăoČă■sä;ġăIJläy■ăRŃçŽDăĠ;æTřäy■ăzëäy■ăRŃç  
 äĚüăĚšéŤoăIJläžŎġijŃăIJlčňňäyĂæoĵäzčĉăAäy■ġijŃăĹšăznărzeġŽăžŽéŤAèġŽëaŃăžEæŎšăžRăĂĆéĂŽēġ  
 æĈædIJæIJL'ăd'Žăyġ acquire() æš■ă;IJecnăŧŃăeŮerČçŤġijŃăRřăžëéĂŽēġĠçġġŃăIJŃăIJřă■ŸăČġijġLT  
 äĂĠèŏ;ă;ăçŽDăžčĉăAæŸrēġŽæăăăEŽçŽĎġjŽ

```

import threading
x_lock = threading.Lock()
y_lock = threading.Lock()

def thread_1():

    while True:
        with acquire(x_lock):
            with acquire(y_lock):
                print('Thread-1')

def thread_2():
    while True:
        with acquire(y_lock):
            with acquire(x_lock):
                print('Thread-2')

t1 = threading.Thread(target=thread_1)
t1.daemon = True
t1.start()

t2 = threading.Thread(target=thread_2)
t2.daemon = True
t2.start()

```

æĈædIJä;äæfŘëaÑefŽăyġġL'ĹăIJŃçŽDăžčĉăAġijŃăfĚăŏŽăijŽăIJL'ăyĂăyġġçġġġŃăRŚçŤšăŧŤ'æžČġijŃă

```

Exception in thread Thread-1:
Traceback (most recent call last):
  File "/usr/local/lib/python3.3/threading.py", line 639, in _
↳bootstrap_inner
    self.run()
  File "/usr/local/lib/python3.3/threading.py", line 596, in run
    self._target(*self._args, **self._kwargs)
  File "deadlock.py", line 49, in thread_1
    with acquire(y_lock):
  File "/usr/local/lib/python3.3/contextlib.py", line 48, in __
↳enter__

```



[illegible]

æ■zēTāæYrærRāyĀäylād'ŽčžčlNčlNāžRéČ;āijZēlcāyt'čŽDāyĀäylēUōécYijLārsāČRāōČæYrærRāyĀ  
čžčlNāRlēČ;āRŅæUūāfIæNāyĀäylēTāijNēfZāūcłNāžRārsāy■āijZēcna■zēTāēUōécYēL'ĀāZrēL'rāĀ

éAǻǻē■zeŦAæYŕaReǻd'ŨäyǻÇg■ègĉaEşæ■zeŦAéŨoécYçZĐæŨzajŕiijNǻIJeſZçlNèŖuǻRŨéŦAçZ  
æ■zeŦAçŁuǻĀĀǻĀĆerAæYŖǻrşçŦZçzZerzeĀĒǻIjǻyçzçCǻzǻǻǻEǻĀĆeAǻǻē■zeŦAçZĐäyçzèeAæĀĀĆşǻ  
æ■zeŦAçZĐäyĀǻyĻǻſEēeAæIǻǻzũiiijNǻzŖēĀĒéAǻǻē■cĻNǻzRēſZǻĒēēæ■zeŦAçŁuǻĀĀǻĀĆ

```
import threading

# The philosopher thread
def philosopher(left, right):
    while True:
        with acquire(left, right):
            print(threading.currentThread(), 'eating')

# The chopsticks (represented by locks)
NSTICKS = 5
chopsticks = [threading.Lock() for n in range(NSTICKS)]

# Create all of the philosophers
for n in range(NSTICKS):
    t = threading.Thread(target=philosopher,
                        args=(chopsticks[n], chopsticks[(n+1) %
NSTICKS]))
    t.start()
```

æIJĀāRŌiijNēēAçL'zāLŋsłāĎRĀlRiijNäyžāžEéAçĀĒ■zeŤAiiijNæL'ĂæIJL'çŽDāŁæĤAæş■ăiIJāfĒÉ  
acquire()      āGjæTrāĀCāeĆædIJäzçcāAäy■çŽDæşŘēĆĀlĒEçzTēfĠacquire

ǎĜĵæTřčŽt'æŎěčTřšèrúéTǎĵĵNěĆčázŁæTř'äyłæ■zéTǎéAŁăĚ■æIJžǎŁŭǎřsäy■èĵuǎĵIJčTřlázĚǎĀĆ

## 14.6 12.6 äŖlǎ■ŸčžŖčlŇčŽĐčŁŭæĀAăŖæAř

éŮőéčŸ

ǎĵǎéIJǎèĕAǎłlǎ■Ÿæ■čǎIJlèŁRèǎNčžŖčlŇčŽĐčŁŭæĀAřĵNèŁŽǎyłčŁŭæĀAřžázŎǎĚŭázŮčŽĐčžŖčlŇæŸ

èĝčǎĚşæŮžæǎĹ

æIJŁæŮŭǎIJǎđ'ŽčžŖčlŇčĵŮčlŇǎy■ĵĵNǎĵǎéIJǎèĕAǎłlǎłǎ■ŸǎĵŞǎŁ■èŁRèǎNčžŖčlŇčŽĐčŁŭæĀAăĀĆ  
èĕAĕŁŽázŁǎAŽĵĵNǎRřǎĵčTřlthread.local()ǎŁŽǎžžǎyĀǎyłæIJňǎIJřčžŖčlŇǎ■ŸǎĆlǎřžèşǎĀĆ  
ǎřžèŁŽǎyłǎřžèşǎčŽĐǎśđæĀĝčŽĐǎłlǎ■ŸǎŠNěřžǎRŮæŞ■ǎĵIJéČĵǎRłǎĵŽǎřžæŁĝèǎNčžŖčlŇǎRřèĝAĵĵNěĀNǎĚ

ǎĵIJǎyžǎĵčTřlǎIJřǎ■ŸǎĆlčŽĐǎyĀǎyłæIJŁèŭččŽĐǎóđéŽĚǎĴNǎ■RĵĵN  
èĀĆčŽŚǎIJŁ8.3ǎRŘèŁČǎóŽázŁèŁĜčŽĐLazyConnectionǎyŁǎyNæŮĜčóǎčŘĚǎŽlčśžǎĀĆ  
ǎyNéłčǎŁŚǎžňǎřžǎóČĕŁŽèǎNǎyĀǎžŽǎRčŽĐǎłŕǎTžǎĵǎĵŮǎóČǎRřǎžééĀĆčTřlázŎǎđ'ŽčžŖčlŇĵĵŽ

```
from socket import socket, AF_INET, SOCK_STREAM
import threading

class LazyConnection:
    def __init__(self, address, family=AF_INET, type=SOCK_STREAM):
        self.address = address
        self.family = AF_INET
        self.type = SOCK_STREAM
        self.local = threading.local()

    def __enter__(self):
        if hasattr(self.local, 'sock'):
            raise RuntimeError('Already connected')
        self.local.sock = socket(self.family, self.type)
        self.local.sock.connect(self.address)
        return self.local.sock

    def __exit__(self, exc_ty, exc_val, tb):
        self.local.sock.close()
    del self.local.sock
```

ǎžččǎAǎy■ĵĵNěĜlǎŭşĕĝČǎřşǎřžázŎself.localǎśđæĀĝčŽĐǎĵčTřlǎĀĆ  
ǎóČĕčńǎŁlǎĝNǎNŮǎřǎyĀǎyłthreading.local()ǎóđǎĴNǎĀĆ  
ǎĚŭázŮæŮžæşTǎŞ■ǎIJĕčńǎ■ŸǎĆlǎyžself.local.sockčŽĐǎéŮæŎĕǎ■ŮǎřžèşǎĀĆ  
æIJŁǎžĚĕŁŽázŽǎřşǎRřǎžéǎIJǎđ'ŽčžŖčlŇǎy■ǎóŁǎĚłčŽĐǎĵčTřlLazyConnection  
ǎóđǎĴNǎžĚǎĀĆǎĴNǎĕČĵĵŽ

```
from functools import partial
def test(conn):
    with conn as s:
```

```

s.send(b'GET /index.html HTTP/1.0\r\n')
s.send(b'Host: www.python.org\r\n')

s.send(b'\r\n')
resp = b''.join(iter(partial(s.recv, 8192), b''))

print('Got {} bytes'.format(len(resp)))

if __name__ == '__main__':
    conn = LazyConnection(('www.python.org', 80))

    t1 = threading.Thread(target=test, args=(conn,))
    t2 = threading.Thread(target=test, args=(conn,))
    t1.start()
    t2.start()
    t1.join()
    t2.join()

```

aóČázNæL'ÄäzëëaÑä; UéÄŽčŽDăŎšăZăæYřæfRäyłčžŁčłNäijŽăĹZăzzăyÄäyłëĠłăŭsăyŠăsdčŽDăëŬăŎ  
 âŽăæ■d'rijNă; Šăy■ăRŇčŽDčžŁčłNæL'ġëaŇăëŬăŎëă■ŬăŠ■ă; IJæŬürijŇčŤšăžŎăŠ■ă; IJčŽDæYřăy■ăRŇčŽ

## èŏłëőž

âIJłăd'ġëČłăĹEčłNăžRăy■ăĹZăzzăŠŇăŠ■ă; IJčžŁčłNčŁ'žăŏŽčŁŭăĂăăžŭăy■ăijŽæIJL'ăžĂăžĹéŬŏécYă  
 äy■ëŁĠrijNă; ŠăĠžăžEëŬŏécYčŽDæŬăăĂŽrijŇéĂŽăyŷæYřăZăăyžæšRăyłăržëšăëčŇăd'ŽăyłčžŁčłNă; ŁčŤłăĹ  
 æŤăëČăyÄäyłăëŬăŎëă■ŬăĹŬăŬĠăžžŭăĂČă; äăy■ëČ; èŏł' æĹ'ĂæIJL'čžŁčłNèr'ăçŇŏăyÄäyłă■ŤčŇŇăržëšărij  
 âŽăăyžăd'ŽăyłčžŁčłNăRŇăŬŭëržăŠŇăEŽčŽDæŬăăĂŽăijŽăžġčŤšăŭăžšăĂČ  
 æIJŇăIJřčžŁčłNă■YăČłéĂŽëŁ'ëŁŽăžŽëĹDăžRăRłëČ; âIJłëčŇă; ŁčŤłčŽDčžŁčłNăy■ăRřëġĂæĹëëġčăEšëŁŽ

æIJŇëĹČăy■rijNă; ŁčŤł thread.local() âRřăžëëŏł'  
 LazyConnection çšžæŤřæŇĂăyÄäyłčžŁčłNăyÄäyłëŁđæŎërijŇ  
 èĂŇăy■æYřăržăžŎăĹ'ĂæIJL'čŽDëŁŽčłNëČ; âRłæIJL'ăyÄäyłëŁđæŎëăĂČ

âĚŭăŎščRĚæYřrijŇærRăyłthreading.local() âŏđă; ŇăyžærRăyłčžŁčłNčzt'æĹd'čĹĂăyÄäyłă■Ťč  
 æĹ'ĂæIJL'æŽŏéĂŽăŏđă; ŇăŠ■ă; IJærŤăçCèŎăRŬăĂăăŁŏăŤžăŠŇăĹăéŽd'ăĂijăžĚăžĚăŠ■ă; IJëŁŽăyłă■Ŭă  
 æŤRăyłčžŁčłNă; ŁčŤłăyÄäyłčŇŇčŇNčŽDă■ŬăĚyăřšăRřăžëăĹëŤĂæŤřæ■ŏčŽDëŽŤčëžăžĚăĂČ

## 14.7 12.7 âĹZăzzăyÄäyłčžŁčłNăšă

### éŬŏécY

ä;ăăĹZăzzăyÄäyłăŭëă; IJëĂĚčžŁčłNăšărijŇčŤłăĹëčŽyăžŤăŏčăĹŭčŇřëŕŭăšČăĹŬăĹ'ġëaŇăĚŭăžŬčŽDă

### èġčăEšăŬžæăĹ

concurrent.futures âĠ;æŤřăžšæIJL'ăyÄäył ThreadPoolExecutor  
 çšžăRřăžëëčŇčŤłăĹëăŏŇăĹRëŁŽăyłăžžăĹăăĂČ äyŇëĹăæYřăyÄäyłčŏĂă■ŤčŽDŤCPæIJ■ăĹăăŽłijNă; ŁčŤłăž

```

from socket import AF_INET, SOCK_STREAM, socket
from concurrent.futures import ThreadPoolExecutor

def echo_client(sock, client_addr):
    '''
    Handle a client connection
    '''
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')
    sock.close()

def echo_server(addr):
    pool = ThreadPoolExecutor(128)
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        pool.submit(echo_client, client_sock, client_addr)

echo_server((' ', 15000))

```

æĆæđĲă;ăæČşæĹŃăĹăĹZăžă;ăëĠăũşĹĐčžċĹŃăşăĭjŃ  
éĂŽăyyăŔŕăžă;ċĉŤĲăŸăăyiQueueăĲë;žăĲăőđĉŎŕăĂĆăyŃeĲăŸŕăyĂăyiċĹăăŕăăyăăŕŇă;ĲăŸŕăĲŃăĹăĲăă

```

from socket import socket, AF_INET, SOCK_STREAM
from threading import Thread
from queue import Queue

def echo_client(q):
    '''
    Handle a client connection
    '''
    sock, client_addr = q.get()
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')

    sock.close()

def echo_server(addr, nworkers):

```

```

# Launch the client workers
q = Queue()
for n in range(nworkers):
    t = Thread(target=echo_client, args=(q,))
    t.daemon = True
    t.start()

# Run the server
sock = socket(AF_INET, SOCK_STREAM)
sock.bind(addr)
sock.listen(5)
while True:
    client_sock, client_addr = sock.accept()
    q.put((client_sock, client_addr))

echo_server(('', 15000), 128)

```

ä;ŁçTÍ ThreadPooŁExecutor çŽyăržăžŌæL'NăŁlăôđçŌřçŽDăyĂăyŁăě;ăd'DăIŁăžŌăőČă;Łă;Ů  
 äzzăŁăæŔŔăžd'èĂĚăŽt'æŮžă;ŁçŽDăžŌèçnërČçTlăĜ;æŤŕăy■ēŌŭăŔŮēŁŤăŽďăĂijăĂČă;NăēČiijNă;ăăŔŕēČ

```

from concurrent.futures import ThreadPoolExecutor
import urllib.request

def fetch_url(url):
    u = urllib.request.urlopen(url)
    data = u.read()
    return data

pool = ThreadPoolExecutor(10)
# Submit work to the pool
a = pool.submit(fetch_url, 'http://www.python.org')
b = pool.submit(fetch_url, 'http://www.pypy.org')

# Get the results back
x = a.result()
y = b.result()

```

ă;Nă■Ŕăy■ēŁŤăŽđçŽDhandleăržēsăäijŽăyŏă;ăăd'DçŔĚăL'ĂæIJLçŽDēŸzăăđăyŌă■Ŕă;IJiijNçDŭăŔŌă  
 çL'žăŁnçŽDŭijNă.result() æŞ■ă;IJiijŽēŸzăăđēŁŽçlNçŽt'ăŁŕăržăžŤçŽDăĜ;æŤŕăL'ġēăNăőNăŁŔăžŭēŁ

èőlèőž

éĂŽăyŷæİèèőšiiijNă;ăăžŤèŕēéĂfăĚ■çijŮăĚçžŁçlNăŤŕēĜŔăŔŕăžēæŮăéŽŔăŁŭăćđēŤŁçŽDçlNăžŔăĂČ

```

from threading import Thread
from socket import socket, AF_INET, SOCK_STREAM

def echo_client(sock, client_addr):
    '''

```

```

    Handle a client connection
    '''
    print('Got connection from', client_addr)
    while True:
        msg = sock.recv(65536)
        if not msg:
            break
        sock.sendall(msg)
    print('Client closed connection')
    sock.close()

def echo_server(addr, nworkers):
    # Run the server
    sock = socket(AF_INET, SOCK_STREAM)
    sock.bind(addr)
    sock.listen(5)
    while True:
        client_sock, client_addr = sock.accept()
        t = Thread(target=echo_client, args=(client_sock, client_
↪addr))
        t.daemon = True
        t.start()

echo_server((' ', 15000))

```

āŕ;çōāēfZāyīāzšāRfāzēāuēā;IJījN ā;EāYfāōCāy■ēC;āEŁā;āēIJL'āzžēŕTāZ;ēĀZēfGāLZāzžāđ'gēGRçž  
 ēĀZēfGā;fçTīēcDāĒLāLiāgNāNŪçŽDçžfçlNāēšāiijNā;āāRfāzēēō;ç;ōāRŅāŪūēfRēāNçžfçlNçŽDāyLēZŔā  
 ā;āāRfēC;āijŽāĒšāfCāLZāzžāđ'gēGRçžfçlNāijZāēIJL'āzĀāzLāRŌæđIJāĀC  
 çŌŕāzçāS■ā;IJçšžçžšāRfāzēā;Lē;zaēI;çŽDāLZāzžāGāā■CāyIçžfçlNçŽDçžfçlNāēšāāĀC  
 çTŽēGŕījNāRŅāŪūāGāā■CāyIçžfçlNç■L'ā;Ēāuēā;IJāzūāy■āijŽāŕfāĒūāzŪāzççāĀāžgçTšāĀgēC;ā;sāS■āĀ  
 ā;SçDūāzĒījNāēCāđIJāL'ĀāIJL'çžfçlNāRŅāŪūēcŋāTđ'ēEšāzūcŋNā■šāIJICPUāyLāL'gēāNījNēCçāŕšāy■  
 ēĀŽāyīijNā;āāzTēŕēāŔlāIJl/Oāđ'DçŔĒçŽyāĒšāzççāĀāy■ā;fçTīçžfçlNāēšāāĀC

āLZāzžāđ'gçŽDçžfçlNāēšāçŽDāyĀāyīāŔŕēC;ēIJāēēĀāĒšāšçŽDēŪōēcYāYfāĒēā■YçŽDā;fçTīāĀC  
 ā;NāēCījNāēCāđIJā;āāIJIOS XçšžçžšāyLēlčāLZāzž2000āyIçžfçlNījNçšžçžšāY;çđ'žPythonēfZçlNā;fçTīā  
 āy■ēfGījNēfZāyīēōāçōŪēĀŽāyīyāYfāēIJL'ēŕŕāuōçŽDāĀCā;SāLZāzžāyĀāyIçžfçlNāŪūījNāēS■ā;IJçšžçžšāi  
 æT;ç;ōçžfçlNçŽDāL'gēāNāēLīijLēĀŽāyīyāYf8MBāđ'gārRījL'āĀCā;EāYfēfZāyīāĒēā■YāŔāēIJL'āyĀārR  
 āZāē■d'iijNPythonēfZçlNā;fçTīāLŕçŽDçIJšāōđāĒēā■YāĒūāōđā;LārR  
 iijLārTāēCījNāržāzŌ2000āyIçžfçlNāēlēēōšīijNāŔlā;fçTīāLŕāzē70MBçŽDçIJšāōđāĒēā■YīijNēĀNāy■āYf  
 āēCāđIJā;āāNēāfCēŽZāēNšāĒēā■Yāđ'gārRījNāŔfāzēā;fçTī  
 stack\_size() āG;āŦŕāēlēēZ■ā;ŌāōCāĀCā;NāēCījN

```

import threading
threading.stack_size(65536)

```

āēCāđIJā;āāLāāyLēfZāēfāēŕ■āŔēāzūāĒ■āēāēfRēāNāL'■ēlčçŽDāLZāzž2000āyIçžfçlNēŕTēlNījN  
 ā;āāijZāŔSçŌŕPythonēfZçlNāŔlā;fçTīāLŕāzēāđ'gāēC210MBçŽDēŽZāēNšāĒēā■YīijNēĀNçIJšāōđāĒēā■Y  
 æšlāēDRçžfçlNāēLāđ'gārRāfĒēāzēGšāŕSāyž32768ā■ŪēLČījNēĀŽāyīyāYfçšžçžšāĒēā■Yēāŕāđ'gārRījL409

```
# findrobots.py

import gzip
import io
import glob

def find_robots(filename):
    '''
        Find all of the hosts that access robots.txt in a single log_
    ↪file
        '''
    robots = set()
```

```

with gzip.open(filename) as f:
    for line in io.TextIOWrapper(f,encoding='ascii'):
        fields = line.split()
        if fields[6] == '/robots.txt':
            robots.add(fields[0])
return robots

def find_all_robots(logdir):
    '''
    Find all hosts across and entire sequence of files
    '''
    files = glob.glob(logdir+'/*.log.gz')
    all_robots = set()
    for robots in map(find_robots, files):
        all_robots.update(robots)
    return all_robots

if __name__ == '__main__':
    robots = find_all_robots('logs')
    for ipaddr in robots:
        print(ipaddr)

```

aL■éIcçŽDçlNāzRā;ŁçTlāzEēĀŽāyŷçŽDmap-reduceēčŌæāijælēcijŪāEŽāĀĆ āĠ;æTŕ  
 find\_robots() āIJlāyĀāyŁæŪĠāzūāR■ēZEāĀRLāyLāAŽmapæ\$■ā;IJiijNāzūāŕEçz\$ædIJæśĠæĀzāyžāyĀā  
 āz\$āŕsæYŕ find\_all\_robots() āĠ;æTŕāy■çŽD all\_robots éZEāĀRLāĀĆ  
 çŌŕāIJiijNāAĠēō;ā;āæCşēeAāŁōæTžēŁZāyŁçlNāzRēōl'āōČā;ŁçTlād'ŽæāyCPUāĀĆ  
 ā;ŁçōĀā■TāĀTāĀTāŕlélJĀēeAāŕEmap()æ\$■ā;IJæŽŁæ■cāyžāyĀāyŁ  
 concurrent.futures āžŞāy■çTşæŁŖçŽDçşzāijijæ\$■ā;IJā■şāŕŕāĀĆ  
 āyNéIcæYŕāyĀāyŁçōĀā■TāŁōæTžçL'ŁæIJiijŽ

```

# findrobots.py

import gzip
import io
import glob
from concurrent import futures

def find_robots(filename):
    '''
    Find all of the hosts that access robots.txt in a single log_
    ↪file

    '''
    robots = set()
    with gzip.open(filename) as f:
        for line in io.TextIOWrapper(f,encoding='ascii'):
            fields = line.split()
            if fields[6] == '/robots.txt':
                robots.add(fields[0])
    return robots

```



```

def find_all_robots(logdir):
    '''
    Find all hosts across and entire sequence of files
    '''
    files = glob.glob(logdir+'/*.log.gz')
    all_robots = set()
    with futures.ProcessPoolExecutor() as pool:
        for robots in pool.map(find_robots, files):
            all_robots.update(robots)
    return all_robots

if __name__ == '__main__':
    robots = find_all_robots('logs')
    for ipaddr in robots:
        print(ipaddr)

```

éÅžè£Gè£Žäyłä£öæŤzâRÕiijNè£RèaÑè£ŽäyłèDŽæIJnäžğçŤšâRÑæäüçŽDçz\$ædIJiijNä;EæYřâIJłâŽZ.ãóðéŽĚçŽDæĀğëÇ;äijYâNŮæŤLædIJæäzæ■öä;ăçŽDæIJžâŽÍCPUæŤřéGRçŽDäy■ăRÑèĀNäy■ăRÑăĀĆ

## èõłèõž

ProcessPoolExecutor çŽDâĚyăđNçŤłæşŤæĆäyNiižŽ

```

from concurrent.futures import ProcessPoolExecutor

with ProcessPoolExecutor() as pool:
    ...
    do work in parallel using pool
    ...

```

ăĚüăŎşçRĚæYřiijNäyĀäył ProcessPoolExecutor  
 âŁZăžžNäyłçNñçñNçŽDPythonèğçéGLăŽłiijN NæYřçşzçzşşyŁéłcâRřçŤÍCPUçŽDäyłæŤřăĀĆă;ăăRřăžééĀŽ  
 ProcessPoolExecutor(N) æłëăłöæŤž âđ'ĐçRĚăŽłæŤřéGRăĀĆè£Žäyłăđ'ĐçRĚæšăäijŽäyĀçŽt'è£Rèa  
 çĐúăRŎăđ'ĐçRĚæšăèćnăĚşéŮ■ăĀĆäy■è£GriijNçłNăžRăijŽäyĀçŽt'ç■L'ă;ĚçŽt'ăŁræL'ĀæIJL'æRŘăžđ'çŽDă

èćnăRŘăžđ'ăŁræšăäy■çŽDăüëă;IJă£ĚéąžèćnăŏŽăžL'äyžäyĀäyłăĜ;æŤřăĀĆæIJL'äyđ'çğ■æŮžæşŤăŎžæ  
 âçĆăđIJă;ăæČşèŎl'äyĀäyłăLŮëăłæŎłăřijæLŮäyĀäył map()  
 æŞ■ă;IJăžűëăNæL'ğëăNçŽDëřIiijNăRřă;£çŤł pool.map():

```

# A function that performs a lot of work
def work(x):
    ...
    return result

# Nonparallel code
results = map(work, data)

# Parallel implementation

```

```
with ProcessPoolExecutor() as pool:
    results = pool.map(work, data)
```

ãĖĖad'ŮrijŇä;ääŔřäzë;ĚčŤl pool.submit() æĭæL'ŇâĽĭčŽĎæŔŔäzd' a■TäyĭäzzâĽäijŽ

```
# Some function
def work(x):
    ...
    return result

with ProcessPoolExecutor() as pool:
    ...
    # Example of submitting work to the pool
    future_result = pool.submit(work, arg)

    # Obtaining the result (blocks until done)
    r = future_result.result()
    ...
```

æĖĖadĬJä;æL'ŇâĽĭæŔŔäzd' äyÄäyĭäzzâĽäijŇčzŞæĎĬJæŸřäyÄäyĭ Future  
 åĎäĭŇâĀĈ èĕAeŮâŔŮæĬJÄçzĽçzŞæĎĬJijŇä;æĭJÄeĕAeŕĈçŤĭåĎĈçŽĎ result()  
 æŮzæŞTâĀĈ åĎĈäijŽeŸzâäĎèĚčĬŇçŽŕ' âĽŕçzŞæĎĬJèċñèĚŤâŽĎæĭæĀĈ

æĖĖadĬJäy■æĈşëŸzâäĎijŇä;æĕĚŸâŔřäzë;ĚčŤl äyÄäyĭäŽĎeŕĈâĜ;æŤŕijŇäĭŇâĕĈrijŽ

```
def when_done(r):
    print('Got:', r.result())

with ProcessPoolExecutor() as pool:
    future_result = pool.submit(work, arg)
    future_result.add_done_callback(when_done)
```

äŽĎeŕĈâĜ;æŤŕæŮeâŔŮäyÄäyĭ Future åĎäĭŇijŇèċñçŤĭæĭeëŮâŔŮæĬJÄçzĽçzŞæĎĬJijĽæŕŤæĖ  
 âŕ;çĕâĎ'ĎĈŔĖæşâäĭĽåĎzæŸŞä;ĚčŤl ijŇâĬĭèĕ;èĕâĎ' ĝĈĬŇâžŔçŽĎæŮüâĀŽeĚŸæŸŕæĬĽ'âĭĽâĎ'ŽeĬJÄeĕAæ

- èĚčĝ■åzüèqŇâĎ'ĎĈŔĖæĽĀæĬŕâŕĭæĀĈçŤĭäžŮeĈçäžŽâŔřäzëèċñâĽĖèĝçäyžâžŞçŽyçŇñçñŇéĈĭâĽĖçŽ
- èċñæŔŔäzd'çŽĎäzzâĽäâĚĖéazæŸŕçĕĀâ■ŤâĜ;æŤŕä;ĕäijŔâĀĈĀŕzâžŮæŮzæŞTâĀAĕŮ■âŇĖâŖŇâĖüâzĽ
- âĜ;æŤŕâŔĈæŤŕâŖŇèĚŤâŽĎâĬijâĚĖéazâĖĭjâĎžpickleijŇâŽäyžèĕAä;ĚčŤĭâĽŕèĚčĬŇéŮŕ'çŽĎéĀŽæĚäij
- èċñæŔŔäzd'çŽĎäzzâĽäâĜ;æŤŕäy■âžŤâĽĭçŤŹçĽüæĀAæĽŮæĬĽ'âĽŕäĬJçŤĭâĀĈéŽĎ' äžĖæĽŸâ■ŕæŮëâ  
 äyÄæŮëâŔŕâĽä;ääy■èĈ;æŮĝâĽüâ■ŔèĚčĬŇçŽĎäzzâ;ŤëâŇäyžijŇâŽâæ■Ď' æĬJâæ;æĚĭæŇAçĕĖĀâ■ŤâŖ
- âĬĬUnixäyĽèĚčĬŇæşâeĀŽèĚĖeŕĈçŤĭ fork() çşçzçşèŕĈçŤĭèċñâĽäžzijŇ

åĎĈäijŽâĖŇéŽEPythonèĝçĕĖĽâŽĭijŇâŇĖæŇñforkæŮüçŽĎæĽ'ÄæĬĽ'çĬŇâžŔçĽüæĀAäĀĈ  
 èĀŇâĬĬWindowsäyĽijŇâĖŇéŽEèĝçĕĖĽâŽĭæŮüäy■äijŽâĖŇéŽEçĽüæĀAäĀĈ åĎĎéŽĖçŽĎ-  
 forkæŞ■äĬJäijŽâĬĭçñäyÄæŇæŕĈçŤĭ pool.map() æĽŮ pool.submit()  
 âŔŮâŔŖçŤŖâĀĈ

- âĭŞä;æüââŔĽä;ĚčŤĭèĚčĬŇæşââŖŇâĎ'ŽçžĚçĬŇçŽĎæŮüâĀŽèĕAçĽ'zâĽŇâŕŔâĚĈâĀĈ

ä;äâžTèrëaIJlälZázžäzä;TçžŁçlNázNäl■āĒLāLZāžžāžūāēĀæt'zèŁŻçlNæsāiijLærTāēCāIJlçlNázRāRf

## 14.9 12.9 PythonçŽĎāĒlāsĀéTĀēUóécŸ

### éUóécŸ

ä;äâžšçžRāRñèrt'èŁĠāĒlāsĀèġcéĠLāZlēTĀGILiijNæNĒāŁCāōČaijŽā;sāŠ■āLřād'ŽçžŁçlNçlNázRçŽĎāē

### èġcāEşæÚzæaŁ

ār;çōaPythonāōNāĒlæTřæNĀād'ŽçžŁçlNçijŮçlNriijN ä;EæYřèġcéĠLāZlçŽĎCèr■ēlĀāōđçŎřéČlāĒēāIJl  
āōđéŽĒäyŁiijNèġcéĠLāZlècnāyĀäyĪāĒlāsĀèġcéĠLāZlēTĀāfĪāŁd'çlĀiijNāōČçāōāfĪāžžā;TæŮūāĀŽéČ;āR  
GILæIJĀād'ġçŽĎéUóécŸārsæYřPythonçŽĎād'ŽçžŁçlNçlNázRāžūāy■ēČ;lĀl'çTlād'ŽæyCPUçŽĎäijYāŁē  
iijLærTāēCāyĀäyĪā;ŁçTlāžEāđ'ŽäyŁçžŁçlNçŽĎēōaçōŮārEēZEāđNçlNázRāRĪaijŽāIJlāyĀäyĪā■TCPUāyŁēlç

āIJlēōlèōžæŽōēĀŽçŽĎGILāžNāl■iijNæIJLāyĀçČžèēAāijžèrČçŽĎæYřGILāRĪaijŽā;sāŠ■āLřéCčāžŽāy  
āēČādIJā;āçŽĎçlNázRāđ'ġéČlāĒēāRĪaijZæŮLāRĀLĀĪ/OiijNærTāēČç;ŠçžIJāžd'āžŠiijNéCčāžLā;ŁçTlād'Žç  
āŽāyžāōČāžnād'ġéČlāĒēāŮūēŮr'ēČ;āIJlç■L'ā;ĒāĀČāōđéŽĒäyŁiijNā;āāōNāĒlāRřāžæēT;āŁççŽĎāLZāžžā  
çŎřāžčæŠ■ā;IJçšçžçšēŁRēāNēŁZāžLād'ŽçžŁçlNæsāēIJLāžžā;TāŎNāLZiijNæsāāTēāRřæNĒāŁCçŽĎāĀČ

ēĀNāržāžŎā;ĪèŮCPUçŽĎçlNázRiijNā;āēIJĀēēAāijDāyĒæēŽæLġēāNçŽĎēōaçōŮçŽĎçLžçČžāĀČ  
ā;NāēČiijNāijYāNŮāžTāsČçōŮæšTēēAærTā;ŁçTlād'ŽçžŁçlNēŁRēāNāfnā;Ůād'ŽāĀČ  
çšžāiijçŽĎiijNçTšāžŎPythonæYřèġcéĠLæLġēāNçŽĎiijNāēČādIJā;āārEēČčāžZæĀġēČ;çŠūécĹāžčçāAçġžā  
ēĀšāžēāžšāijZæRŘā■ĠçŽĎā;ĹāfnāĀČāēČādIJā;āēēAæŠ■ā;IJæTřçžĎiijNéCčāžLā;ŁçTlNumPyēŁZæāūçŽĎ  
æIJĀāRŎiijNā;āēŁYāRřāžēēĀČēZšāyNāĒūāžŮārRēĀLāōđçŎřæŮžæaŁiijNærTāēCPyPyiijNāōČēĀŽēŁĠāy  
iijLāy■ēŁĠāIJlāEŽēŁZæIJnāžççŽĎæŮūāĀŽāōČēŁYāy■ēČ;æTřæNĀPython 3iijLāĀČ

ēŁYæIJLāyĀçČžèēAæšlāēDRçŽĎæYřiijNçžŁçlNāy■æYřāyŠēŮlçTlāēēāijYāNŮæĀġēČ;çŽĎāĀČ  
āyĀäyĪCPUā;ĪèŮādNçlNázRāRřēČ;āijŽā;ŁçTlçžŁçlNāēĪēōaçRēāyĀäyĪāZ;ā;ççTlāēŁūçTŊēlčāĀāyĀäyĪç  
ēŁZæŮūāĀŽiijNĠILāijZāžġçTšāyĀāžZēŮóécŸiijNāžāyžāēČādIJāyĀäyĪçžŁçlNēTŁæIJšæNĀæIJLĠILçŽĎ  
āžNāōđāyŁiijNāyĀäyĪāEŽçŽĎāy■āē;çŽĎCèr■ēlĀāŁr'āsTāijZārijeĠt'èŁŽāyĪēŮóécŸæŽt'āŁāyēēēG■iijN  
ār;çōāžčçāAçŽĎēōaçōŮēČlāĒēāijZærTāžNāl■ēŁRēāNçŽĎæŽt'āfnāžZāĀČ

èrt'āžEēŁZāžLād'ŽiijNçŎřāIJlæČšèrt'çŽĎæYřæŁšāžnæIJLāyđ'çġ■ç■ŮçTēæĪēēġcāEşGILçŽĎçijççČžā  
ēēŮāĒiijNāēČādIJā;āāōNāĒlāūēā;IJāžŎPythonçŎřāČCāy■iijNā;āāRřāžēā;ŁçTl  
multiprocessing āēlāŮāēĪāLZāžžāyĀäyĪēŁZçlNæsāiijN  
āžūāČRā■RāRñād'ĐçRĒāZlāyĀæāūçŽĎā;ŁçTlāōČāĀČā;NāēČiijNāĀĠāēČā;āæIJLāēČāyNçŽĎçžŁçlNázçç

```
# Performs a large calculation (CPU bound)
def some_work(args):
    ...
    return result

# A thread that calls the above function
def some_thread():
    while True:
        ...
        r = some_work(args)
    ...
```

```
# Processing pool (see below for initialization)
pool = None

# Performs a large calculation (CPU bound)
def some_work(args):
    ...
    return result

# A thread that calls the above function
def some_thread():
    while True:
        ...
        r = pool.apply(some_work, (args))
        ...

# Initialize the pool
if __name__ == '__main__':
    import multiprocessing
    pool = multiprocessing.Pool()
```

ɛfZäyɫɛÄZɛfGä;fçTɫäyÄäyɫæŁÄäüǵäŁ'çTɫɛfZçɫNæśäðǵcǎEşǵzEĞILçZĐEŮðécYǎĂĆ  
 ǎ;ŞäyÄäyɫçzƒçɫNæĆşðeAæŁ'gèǎŃCPUǎrEǵEZEǎđNǎüèǎ;IJæŮüiijNǎijZǎrEǵzzǎŁǎǎRŚçzZɛfZçɫNæśǎǎĂĆ  
 çDüǎRŌɛfZçɫNæśǎiijZǎIJǎRǵǎđ'ŮäyÄäyɫɛfZçɫNäy■ǎRfǎŁäyÄäyɫǎ■TçNñçZĐPythonèǵcǎGŁǎZɫǎièǎüèǎ;I  
 ǎ;ŞçzƒçɫNç■Ł'ǎ;ĖçzŞǎđIJçZĐæŮüǎÄZǎijZǎEĞŁæTç;GILǎĂĆǎzüäyTrijNçTśǵzŌèðǎçŮŮǎzzǎŁǎǎIJǎ■TçNñèǵ  
 ǎIJäyÄäyɫǎđ'ZǎäyçşçzçşǎyŁÉɫrijNǎ;ǎiijZǎRŚçŌrɛfZäyɫæŁÄæIJǎRǎfǎzèèðŮ'ǎ;ǎǎ;Łǎè;çZĐǎŁ'çTɫǎđ'ZCPU  
 ǎRǵǎđ'ŮäyÄäyɫèǵcǎEşGILçZĐç■ŮçTǵæYǎ;fçTɫɫæŁ'ǎśTçijŮçɫNæŁÄæIJǎĂĆ  
 äyžèeAæǎǎæĆşǎYǎrǎEèðǎçŮŮǎrEǵEZEǎđNǎzzǎŁǎè;ñçǵçzçZCrijNèüşPythonçNñçNŮiijNǎIJǎüèǎ;IJçZĐæŮü  
 ɛfZǎRǎfǎzèèÄZɛfGǎIJǎCǎzççǎÄy■ǎRŚǎĖäyNÉɫçɛfZǎüçZĐçŁ'zǎðŁǎŌRǎièǎŌNǎŁRiijZ

```
#include "Python.h"
...

PyObject *pyfunc(PyObject *self, PyObject *args) {
    ...
    Py_BEGIN_ALLOW_THREADS
    // Threaded C code
    ...
    Py_END_ALLOW_THREADS
    ...
}
```

æĈæđIJä;ää;£çŦlĀĔüüzŰauēāĔüēō£éŰōCèr■ēlĀiijŊærŦæĈarfzäžŎCythonçŽĐctypesāžŠiijŊä;ääy■ēIJĀ  
 ä;ŊæĈiijŊctypesāIJlērĈçŦlĀCæŰüäijŽèĠlĀLéĠLæŦ;GILāĀĈ

## ëöíëöž

ëöyad' ŽčlNāzRāSŸaIJlélcārzcžŁčlNāĀğēČ;éŮóécŸčŽDæŮūāĀŽiijNēl'nāyLārsāijŽæĀłç;łGILiijNāzĀā  
āĒūāōđēŁŽæūāā■Rād' lāy■āŌŽéAŞāzşād' lād' l'çIJşāzEçČzāĀČ  
ä;IJāyžāyĀäylçIJşāōđčŽDä;Nā■RiijNāIJlād' ŽčžŁčlNčŽDç;ŚçzIJçijŮčlNāy■čēđçgŸčŽD  
stalls āRrēČ;æŸrāZāyžāĒūāzŮāŌşāZāæfTāçCāyĀäyłDNSæşēæL'łāzūāŮūiijNēĀNēuşGILærnāŮāāĒş  
æIJāāRŌā;āçIJşçŽDēIJĀēçAāĒLāŌzæRđæĠCā;āçŽDāzčçāAæŸrāRēçIJşçŽDēčnGILā;śāŞ■āĒlāĀČ  
āRŌNæŮūēŁŸēçAæŸŌçŽ;GILād' gēČlāŁēēČ;āžTēfēāRlāĒşæşlCPUçŽDād' DçRĒēĀNāy■æŸfI/O.

āçČæđIJā;āāĠĒād' Gā;ŁçTlāyĀäylād' DçRĒāZlāēšāiijNāēşlāēDŖçŽDæŸrēŁŽæūāāĀZæūL'āRĒlāĒræTŕæ■  
ēčnāēL'gēāNçŽDæŞ■ā;IJēIJĀēçAæT;āIJlāyĀäylēĀZēŁgđēfē■āRēāōŽāzL'çŽDPythonāĠ;æTŕāy■iijNāy■ēČ;  
āzūāyTāĠ;æTŕāRČæTŕāŠNēŁTāZđāĀijāŁĒēāzēçAāĒiijāōžpickleāĀČ  
āRŌNæūiijNēçAæL'gēāNçŽDāzāŁāēĠRāŁĒēāzēūşād' şād' gāzēāiijēāēēčlād' ŮçŽDēĀZāfāāijĀēTĀāĀČ

āRēād' ŮāyĀäylēŽ;çČzæŸrā;ŞæūūāRĒlā;ŁçTlčžŁčlNāŠNēŁŽčlNāēšçŽDæŮūāĀŽāijŽēōl'ā;āā;Łād' l'çŮ  
āçČæđIJā;āēçAāRŌNæŮūā;ŁçTlāyđ'ēĀĒiijNāIJĀāē;āIJlčlNāzRāRŕāŁlāēŮūiijNāŁZāzžāzā;TçžŁčlNāzNāL'■  
çDūāRŌçžŁčlNā;ŁçTlāRŌNæūçŽDēŁŽčlNāēšāēŁēēŁZēāNāōČāzñçŽDēōaçŮŮārĒēŁēĀđNāūēā;IJāĀČ

CæL'l'āsTæIJĀēĠēçAçŽDçL'zā;AæŸrāōČāzñāŠNPythonēğçēĠLāZlāēŸrāŁlāēNĀçNñçñNçŽDāĀČ  
āzşārşæŸrēf'tiijNāēČæđIJā;āāĠĒād' ĠārĒPythonāy■çŽDāzāŁāāŁēēĒ■āĒlČāy■āŌzæL'gēāNriijN  
ā;āēIJĀēçAçāōāŁĠCāzčçāAçŽDæŞ■ā;IJēūşPythonāŁlāēNĀçNñçñNriijN  
ēŁZārşæĠRāŞşçlĀāy■ēçAā;ŁçTlPythonæTŕæ■ōçşşæđDāzēāRĒlāy■ēçAērČçTlPythonçŽDC  
APIāĀČ āRēād' ŮāyĀäylārsæŸrā;āēçAçāōāŁĠCæL'l'āsTæL'ĀāĀŽçŽDāūēā;IJæŸrēūşād' şçŽDriijNāĀijā;Ůā;ā  
āzşārşæŸrēf't' CæL'l'āsTæNĒēt' şēŭāzĒād' gēĠRçŽDēōaçŮŮāzāŁāiijNēĀNāy■æŸrārşæTŕāĠāyylēōaçŮŮāĀČ

ēŁZāžZēğçāĒşGILçŽDæŮzæāŁāzūāy■ēČ;ēĀČçTlāžŌæL'ĀæIJL'ēŮóécŸāĀČ  
ā;NāēČiijNāēşRāžZçşādNçŽDāžTçTlčlNāzRāēČæđIJēčnāŁēēğçāyžād' ŽāylēŁŽčlNād' DçRĒçŽDēlāzūāy■ē  
āzşāy■ēČ;ārĒāōČçŽDēČlāŁēāzčçāAæTzæŁRČēr■ēlāæL'gēāNāĀČ  
āržāžŌēŁZāžZāžTçTlčlNāzRiijNā;āārşēçAēĠlāūšēIJĀāēšČēğçāĒşæŮzæāŁāžĒ  
riijLārTāçCād' ŽēŁŽčlNēōŁēŮōāĒşāzñāĒēĀ■ŸāNžriijNād' ŽēğçæđRāZlēŁRēāNāžŌāRŌNāyĀäylēŁŽčlNç■L'riijL  
æLŮēĀĒriijNā;āēŁŸārRāzēēĀČēŽŞāyNāĒūāzŮçŽDēğçēĠLāZlāōđčŌriijNārTāçCPyPyāĀČ

āžĒēğçæZt'ād' ŽāĒşāžŌāIJlCæL'l'āsTāy■ēĠLæT;GILiijNērūāRČēĀČ15.7āŠN15.10ārRēŁČāĀČ

## 14.10 12.10 āōŽāzL'āyĀäylActorāzžāŁā

### éŮóécŸ

ā;āæČşāōŽāzL'ēūşactoræłāāijRāy■çşzāiijāĀIactorsāĀlēğŞēL'şçŽDāzāŁā

### ēğçāĒşæŮzæāŁ

actoræłāāijRæŸrāyĀçg■æIJĀāRđ'ēĀAçŽDāžşæŸræIJĀçŌĀ■TçŽDāzūēāNāŠNāŁēāyČāijRēōaçŮŮēğç  
āžNāōđāyŁiijNāōČād' l'çTşçŽDçŌĀ■TæĀğæŸrāŌČāçCæ■đ' āRŮāñçēŁŌçŽDēĠēçAāŌşāZāāzNāyĀāĀČ  
çŌĀ■TælēēōšriijNāyĀäylactorārşæŸrāyĀäylāzūāRŞæL'gēāNçŽDāzāŁāāiijNārĒæŸrçŌĀ■TçŽDæL'gēāNār  
āŞ■āžTēŁZāžZæūLæĀræŮūiijNāōČāRrēČ;ēŁŸāijŽçžZāĒūāzŮactorārŞēĀAæŽt'ēŁZāyĀæ■ēçŽDæūLæĀrā  
actorāžNēŮt'çŽDēĀZāfāæŸrā■TārŞāŠNāijCæ■ēçŽDāĀČāZāæ■đ' iijNæūLæĀrārŞēĀAēĀĒāy■çşēēAŞæūL  
āzşāy■āijZæŌēæTūāĒrāyĀäylæūLæĀrāūšēčnād' DçRĒçŽDāZđāžTæLŮēĀZçşēāĀČ

čzŠaŘLä;čçTlāyÄäyłçžŁçlŃaŠNāyÄäyłčYšāLŮaRřazeāŁăőzæYšçŽDăőŽázL'actoriijNăĹNăĕĆiijŽ

```
from queue import Queue
from threading import Thread, Event

# Sentinel used for shutdown
class ActorExit(Exception):
    pass

class Actor:
    def __init__(self):
        self._mailbox = Queue()

    def send(self, msg):
        '''
        Send a message to the actor
        '''
        self._mailbox.put(msg)

    def recv(self):
        '''
        Receive an incoming message
        '''
        msg = self._mailbox.get()
        if msg is ActorExit:
            raise ActorExit()
        return msg

    def close(self):
        '''
        Close the actor, thus shutting it down
        '''
        self.send(ActorExit)

    def start(self):
        '''
        Start concurrent execution
        '''
        self._terminated = Event()
        t = Thread(target=self._bootstrap)

        t.daemon = True
        t.start()

    def _bootstrap(self):
        try:
            self.run()
        except ActorExit:
            pass
        finally:
            self._terminated.set()
```

```

def join(self):
    self._terminated.wait()

def run(self):
    '''
    Run method to be implemented by the user
    '''
    while True:
        msg = self.recv()

# Sample ActorTask
class PrintActor(Actor):
    def run(self):
        while True:
            msg = self.recv()
            print('Got:', msg)

# Sample use
p = PrintActor()
p.start()
p.send('Hello')
p.send('World')
p.close()
p.join()

```

```

    def run(self):
        while True:
            msg = self.recv()
            print('Got:', msg)

# Sample use
p = PrintActor()
p.start()
p.send('Hello')
p.send('World')
p.close()
p.join()

```

The `PrintActor` class is a subclass of `Actor`. It implements the `run` method, which is a loop that receives messages from the mailbox and prints them. The `join` method is used to wait for the actor to finish its execution.

```

def print_actor():
    while True:

        try:
            msg = yield          # Get a message
            print('Got:', msg)
        except GeneratorExit:
            print('Actor terminating')

# Sample use
p = print_actor()
next(p)          # Advance to the yield (ready to receive)
p.send('Hello')

```

```
p.send('World')
p.close()
```

## èóìèőž

actoræĺaaijRçŽĐē■ĖāŁŻārsāIJlāžŌāóČžĐčōĀā■TæĀgāĀĆ  
āóđéŽĚäyŁiijNēŁŽéGŇāžĚāžĖāRlæIJL'äyĀäyłæäyāŁČæŠ■ā;IJ send() .  
çŤŽēGšiiJNāržāžŌāIJlāšžāžŌactorçšžçžšäy■çŽDāĀIJæūŁæAřāĀlçŽĐæšŽāNŪæçČāŁtāRřāžēāũsād'Žçg■æŮ  
ā;NāēČriijNā;āāRřāžēāžēāĚČčžDā;čāijRāijāéĀŠæāGç■;æūŁæAřiiJNèol'actoræL'gēāNāy■āRŇçŽĐæŠ■ā;IJiij

```
class TaggedActor(Actor):
    def run(self):
        while True:
            tag, *payload = self.recv()
            getattr(self, 'do_'+tag)(*payload)

    # Methods corresponding to different message tags
    def do_A(self, x):
        print('Running A', x)

    def do_B(self, x, y):
        print('Running B', x, y)

# Example
a = TaggedActor()
a.start()
a.send(('A', 1))      # Invokes do_A(1)
a.send(('B', 2, 3))   # Invokes do_B(2,3)
```

ā;IJāyžāRēād'ŮäyĀäyłā;Nā■RriijNāyNēlčžŽDactorāĖAēōyāIJlāyĀäyłāũčā;IJēĀĚäy■ēŁRēāNāžžæĎRçŽ  
āžūäyŤéĀŽēŁGäyĀäyłçL'žæōŁçŽĐResultāržžēšāēŁŤāŽđçžŠæđIJiijŽ

```
from threading import Event
class Result:
    def __init__(self):
        self._evt = Event()
        self._result = None

    def set_result(self, value):
        self._result = value

        self._evt.set()

    def result(self):
        self._evt.wait()
        return self._result

class Worker(Actor):
    def submit(self, func, *args, **kwargs):
```



```

        r = Result()
        self.send((func, args, kwargs, r))
        return r

    def run(self):
        while True:
            func, args, kwargs, r = self.recv()
            r.set_result(func(*args, **kwargs))

# Example use
worker = Worker()
worker.start()
r = worker.submit(pow, 2, 3)
print(r.result())

```

æIJĀŖŌiijNāĀIJāRSéĀAāĀIāyĀäyġāzzāLāæŭLæAŕçŽDæÇĀŧŧāRŕāzēēcnæL'ŕāsŧāLŕād'ŽēŧŽçlNçŧŽ  
 äĴNāēČiijNāyĀäyġçszactorārŕzēsāçŽD send() æŰzæŧŧāRŕāzēēcnçijŰçlNēōŕ'āōČēČĴāIJāyĀäyġāēŰæŌēāŰ  
 æLŰēĀŽēŧGæŧŕāzŽæŭLæAŕāyŰŰ'āzŭiijLæŕŧāçĀMQPāĀAZMQçŰL'iijLæŭēāRSéĀAāĀC

## 14.11 12.11 āōđçŌŕæŭLæAŕāŖSāyČ/ēōcéYĔæŭāđN

éŰōécŸ

äĴæIJL'äyĀäyġāŧžāžŌçžŧçlNéĀŽāŧçŽDçlNāžŖiijNæČŧēōŕ'āōČāznāōđçŌŕāŖSāyČ/ēōcéYĔæŭāđiijŖçŽ

ēğçāEŧæŰzæāĴ

ēēAāōđçŌŕāŖSāyČ/ēōcéYĔçŽDæŭLæAŕēĀŽāŧæŭāđiijŖiijN  
 äĴæĀŽāyŷēēAāiijŧāĔēäyĀäyġāŰŧçNñçŽDāĀIJāžd' æŰçæIJžāĀĀLŰāĀIJçĴSāĔŧāĀŕāŕzēsāqĴIJäyžæL'ĀæIJL'æ  
 āžŧāŕŧæYŕēŧ'iijNāyŰçŽŧ'æŌēāŕEæŭLæAŕāzŌäyĀäyġāzzāLāāŖSéĀAāLŕāŖēäyĀäyġiijNēĀNæYŕāŕEāĔŭāŖSé  
 çDŭāŖŌçŧŧāžd' æŰçæIJžāŕEāōČāŖSéĀAçžŽäyĀäyġæLŰāđ'ŽäyġēcnāĔŧēAŧāzzāLāāĀČäyNéŭæYŕāyĀäyġēŭ

```

from collections import defaultdict

class Exchange:
    def __init__(self):
        self._subscribers = set()

    def attach(self, task):
        self._subscribers.add(task)

    def detach(self, task):
        self._subscribers.remove(task)

    def send(self, msg):
        for subscriber in self._subscribers:
            subscriber.send(msg)

```

```

# Dictionary of all created exchanges
_exchanges = defaultdict(Exchange)

# Return the Exchange instance associated with a given name
def get_exchange(name):
    return _exchanges[name]

```

äyÄäyläzd' æ■caeIJzärsæYräyÄäylæZóéÄZärfzèsaiijNèt' šèt' ččzt' æŁd' äyÄäylæt' zèuČçŽDèócéYĚèÄĚéZ  
 æfRäyläzd' æ■caeIJzéÄŽèŁGäyÄäyläR■çgräóŽä;■iijNget\_exchange()  
 éÄŽèŁGçzŽäóŽäyÄäyläR■çgrèŁTäZđçŽyāžTçŽD Exchange āōđä;NāĀĆ

äyNéIcaeYräyÄäylçóĀā■Tä;Nā■RiijNæijTçd' žāžEāēĆä;Tä;ŁçTlāyÄäyläzd' æ■caeIJziijŽ

```

# Example of a task. Any object with a send() method

class Task:
    ...
    def send(self, msg):
        ...

task_a = Task()
task_b = Task()

# Example of getting an exchange
exc = get_exchange('name')

# Examples of subscribing tasks to it
exc.attach(task_a)
exc.attach(task_b)

# Example of sending messages
exc.send('msg1')
exc.send('msg2')

# Example of unsubscribing
exc.detach(task_a)
exc.detach(task_b)

```

ār;čōāārzāžŎēŁZäyléUóécYæIJL' ā;Łād' ŽčŽDāRŸçg■iijNäy■èŁGäyGāRŸäy■çzāĚūāóUāĀĆ  
 æŭLæAřaijŽēcāRŠéĀAçzŽäyÄäyläzd' æ■caeIJziijNçDŭāRŎäžd' æ■caeIJzaijŽāřEāóCāznāRŠéĀAçzŽēcñçzŠā

## èõlèõž

éÄŽèŁGéYšāLŪāRŠéĀAæŭLæAřçŽDāzzāŁæāŁŮçžŁćÍNçŽDælaaijRā;ŁāóžæYšēcāāōđçŎřāzūäyTāzš  
 äy■èŁGiiijNä;ŁçTlāRŠäyČ'èócéYĚælaaijRçŽDäē;ād'DæZt' āŁäæYŎæY;āĀĆ

éēŪāĚLiiijNä;ŁçTlāyÄäyläzd' æ■caeIJzāRřāzčçóĀāNŮād' gēĆlāŁEæŭL' āRŁāŁřçžŁćÍNéÄŽāŁaçŽDāuēä;I  
 æŪāēIJĀāŎžāEŽéÄŽèŁGād' ŽèŁŽçÍNælaaiŪælēæš■ā;IJād' ŽäylçžŁćÍNiiijNä;āāRléIJĀēēAä;ŁçTlēŁZäyläzd' æ

æſŖçğ■çİŇăžēyĹiijŇēſZăylăſēuſæŮēăſŮăſăİŮçŽĐăuēăĲĲăŖſçŖĒçſzăiijăĂĆ  
ăôđēŽĒăyĹiijŇăôĈăŖăžēēĲăĲçŽĐēğçēĂēçİŇăžŖăy■ăđ'ŽăylăžzăĹăăĂĆ

ăĒŮăŇăiijŇăžđ'æ■ăĲăžăſăſ■ăŮĹăĂſçžŽăđ'ŽăylēôcéŸĒēĂĒçŽĐēĈĲăĹăŽăyēăĲēăžĒăyĂăylăĒĲăŮſçž  
ăĲŇăēĈiijŇăĲăăŖăžēăĲçŤĲăđ'ŽăžzăĹăçſžçzſăĂăăžſăſ■ăĹŮăĹ'ĠăĠăăĂĆ  
ăĲăēſŸăŖăžēēĂŽēſĠăžēăŽôēĂŽēôcéŸĒēĂĒēžŇăžĲçžŖăôžăĲēăđĐăžžēſĈēſŤăŖŇēſĹăŮ■ăŮēăĒŮăĂĆ  
ăĲŇăēĈiijŇăyŇēĲăŸŖăyĂăylçôĂă■ŤçŽĐēſĹăŮ■çſzĲiijŇăŖăžēăŸĲçđ'žēčŇăŖŖſēĂăçŽĐăŮĹăĂſiijŽ

```
class DisplayMessages:
    def __init__(self):
        self.count = 0
    def send(self, msg):
        self.count += 1
        print('msg[{}]: {}'.format(self.count, msg))
```

```
exc = get_exchange('name')
d = DisplayMessages()
exc.attach(d)
```

æĲĲăŖŖŖiijŇēſăôđçŖŖçŽĐăyĂăylēĠēēĂçĹ'žçĈăſŸŖăôĈēĈĲăĒiijăôžăđ'ŽăylăĂĲtask-  
likeăĂĲſŖžēſăăĂĆăĲŇăēĈiijŇăŮĹăĂſăŖăŖŖŮēĂĒăŖăžēăŸſſſſiijĲ12.10ăſŖŖēĹĈăžŇçž■iijĲăĂăă■ŖçİŇ  
send()æŮžăſŤçŽĐăyĲēēſăĂĆ

ăĒſăžŖŖăžđ'æ■ăĲăžçŽĐăyĂăylăŖŖēĈĲēŮôcéŸăŸŖăſăžŖŖŖôôcéŸĒēĂĒçŽĐă■ăçăôçžŖăôžăŖŖŇēğççžŖăă  
ăyžăžĒă■ăçăôçžŽĐçôçŖĒēĲăžŖiijŇăſŖăyĂăylçzŖăôžçŽĐēôcéŸĒēĂĒăſĒēăžăĲăçžĲēăĒēğççžŖăăĂĆ  
ăĲĲăžççăĂăy■ăŽăyŷăiijŽăŸŖăĈŖăyŇēĲēſŽăăŮçŽĐăĲăăiijŖiijŽ

```
exc = get_exchange('name')
exc.attach(some_task)
try:
    ...
finally:
    exc.detach(some_task)
```

æſŖçğ■ăĎŖăžĹăyĹiijŇēſZăylăŖŖŇăĲçŤĲăŮĠăžŷăăĂăēŤĂăŖŖſçſzăiijăſſſſăăĲăĈŖăăĂĆ  
ēĂŽăyŷăĲăĲôžăŸſăiijŽăſŸēôſăĲĲăŖŖŖŖŖŽĐdetach()æ■ēēĲăăĂĆ  
ăyžăžĒçôĂăŮŮēſZăylăiijŇăĲăăŖăžēēĂĈēŽŖăĲçŤĲăyĲăyŇăŮĠçôçŖĒēăžăĲăŖēôôăĂĆ  
ăĲŇăēĈiijŇăĲĲăžđ'æ■ăĲăſſſſăăyĲăăđăĲăăyĂăylsubscribe()  
æŮžăſſſſiijŇăēĈăyŇiijŽ

```
from contextlib import contextmanager
from collections import defaultdict

class Exchange:
    def __init__(self):
        self._subscribers = set()

    def attach(self, task):
        self._subscribers.add(task)

    def detach(self, task):
```

```

        self._subscribers.remove(task)

    @contextmanager
    def subscribe(self, *tasks):
        for task in tasks:
            self.attach(task)
        try:
            yield
        finally:
            for task in tasks:
                self.detach(task)

    def send(self, msg):
        for subscriber in self._subscribers:
            subscriber.send(msg)

# Dictionary of all created exchanges
_exchanges = defaultdict(Exchange)

# Return the Exchange instance associated with a given name
def get_exchange(name):
    return _exchanges[name]

# Example of using the subscribe() method
exc = get_exchange('name')
with exc.subscribe(task_a, task_b):
    ...
    exc.send('msg1')
    exc.send('msg2')
    ...

# task_a and task_b detached here

```

æIJĀāRŌēfYāzTēreæslæDRčŽDæYřāĚšāzŌāzd' æ■cæIJžčŽDæĀīæČsæIJL'āĭLād'Žčg■čŽDæL'āsTāōd  
 āĭNāeČījNāzd' æ■cæIJžāRřāzēāōdčŌřāyĀæT'āyĭæūLæAřēĀŽéAšéZEāRĹLĹŪæRŘāĭZāzd' æ■cæIJžāR■čg  
 āzd' æ■cæIJžēfYāRřāzēēcnæL'āsTāLřāLEāyČāijRēōačōŪčĹNāzRāy■ījLærTāeČījNāřEæūLæAřēūřčTśāLřā

## 14.12 12.12 ä;ĚčTĭčTšæLŘāZĭāzčæŽĚčžĚčĹN

### éUőécŸ

ä;āæČšā;ĚčTĭčTšæLŘāZĭījLā■RčĹNījLæŽĚāzččšzčzščžĚčĹNæĭēāōdčŌřāzūāRŚāĀČēĚZāyĭæIJL'æŪūāĹ

### èğčāĚsæŪzæāĹ

ēēAä;ĚčTĭčTšæLŘāZĭāōdčŌřēGĭāūsčŽDāzūāRŚījNā;āēēŪāĚLēēAārččTšæLŘāZĭāG;æTřāŠN  
 yield ēř■āRēæIJL'æūsāLzčRĚèğčāĀČ yield ēř■āRēāijŽēōĹ'āyĀāyĭčTšæLŘāZĭāNČēĭūāōČčŽDæL'ğēāNřī

ärEçTŧšæLŖăZlă;ŞăAŽæšŖçğ■ăĂIJăzzăLăăĂlăzŭă;ŧçTlăzzăLăă■Ră;IJăLŖă■cæİæŽŧæ■căŏCăzŋçŽDæL'ğ  
èçAæijTçd'žèŧŽçğ■æĂlăCŧijŇèĂCèŽSăyŇélcăyd'âyŧă;ŧçTlçŏĂă■TçŽD yield  
èr■ăŖēcŽDçTŧšæLŖăZlăĜ;æTŧijŽ

```
# Two simple generator functions
def countdown(n):
    while n > 0:
        print('T-minus', n)
        yield
        n -= 1
    print('Blastoff!')

def countup(n):
    x = 0
    while x < n:
        print('Counting up', x)
        yield
        x += 1
```

èŧŽăžŽăĜ;æTŧăIJăLŖăĚéCłă;ŧçTlŧyieldèr■ăŖēcijŇăyŇélcæYŕăyĂăyŧăŏđçŐŕăžEçŏĂă■TăzzăLăæŧCăžęăŽl

```
from collections import deque

class TaskScheduler:
    def __init__(self):
        self._task_queue = deque()

    def new_task(self, task):
        '''
        Admit a newly started task to the scheduler
        '''
        self._task_queue.append(task)

    def run(self):
        '''
        Run until there are no more tasks
        '''
        while self._task_queue:
            task = self._task_queue.popleft()
            try:
                # Run until the next yield statement
                next(task)
                self._task_queue.append(task)
            except StopIteration:
                # Generator is no longer executing
                pass

# Example use
sched = TaskScheduler()
sched.new_task(countdown(10))
```

```

sched.new_task(countdown(5))
sched.new_task(countup(15))
sched.run()

```

TaskScheduler çşzâIJläyÄäylâ;İçÖräy■èĒRèaŇçTşæLRâZİléZEâRLâATâATæfRäyİéÇ;èĒRèaŇâLİçç  
èĒRèaŇèĒZäylâ;Ňâ■RiijNè;ŞâGzâæCâyŇiijZ

```

T-minus 10
T-minus 5
Counting up 0
T-minus 9
T-minus 4
Counting up 1
T-minus 8
T-minus 3
Counting up 2
T-minus 7
T-minus 2
...

```

âLræ■d'äyæ■ciijŇæLSäznâôdéZĒäyLâüşçzRâôđçÖräZEäyÄäylâÄIJæŞ■ä;IJçşzçzşşâÄİçZDæIJÄârRæäy  
çTşæLRâZİlâG;æTřärsæYřèôd'äyziijŇèÄŇyieldeİ■âRèæYřäzzâLææŇCètûçZDäĒaâRûâĂĆ  
èĒCâæZİlâ;İçÖräçÄæŞèäzzâLââLÜèaİçZt'âLræşæaIJL'äzzâLæèAæLğèaNäyæ■câĂĆ

âôdéZĒäyLiiŇNä;ââRrèÇ;æČşèAä;İçTİçTşæLRâZİlæİèâôđçÖřçôĀâ■TçZDäzûâRSâĂĆ  
éĆçäZİijŇâIJlâôđçÖřactoræLŪç;ŞçzIJæIJ■âLâZİçZDæŪûâĂZä;ââRräzèä;İçTİçTşæLRâZİlæİèæZĒäzççZĒç

äyŇéİççZDäzççAæijTçd'zäZĒä;İçTİçTşæLRâZİlæİèâôđçÖräyÄäylây■ä;İèTŪçzĒçİŇçZDactoriijZ

```

from collections import deque

class ActorScheduler:
    def __init__(self):
        self._actors = { }           # Mapping of names to actors
        self._msg_queue = deque()    # Message queue

    def new_actor(self, name, actor):
        '''
        Admit a newly started actor to the scheduler and give it a_
↪name
        '''
        self._msg_queue.append((actor, None))
        self._actors[name] = actor

    def send(self, name, msg):
        '''
        Send a message to a named actor
        '''
        actor = self._actors.get(name)
        if actor:
            self._msg_queue.append((actor, msg))

```



```

def handle_yield(self, sched, task):
    pass
def handle_resume(self, sched, task):
    pass

# Task Scheduler
class Scheduler:
    def __init__(self):
        self._numtasks = 0      # Total num of tasks
        self._ready = deque()   # Tasks ready to run
        self._read_waiting = {} # Tasks waiting to read
        self._write_waiting = {} # Tasks waiting to write

    # Poll for I/O events and restart waiting tasks
    def _iopoll(self):
        rset, wset, eset = select(self._read_waiting,
                                   self._write_waiting, [])

        for r in rset:
            evt, task = self._read_waiting.pop(r)
            evt.handle_resume(self, task)
        for w in wset:
            evt, task = self._write_waiting.pop(w)
            evt.handle_resume(self, task)

    def new(self, task):
        '''
        Add a newly started task to the scheduler
        '''

        self._ready.append((task, None))
        self._numtasks += 1

    def add_ready(self, task, msg=None):
        '''
        Append an already started task to the ready queue.
        msg is what to send into the task when it resumes.
        '''

        self._ready.append((task, msg))

    # Add a task to the reading set
    def _read_wait(self, fileno, evt, task):
        self._read_waiting[fileno] = (evt, task)

    # Add a task to the write set
    def _write_wait(self, fileno, evt, task):
        self._write_waiting[fileno] = (evt, task)

    def run(self):
        '''
        Run the task scheduler until there are no tasks

```



```

'''
while self._numtasks:
    if not self._ready:
        self._iopoll()
    task, msg = self._ready.popleft()
    try:
        # Run the coroutine to the next yield
        r = task.send(msg)
        if isinstance(r, YieldEvent):
            r.handle_yield(self, task)
        else:
            raise RuntimeError('unrecognized yield event')
    except StopIteration:
        self._numtasks -= 1

# Example implementation of coroutine-based socket I/O
class ReadSocket(YieldEvent):
    def __init__(self, sock, nbytes):
        self.sock = sock
        self.nbytes = nbytes
    def handle_yield(self, sched, task):
        sched._read_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        data = self.sock.recv(self.nbytes)
        sched.add_ready(task, data)

class WriteSocket(YieldEvent):
    def __init__(self, sock, data):
        self.sock = sock
        self.data = data
    def handle_yield(self, sched, task):
        sched._write_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        nsent = self.sock.send(self.data)
        sched.add_ready(task, nsent)

class AcceptSocket(YieldEvent):
    def __init__(self, sock):
        self.sock = sock
    def handle_yield(self, sched, task):
        sched._read_wait(self.sock.fileno(), self, task)
    def handle_resume(self, sched, task):
        r = self.sock.accept()
        sched.add_ready(task, r)

# Wrapper around a socket object for use with yield
class Socket(object):
    def __init__(self, sock):
        self._sock = sock

```

```

def recv(self, maxbytes):
    return ReadSocket(self._sock, maxbytes)
def send(self, data):
    return WriteSocket(self._sock, data)
def accept(self):
    return AcceptSocket(self._sock)
def __getattr__(self, name):
    return getattr(self._sock, name)

if __name__ == '__main__':
    from socket import socket, AF_INET, SOCK_STREAM
    import time

    # Example of a function involving generators. This should
    # be called using line = yield from readline(sock)
    def readline(sock):
        chars = []
        while True:
            c = yield sock.recv(1)
            if not c:
                break
            chars.append(c)
            if c == b'\n':
                break
        return b''.join(chars)

    # Echo server using generators
    class EchoServer:
        def __init__(self, addr, sched):
            self.sched = sched
            sched.new(self.server_loop(addr))

        def server_loop(self, addr):
            s = Socket(socket(AF_INET, SOCK_STREAM))

            s.bind(addr)
            s.listen(5)
            while True:
                c, a = yield s.accept()
                print('Got connection from ', a)
                self.sched.new(self.client_handler(Socket(c)))

        def client_handler(self, client):
            while True:
                line = yield from readline(client)
                if not line:
                    break
                line = b'GOT:' + line
                while line:
                    nsent = yield client.send(line)

```

```

        line = line[nsent:]
    client.close()
    print('Client closed')

sched = Scheduler()
EchoServer(('', 16000), sched)
sched.run()

```

èŁŻæŁăžçčăAæIJL'ćĆăđ'■æĬăĂĈăy■èŁĜiijŃăőĈăđçŎřăžĚăyĀăyĽăŕăđŃčŽĐăŞ■ă;IJçşžçžşăĂĈ  
 æIJL'ăyĀăyĽăŕşçžłçŽĐăžžăĽăéŸşăĽŮiijŃăžŮăyŤéŸæIJL'ăŽăĽ/OăiijŚçIJăçŽĐăžžăĽăç■Ľă;ĚăŃžăşşăĂĈ  
 èŁŸæIJL'ă;Ľăđ'ŽërĈăžęăŽĽèt'şet'căIJĽăŕşçžłçŸşăĽŮăŞŃĽ/Oç■Ľă;ĚăŃžăşşăžŃéŮŤ'çğžăĽăžžăĽăăĂĈ

## èőłèőž

ăIJăđĐăžžăşşăžŎçŤşæĽŔăŽĽçŽĐăžŮăŔŚăæĚăđŮăŮŮiijŃéĂŽăyŷăiijŽă;ŁçŤĽăŽŤ'ăyŷèğAçŽĐyielďă;ćă

```

def some_generator():
    ...
    result = yield data
    ...

```

ä;ŁçŤĽèŁŻçğ■ă;ćăiijŔçŽĐyielďër■ăŔëçŽĐăĜ;æŤŕëĂŽăyŷèćŋçğŕăyžăăIJă■ŔçĽŃăĂĽăĂĈ  
 éĂŽèŁĜërĈăžęăŽĽiijŃyielďër■ăŔëăIJăyĀăyĽă;łçŎřăy■èćŋăđ'ĐçŔĚiijŃăęĈăyŃŮiijŽ

```

f = some_generator()

# Initial result. Is None to start since nothing has been computed
result = None
while True:
    try:
        data = f.send(result)
        result = ... do some calculation ...
    except StopIteration:
        break

```

èŁŻéĜŃçŽĐéĂžè;ŚçĽ■ă;őæIJL'ćĆăđ'■æĬăĂĈăy■èŁĜiijŃèćŋăiijăçžŽ  
 send() çŽĐăĂijăőŽăžĽăžĚăIJĽyielďër■ăŔëéĚşæĽăŮŮçŽĐèŁŤăŽđăĂijăĂĈ  
 äŽăæ■đ'ŮiijŃăęĈăđIJăyĀăyĽyielďăĜĚăđ'ĜăIJĽăŕžăžŃăĽ■yielď-  
 æŤŕæ■őçŽĐăŽđăžŤăy■èŁŤăŽđçžşæđIJăŮŮiijŃăiijŽăIJăyŃăyĀăŋă send()  
 æŞ■ă;IJèŁŤăŽđăĂĈăęĈăđIJăyĀăyĽçŤşæĽŔăŽĽăĜ;æŤŕăĽŽăiijĂăğŃèŁŔëăŃŮiijŃăŔŚéĂăăyĀăyĽŃoneăĂijăiij

éŽđ'ăžĚăŔŚéĂăăĂijăđ'ŮŮiijŃèŁŸăŔŕăžęăIJăyĀăyĽçŤşæĽŔăŽĽăyĽéĽăĽ'ğëăŃăyĀăyĽ  
 close() æŮžăęŤăĂĈăőĈăiijŽăŕijèĜŤ'ăIJăĽ'ğëăŃyielďër■ăŔëăŮŮăĽăŽăĜžăyĀăyĽ  
 GeneratorExităiijĈăyŷŮiijŃăžŎèĂŃçžĽă■ćăĽ'ğëăŃăĂĈ  
 ăęĈăđIJăĚŸăyĀă■èèő;èőăiijŃăyĀăyĽçŤşæĽŔăŽĽăŔŕăžęă■ŤèŎŮèŁŸăyĽăiijĈăyŷăžŮăĽ'ğëăŃăyĚçŔĚăŞ■ă;IJ  
 ăŔŃăăŮèŁŸăŔŕăžęă;ŁçŤĽçŤşæĽŔăŽĽçŽĐ throw() æŮžăęŤăIJyielď-  
 èŮăŔëăĽ'ğëăŃăŮŮçŤşæĽŔăyĀăyĽăžžăđŔçŽĐăĽ'ğëăŃăŃĜăžđ'ăĂĈ  
 ăyĀăyĽăžžăĽăăĚërĈăžęăŽĽăŔŕăĽŤ'çŤăőĈăĽëăIJĽèŁŔëăŃçŽĐçŤşæĽŔăŽĽăy■ăđ'ĐçŔĚéŤŽërŕăĂĈ

æIJĀāRŌäyÄäyĭä;Nā■Räy■ä;fçTĭçŽD yield from èr■āRēècñçTĭlæĭëāōđçŌrā■RçĭNĭijNāRfrazèècñāĖ  
æIJnètĭäyĽārsæYřāEæŌgāLūæĭČēARæYŌçŽDäijäe;ŞçzZæŪřçŽDāG;æTřāĀĆ  
äy■āČRæŽōēĀŽçŽDçTşæĽRāZĭijNäyÄäyĭä;fçTĭ yield from  
ècñèrČçTĭçŽDāG;æTřāRfrazèèfTāZđäyÄäyĭä;IJäyž yield from  
èr■āRēçzŞæđIJçŽDāĀijāĀĆ āĖşāžŌ yield from çŽDæŽt'ād'ŽāfæAřāRfrazèāIJĭ PEP  
380 äy■æL;āĽRāĀĆ

æIJĀāRŌĭijNāeČæđIJä;fçTĭçTşæĽRāZĭçijŪçĭNĭijNēeAæRŘēEŞä;ăçŽDæYřāōČèfYæYřæIJĽ'ā;Ľād'Žç  
çĽ'zāĽnæYřĭijNä;āä;Ūäy■āĽrazzä;TçžfçĭNāRfrazèæRŘä;ŽçŽDæ;ād'DāĀĆä;NāeČĭijNāeČæđIJä;āæĽ'gèaŃ  
āōČäijZārEæTř'äyĽāzzāĽæŃCètũçşēēAŞæŞ■ä;IJāōNæĽRāĀĆäyžāzEègçāEşèfZäyĭēŪōēcYřĭijN  
ä;āāRĭèČ;éĀĽ'æNř'ārEæŞ■ä;IJāgTæt'çzZāRēād'ŪäyÄäyĭāRfrazèçNñçñNēfRēaŃçŽDçžfçĭNæĽŪèfZçĭNāĀ  
āRēād'ŪäyÄäyĭēZŘāĽūæYřād'gēČĭāĽEPythonāžŞāzūäy■ēČ;ā;Ľāē;çŽDāEijāōzāşzāžŌçTşæĽRāZĭçŽDçžfçĭ  
āeČæđIJä;āēĀĽ'æNř'èfZäyĭæŪzæaĽĭijNä;āäijZāRŞçŌrā;æIJĀēeAēĜĭāūsæTzāEŽā;Ľād'ŽæāĜāĜEāžŞāG;æ  
ä;IJäyžæIJnēĽCæRŘāĽřçŽDā■RçĭNāSŃçŽyāĖŞæĽĀæIJřçŽDäyÄäyĭāşžçāĖeČNæŽřĭijNāRfrazèæşēçIJN  
PEP 342 āSŃ āĀIJā■RçĭNāSŃNāžūāRŞçŽDäyĀēŪĭæIJĽ'èüçèr;çĭNāĀĭ

PEP 3156 āRŃæāūæIJĽ'äyÄäyĭāĖŞāžŌä;fçTĭā■RçĭNçŽDäijČæ■ēI/OæĭāādNāĀĆ  
çĽ'zāĽnçŽDĭijNä;āäy■āRřèČ;èĜĭāūsāŌzāōđçŌrāyÄäyĭāžTāsČçŽDā■RçĭNèrČāžæāZĭāĀĆ  
äy■ēfĜĭijNāĖŞāžŌā■RçĭNçŽDæĀĭæČşæYřā;Ľād'ŽætAèaŃāžŞçŽDāşžçāĀĭijN āNĖæNñ  
gevent, greenlet, Stackless Python āžēāRĽāĖŪāžŪçşzäijjāūēçĭNāĀĆ

## 14.13 12.13 ād'ŽäyĭçžfçĭNéYşāĽŪè;ōèrc

### éŪōēcY

ä;āæIJĽ'äyÄäyĭçžfçĭNéYşāĽŪèZEāRĽĭijNæČşäyžāĽræĭèçŽDāĖČçt'æ;ōèrcāōČāznĭijN  
ārşèuşä;äyžäyÄäyĭāōcæĽuçñrèrūæśČāŌzè;ōèrcäyÄäyĭç;ŞçzIJēđæŌēēZEāRĽçŽDæŪzāijRäyĀæāūāĀĆ

### ègçāEşæŪzæaĽ

ārzāžŌè;ōèrcéŪōēcYçŽDäyÄäyĭäyègAègçāEşæŪzæaĽäy■æIJĽ'äyĭä;ĽārŞæIJĽ'āžžçşēēAŞçŽDæĽĀāū  
æIJnètĭäyĽèōşāĖŪæĀĭæČşārşæYřĭijZāržāžŌæRĀyĭä;āæČşèeAè;ōèrcçŽDēYşāĽŪĭijNä;āāĽzāžäyĀāržèfđā  
çĐūāRŌā;āāIJĭāĖŪäy■äyÄäyĭāēŪæŌēā■ŪäyĽēĭçijŪāEŽāžççāAæĭæāĜèrEā■YāIJĭçŽDæTřæ■ōĭijN  
āRēād'ŪäyÄäyĭāēŪæŌēā■ŪēcñäijäçzŽ select () æĽŪçşzäijijçŽDäyÄäyĭē;ōèrcæTřæ■ōāĽrē;ççŽDāG;æTř

```
import queue
import socket
import os

class PollableQueue(queue.Queue):
    def __init__(self):
        super().__init__()
        # Create a pair of connected sockets
        if os.name == 'posix':
            self._putsocket, self._getsocket = socket.socketpair()
        else:
            # Compatibility on non-POSIX systems
            server = socket.socket(socket.AF_INET, socket.SOCK_
↳STREAM)
```

```

server.bind(('127.0.0.1', 0))
server.listen(1)
self._putsocket = socket.socket(socket.AF_INET, socket.
→SOCK_STREAM)
self._putsocket.connect(server.getsockname())
self._getsocket, _ = server.accept()
server.close()

def fileno(self):
    return self._getsocket.fileno()

def put(self, item):
    super().put(item)
    self._putsocket.send(b'x')

def get(self):
    self._getsocket.recv(1)
    return super().get()

```

aIJlëfZäyläzččāAäy■rijNäyÄäylæŮřčŽD Queue āōdāĹNčšzādNēcñāōZāzL'rijNāzTāsCæYřayÄäylēcñēf  
 āIJlUnixæIJzāZlāyŁčŽD socketpair() āĜ;æTřèČ;è;zæĹčŽDāLZāzzèfZæuĉŽDāēŮæŌēā■ŮāĀĆ  
 āIJlWindowsäyLélcijNā;āāfĒéazä;ĤčTlčszāijijāzččāAælēāĹæNšāōČāĀĆ  
 čDūāRŌāōZāzL'æZŏéĀŽčŽD get() āŠN put() æŮzæšTāIJlëfZāzZāēŮæŌēā■ŮäyLélcælēæL'gèaNI/OæS  
 put() æŮzæšTāE■ārEæTřæ■ōæTĹāĒēēYšāLŮāRŌāijZāEZäyÄäylā■Tā■ŮèLCāLřæšRāylāēŮæŌēā■Ůäy■ā  
 èĀN get() æŮzæšTāIJlāzŌēYšāLŮäy■čgžéZd'äyÄäylāĒČčt'āæŮūāijZāzŌāRēād'ŮäyÄäylāēŮæŌēā■Ůäy■ā

fileno() æŮzæšTā;ĤčTlāyÄäylāĜ;æTřærTāēĆ select()  
 ælēēōl'ēfZäylēYšāLŮāRřazēēcñē;ŏērčāĀĆ āōČāzĒāzĒāRlæYřæZt'ēIJšāzEāzTāsCècñ  
 get() āĜ;æTřā;ĤčTlāLřčŽDsocketčŽDæŮĜāzūæRŘēfřčņēèĀNāūsāĀĆ

äyNélcæYřayÄäylāĹNā■RrijNāōZāzL'āzEäyÄäylāyžāLřælēčŽDāĒČčt'āčZSæŌgād'ZäylēYšāLŮčŽDæū

```

import select
import threading

def consumer(queues):
    '''
    Consumer that reads data on multiple queues simultaneously
    '''
    while True:
        can_read, _, _ = select.select(queues, [], [])
        for r in can_read:
            item = r.get()
            print('Got:', item)

q1 = PollableQueue()
q2 = PollableQueue()
q3 = PollableQueue()
t = threading.Thread(target=consumer, args=(q1, q2, q3,))
t.daemon = True
t.start()

```

```
# Feed data to the queues
q1.put(1)
q2.put(10)
q3.put('hello')
q2.put(15)
...
```

ǎċĆæđIJă;ǎērTçİĂèŁRëąŃáoČñjŇä;ǎäijŽāRŚčŔřēfZăȳłæúLèt zèĂĖäijŽæŒěāRŮălŁræL'ĂæIJL'čŽĐēcń

èóìèőž

árzážŎè;ðèrcéÍdčšzæŮĜázũáržèšajijŇærŤæĆeŸšáLŮéAŽäyýeČ;æŸræŕŤèĹČæčŸæLŇčŽDěŮóécŸăĂ  
 äĹŇæĈriĴŇæĈCăđIĴă;ăäy■ă;ĤčŤlăyĹeĹčŽDăeŮăŎă■ŮăĹĂăIĴriĴŇ  
 äĴăăŤŕăyĂčŽDěĂĹ'æŇŤ'ăršæŸřciĴŮăĚŽăžččĂăĹeăĹčŎŕéA■ăŎĖèĤŽăžŸšáĹŮăžũă;ĤčŤlăyĂăyĹăôŽăŮă

```
import time
def consumer(queues):
    while True:
        for q in queues:
            if not q.empty():
                item = q.get()
                print('Got:', item)

        # Sleep briefly to avoid 100% CPU
        time.sleep(0.01)
```

ɛfZæʌuʌAŽăEũăoɔäy■āRĹčRĖrijNēfYāijŽāijTăĖĕăEũăzŮčŽDăĂgĕČjĕŮĕĕYăĂČ  
 äĲNăĕCrijNăĕCădIJăŮčŽDăTŕă■ōĕcŋăĹăăĖĕăĹŕăyĂăyĭĖYšăĹŮăy■ijNĕGšăŕSĕĕĂĕĹS10æŋĕgšăĹ■ĕČjĕ  
 äĕCădIJăjăăzNăĹ■čŽDĕjōĕŕĕĕfYĕĕĂăŌzĕjōĕŕĕăEũăzŮăŕzĕšăqijNăŕTăĕČjŚčzIJăĕŮăŌă■ŮĕCĕĕfYăijŽăĹ  
 äĲNăĕCrijNăĕCădIJăjăăCšăRŊăŮũĕjōĕŕĕăĕŮăŌă■ŮăŠNĕYšăĹŮijNăjăăRfĕČjĕĕĂăČŔăyNĕĬĕĕfZăăuăj

```
import select

def event_loop(sockets, queues):
    while True:
        # polling with a timeout
        can_read, _, _ = select.select(sockets, [], [], 0.01)
        for r in can_read:
            handle_read(r)
        for q in queues:
            if not q.empty():
                item = q.get()
                print('Got:', item)
```

ɛʃZäyʎæŮzæqʎÉĀŽɛʃĠärĖéYšáʎŮáŠŇăĕŮæŌĕā■Ůç■L'ăRŇărzăŮĖĖæiĕĕgĕcăĖsăzĖĖăd'gĕĆiăĹĖçŽĐĖŮŭ  
 äyĂäyʎă■TçNŋçŽĐ                      select ()                      ěrČčTĹăRřĕcŇăRŇăŮŮŮçTĹăiĕĕ;ŭĕrĕăĂĈ  
 äjĕçTĹiĕŮĖăŮŮăĹŮăĖŮăzŮăšzăŮăŮŮĕŮ'çŽĐăIŮzăĹŮăiĕăL'gĕăŇăŇăŚiăIŮšăĂgăĕĕĂăšĕăzŮăšăăIŮ'ăĖĖĕĕ

çTŽeGšijNæCædIæTṛæ■ōēcñāLāāĖēāLṛāyĀäyĭeYšāLŪiijNæŭlèt'zèĀĖāGāāzŌāRṛāzēāōđæŪūçŽDēcñéĀ  
ār;çōāijŽæIJL'äyĀçCžçCžāžTṛāsCçŽDI/Oæ■šèĀŪiijNä;£çTlāōCéĀŽāyāijŽeŌūā; ŪæŽt'āē;çŽDā\$■āžTæŪ

## 14.14 12.14 āJÍUnixçşçzçşäyŁéÍcāRṛāŁlāōŁæŁd'è£ŽçÍNè£

éŪōécŸ

ä;āæČşçijŪāĖŽäyĀäyĭä;IJäyžäyĀäyĭāIJÍUnixæŁŪçşçzçşäyŁéÍcè£RēāNçŽDāōŁæŁd'è£ŽçÍNè£

èğcāEşşæŪzæąŁ

āŁŽāžžäyĀäyĭæ■čçāōçŽDāōŁæŁd'è£ŽçÍNèIJĀēçĀäyĀäyĭçş;çāōçŽDçşçzçşēçČçTlāžRāŁŪāžēāRŁāržāž  
äyNéÍcçŽDāžççāĀāsTçd'žāžĖæĀŌæūāōŽāžL'äyĀäyĭāōŁæŁd'è£ŽçÍNiiijNāRṛāzēāRṛāŁlāRŌā;ŁāōžæYŞçŽD

```
#!/usr/bin/env python3
# daemon.py

import os
import sys

import atexit
import signal

def daemonize(pidfile, *, stdin='/dev/null',
               stdout='/dev/null',
               stderr='/dev/null'):

    if os.path.exists(pidfile):
        raise RuntimeError('Already running')

    # First fork (detaches from parent)
    try:
        if os.fork() > 0:
            raise SystemExit(0)    # Parent exit
    except OSError as e:
        raise RuntimeError('fork #1 failed.')

    os.chdir('/')
    os.umask(0)
    os.setsid()
    # Second fork (relinquish session leadership)
    try:
        if os.fork() > 0:
            raise SystemExit(0)
    except OSError as e:
        raise RuntimeError('fork #2 failed.')

    # Flush I/O buffers
```

```

sys.stdout.flush()
sys.stderr.flush()

# Replace file descriptors for stdin, stdout, and stderr
with open(stdin, 'rb', 0) as f:
    os.dup2(f.fileno(), sys.stdin.fileno())
with open(stdout, 'ab', 0) as f:
    os.dup2(f.fileno(), sys.stdout.fileno())
with open(stderr, 'ab', 0) as f:
    os.dup2(f.fileno(), sys.stderr.fileno())

# Write the PID file
with open(pidfile, 'w') as f:
    print(os.getpid(), file=f)

# Arrange to have the PID file removed on exit/signal
atexit.register(lambda: os.remove(pidfile))

# Signal handler for termination (required)
def sigterm_handler(signo, frame):
    raise SystemExit(1)

signal.signal(signal.SIGTERM, sigterm_handler)

def main():
    import time
    sys.stdout.write('Daemon started with pid {} \n'.format(os.
↳ getpid()))
    while True:
        sys.stdout.write('Daemon Alive! {} \n'.format(time.ctime()))
        time.sleep(10)

if __name__ == '__main__':
    PIDFILE = '/tmp/daemon.pid'

    if len(sys.argv) != 2:
        print('Usage: {} [start|stop]'.format(sys.argv[0]),
↳ file=sys.stderr)
        raise SystemExit(1)

    if sys.argv[1] == 'start':
        try:
            daemonize(PIDFILE,
                        stdout='/tmp/daemon.log',
                        stderr='/tmp/dameon.log')
        except RuntimeError as e:
            print(e, file=sys.stderr)
            raise SystemExit(1)

    main()

```



```

elif sys.argv[1] == 'stop':
    if os.path.exists(PIDFILE):
        with open(PIDFILE) as f:
            os.kill(int(f.read()), signal.SIGTERM)
    else:
        print('Not running', file=sys.stderr)
        raise SystemExit(1)

else:
    print('Unknown command {!r}'.format(sys.argv[1]), file=sys.
↪stderr)
    raise SystemExit(1)

```

èeAǎRǎLlèfZǎylǎoLǎLd'èfZǎlNijNçTlǎLúeIJǎèeAǎ;fçTlǎeCǎyNçZDǎS;ǎzd'iijZ

```

bash % daemon.py start
bash % cat /tmp/daemon.pid
2882
bash % tail -f /tmp/daemon.log
Daemon started with pid 2882
Daemon Alive! Fri Oct 12 13:45:37 2012
Daemon Alive! Fri Oct 12 13:45:47 2012
...

```

ǎoLǎLd'èfZǎlNǎRǎzèǎoNǎElǎIJlǎRǎOǎRǎfèfRǎeǎNijNǎZǎe■d'èfZǎylǎS;ǎzd'ǎijZçnNǎ■şèfTǎZdǎǎC  
ǎy■èfGrijNǎ;ǎǎRǎzèǎCǎyLéIcéCǎǎüǎşçIJNǎyǎOǎoCçZyǎEşçZDpidǎŬGǎzǎǎSǎŬèǎfŬǎǎCèeAǎAIJǎ

```

bash % daemon.py stop
bash %

```

## èõléõž

ǎIJnèLĆǎoZǎZLǎzEǎyǎǎylǎG;ǎTǎr daemonize() iijNǎIJlǎlNǎzRǎRǎLǎLǎŬűècnerCçTlǎ;fǎ;ŬçlNǎzR  
daemonize() ǎG;ǎTǎRǎLǎŬèǎRŬǎEşèTǎǎ■ŬǎRĆǎTǎrijNèfZǎǎüçZDèrlǎRǎéǎLǎǎRĆǎTǎRǎIJlècǎ;fçTlǎŬ  
ǎoCǎijZǎijzǎLűçTlǎLǎǎCǎyNéIcéfZǎǎüǎ;fçTlǎoCǎijZ

```

daemonize('daemon.pid',
          stdin='/dev/null',
          stdout='/tmp/daemon.log',
          stderr='/tmp/daemon.log')

```

èǎNǎy■ǎYǎǎCǎyNéIcéfZǎǎüǎRǎnçşLǎy■ǎyEçZDèrCçTlǎijZ

```

# Illegal. Must use keyword arguments
daemonize('daemon.pid',
          '/dev/null', '/tmp/daemon.log', '/tmp/daemon.log')

```

ǎLZǎzzǎyǎǎylǎoLǎLd'èfZǎlNçZDǎ■èeldçIJNǎyLǎŬzǎy■ǎYǎǎ;LǎYşǎeGǎCǎijNǎ;EǎYǎǎd'ǎǎ;şǎǎIǎC

ééŮăĚĹiijNăyĂăyĹăôĹăŁd'êĚŹċĹNăĤĚăzđēAăzŎċĹŮêĚŹċĹNăy■ēĎŝċzăĂĈ ēĚăŸřċŤŝ  
os.fork() æŞ■ăĹJăĹăôĎăĹŔċŹĎiijNăzŭċŋNă■ŝċċŋċĹŮêĚŹċĹNċzĹă■ċăĂĈ

ăĹJăĹă■ŔēĚŹċĹNăŔŸăĹŔă■d'ăĎĤăŔŎiijNērĈċŤĪ os.setsid()  
ăĹŹăzŹăzĚăyĹăĂăyĹăĹăĹăŮŕċŹĎēĚŹċĹNăijŹērĹiijNăzŭēōċ;ôă■ŔēĚŹċĹNăyŹēēŮēċĚăĂĈ  
ăôĈăijŹēōċ;ôēĚŹăyĹă■ŔēĚŹċĹNăyŹăŮŕċŹĎēĚŹċĹNċzĎċŹĎēēŮēċĚiijNăzŭċăôăĤăy■ăijŹăĚă■ăĹJăĹăŎġăĹŮċ  
ăēĈăĎĹJēĚŹăzŹăŔăŋăyĹăĹăŎăđ'Ĥē■ŤăzŹiijNăŹăăyŹăôĈĚĹJăĹăēĤăŔăĤăôĹăŁd'êĚŹċĹNăŔŋċzĹċŋŕăĹăĤċzăijĂăzŭ  
ērĈċŤĪ os.chdir() âŖŊ os.umask(0) æŤŹăŔŸăzĚă;ŞăĹă■ăŭēă;ĹJċŹôă;ŤăzŭēĠċ;ôăŮĠăzŭăĹĈĚŹŔăĹă  
ăĤôăŤŹċŹôă;ŤăĂŹăyŹăŸŕăyĹăē;ăyŹăĎŔiijNăŹăăyŹăēĚŹăăŭăŔŕăzăă;ĤăĹăŮăôĈăy■ăĤă■ăŭēă;ĹJăĹJăĹăċŋăŔŕăĹăĹă

ăŔēăđ'ŮăyĂăyĹērĈċŤĪ os.fork() ăĹJăĹăŹēĠŊăŹŕăĹăċċċġŸċĈăăĂĈ  
ēĚŹăyĂă■ă;ĤăĹăŮăôĹăŁd'êĚŹċĹNăđ'ŝăŎăzăŹĚēŎăăŔŮăŮŕċŹĎăŎġăĹŮċzĹċŋŕċŹĎĈ;ăĹŹăzŭăyŤēôĹăôĈă  
iijĹăĹJŋērăĹăyĹiijNērēĎăemonăŤĹăijĈăzĚăôĈċŹĎăijŹērĹēēŮēċĚăĹăŎă;■iijNăŹăă■ăđ'ăĤăăzŝăŝăăĹJăĹăĹĈĚŹŔăĹă  
ăŕĹċôăă;ăăŔŕăzăăĤĹċŤēēĚŹăyĂă■iijNăĹăĤăŸŕăĹJăăē;ăy■ēĤăĚăŹăĹăĂŹăĂĈ

ăyĂăŮăôĹăŁd'êĚŹċĹNēċŋă■ċăôċŹĎăĹăĤċzăiijNăôĈăijŹēĠă■ŮŕăĹăġŋăŋŮăăĠăĠĚĹ/OăĤăĤăŋĠăăŋĠăă  
ēĚŹăyĂăĈăĹăĤăĹăĹJĹċĈŹēŹĹăĠĈăĂĈċŭŝăăĠăĠĚĹ/OăĤăĤăŹăăŤăŤăŝċŹĎăŮĠăzŭăŕŕăzăŝċŹĎăijŤċŤĹăĹJăġċĈĠăă  
iijĹsys.stdout, sys.\_\_stdout\_\_ċ■ĹiijĹăĂĈ ăzĚăzĚċôĂăŤċŹĎăŤŝċŮ■  
sys.stdout ăzŭēĠă■ŮŕăŋĠăôŹăôĈăŸŕăăNăy■ăĂŹċŹĎiijNă  
ăŹăăyŹăŝăăĹăđăŝŤċŝēăĂŝăôĈăŸŕăŔăăĤăĹēĈĹĈ;ăŸŕċŤĹċŹĎăŸŕ sys.stdout ăĂĈ  
ēĚŹēĠŊiijNăĹăŤăzŋăĹăŤăijĂăzĚăyĂăyĹă■ŤċŋċŹĎăŮĠăzŭăŕŕăzăăiijNăzŭērĈċŤĪ os.  
dup2() iijNăċŤĹăôĈăĹăăzċăŹĤēċŋ sys.stdout ăĤĹċŤĹċŹĎăŮĠăzŭăŔŕăĤŕċŋăĂĈ  
ēĚŹăăŭiijNăsys.stdout ăĤĹċŤĹċŹĎăŎŝăġŋăŮăŮĠăzŭăiijŹēċŋăĤŝċŮăzŭċŤŝăŮŕċŹĎăĹăēĚăă■ăĂĈ  
ēĚŸēĤăĤijŹērĈċŹĎăŸŕăzăă;ŤċŤĹăzŎăŮĠăzŭċijŮċăĂăĹŮăŮĠăĹJăăđ'ĎċŔĚċŹĎăăĠăĠĚĹ/OăĤăĤăŸăĤăŸăĤăŹăă

ăôĹăŁd'êĚŹċĹNċŹĎăyĂăyĹăĂŹăyŹăôđēŭăŸŕăĹJăyĂăyĹăŮĠăzŭăy■ăĤăăĤēēĚŹċĹNĎiijNăŔŕăzăēċŋăă  
daemonize() âĠ;ăŤŕċŹĎăĹJăăŔŎēĈăĹăĤăăĤăăzăŹēĚŹăyĹăŮĠăzŭiijNăĹăĤŕăĹJăĹăŤăŹŔċzĹă■ăĹăŮăĹăă  
atexit.register() âĠ;ăŤŕăŝăĹăĤăăŹăŹăyĂăyĹăĠăĠ;ăŤŕăĹJăŤŭŋēġċĠăăŹĹċzĹă■ăĹăŮăĹăġăăŋăăĂĈ  
ăyĂăyĹăŕŕăzăăŎSIGTERMċŹĎăĤăăŔăăđ'ĎċŔĚăŹĹċŹĎăôŹăzĹăŕŋăăŭēĹJăăēĤăċŋăijŸēŹĚċŹĎăĤŝċŮăăĂĈ  
ăĤăăŔăăđ'ĎċŔĚăŹĹċôĂă■ŤċŹĎăĹăăĠăăzăĤē SystemExit() ăijĈăyŹăĂĈ  
ăĹŮēōŷēĚŹăyĂă■ēċĹNăyĹăĹăŎăŝăăĤēēĤăĤiijNăĹăĤŕăŝăăĹJăăôĈiijNă  
ċzĹă■ăĤăăŕăăŕăăĹăijŹă;ĤăĹăŮăy■ăĹăġăăŋ atexit.register()  
ăŝăĹăĤŋċŹĎăyĤċŔĚăŝă;ĹJċŹĎăŮăăĂŹăŕŝăăĹăăŎĹăăŹĚċĈĠăăŹĹăăĂĈ  
ăyĂăyĹăĹăăŎĹăēĚŹċĹNċŹĎăĹăŋăŔăzċăăĂăŔŕăzăăĹJăĹăŤăŹŔăĹJăăŔŎċŹĎ stop  
ăŖăzđ'ċŹĎăŝă;ĹJăy■ċĹNăĹăŕăăĂĈ

ăŹŕăđ'ŹăĤŝăzŎċijŮăĤăăôĹăŁd'êĚŹċĹNċŹĎăĤăăĂăŔŕăzăăŝċĹJăăăĂĹUNIX  
ċŎŕăĈĈċŋŸċŹġċijŮċĹNăăŋ, ċŋŋăzŊċĹĹ by W. Richard  
Stevens and Stephen A. Rago (Addison-Wesley, 2005)ăĂĈ  
ăŕĹċôăăôĈăŸŕăĤŝăŝăĹăŮĈēr■ēĹăċijŮċĹNăiijNăĹăĤŕăĹăĂăĹJăċŹĎăĤăăôŹēĈĹăĂĈŤĹăzŎŤŭŋiijNă  
ăŹăăyŹăĹăĂăĹJăĹăĹăĤăăĤăăŤăăŕăzăăĹJăăăĠăăŹăŝăy■ăĹăĹăĹăĂĈ

## 15 ċŋŋă■ĂăyĹăċŋăiijŹēĎŹăĹJŋċijŮċĹNăyŎċŝzċzŝċôăċŔĚ

ēōŷăđ'Źăzăă;ĤċŤĹŤŭŋăĹJăyŹăyĂăyĹshellēĎŹăĹJŋċŹĎăŹăzăċiijNăŤĹăĹăôđĈŎŕăyŷċŤĹċŝzċzŝăzăăĹăă

Contents:

## 15.1 13.1 éĀŽèĚĜéĜ■āōŽāŘŠ/çóaqéAŞ/æŮĜāzúæŌěāRŮèĹŞāĚě

### éŮóécŸ

äjaäyŊæIJŽä;áčŽĎëĎŽæIJŋæŌěāRŮäzzä;TçŤlæLüèød'äyžæIJĀçóĀā■TçŽĎèĹŞāĚěæŮžāijRāĀCāŊĚæ  
éĜ■āōŽāŘŠæŮĜāzúāLrèrèèĎŽæIJŋijŊæLŮāIJlāS;āzd'èaŊäy■āijæĀŞäyĀäylæŮĜāzúāŘ■æLŮæŮĜāzúāŘ■

### èġcāEşæŮzæaĹ

PythonāEĚç;őçŽĎ fileinput ælāāIŮèōŖèŁŽäylāRŸāĹŮçōĀā■TāĀCæCæđIJā;āæIJL'äyĀäylāyŊéIcé

```
#!/usr/bin/env python3
import fileinput

with fileinput.input() as f_input:
    for line in f_input:
        print(line, end='')
```

éĆcāzĹā;āārseĈ;āzèāL■éIcæRRāĹŖçŽĎæL'ĀæIJL'æŮžāijRæIěäyžæ■d'èĎŽæIJŋæRRāĹŽèĹŞāĚěāĀCāA  
filein.py āzūārEāĚŮāRŸäyžāRfæL'ġèaŊæŮĜāzúīijŊ éĆcāzĹā;āāRfāzèāĈRāyŊéIcéŁŽæāuèĈçŤlāōĈijŊ

```
$ ls | ./filein.py           # Prints a directory listing to stdout.
$ ./filein.py /etc/passwd   # Reads /etc/passwd to stdout.
$ ./filein.py < /etc/passwd # Reads /etc/passwd to stdout.
```

### èőlèőž

fileinput.input() āĹŽāzžāzūèŁTāŽđäyĀäyl FileInput çşçŽĎāōđāĹŊāĀĆ  
èrèāōđāĹŊéŽđ'āžEæŊæIJL'äyĀāzŽæIJL'çŤlçŽĎāyōāĹL'æŮžæşŤād'ŮīijŊāōĈèŁŸāRfècāā;ŞāAŽäyĀäylāyL  
āŽāæ■d'īijŊæŤŖ'āRĹèŭæIēīijŊæCæđIJæĹSāznèçAāEŽäyĀäylæL'Şā■rād'ŽäylæŮĜāzūèĹŞāĜžçŽĎëĎŽæIJŋ

```
>>> import fileinput
>>> with fileinput.input('/etc/passwd') as f:
>>>     for line in f:
...         print(f.filename(), f.lineno(), line, end='')
...
/etc/passwd 1 ##
/etc/passwd 2 # User Database
/etc/passwd 3 #

<other output omitted>
```

éĀŽèŁĜārEāōCā;IJäyžäyĀäylāyLāyŊæŮĜçóaqŘEāZlā;ŁçŤlīijŊāRfāzèçāōāĹlāōĈäy■āE■ā;ŁçŤlæŮūæŮ  
èĀŊäyŤæĹSāznāIJlāzŊāRŌèŁŸæijŤçđ'žāžE FileInput çŽĎāyĀāzŽæIJL'çŤlçŽĎāyōāĹL'æŮžæşŤæIèèŮ



```

parser.add_argument(dest='filenames',metavar='filename', nargs='*')

parser.add_argument('-p', '--pat',metavar='pattern', required=True,
                    dest='patterns', action='append',
                    help='text pattern to search for')

parser.add_argument('-v', dest='verbose', action='store_true',
                    help='verbose mode')

parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')

parser.add_argument('--speed', dest='speed', action='store',
                    choices={'slow','fast'}, default='slow',
                    help='search speed')

args = parser.parse_args()

# Output the collected arguments
print(args.filenames)
print(args.patterns)
print(args.verbose)
print(args.outfile)
print(args.speed)

```

ěřčĺŇăžŘăőŽăzĹ'ăžĚăŷĂăŷłăęĆăŷŇă;ęćŤĺçŽďăŚ;ăzd'ëąŇęğćăđŘăŽĺijŽ

```

bash % python3 search.py -h
usage: search.py [-h] [-p pattern] [-v] [-o OUTFILE] [--speed {slow,
→fast}]

                [filename [filename ...]]

Search some files

positional arguments:
  filename

optional arguments:
  -h, --help            show this help message and exit
  -p pattern, --pat pattern
                        text pattern to search for
  -v                    verbose mode
  -o OUTFILE            output file
  --speed {slow,fast}  search speed

```

ăŷŇéĺćçŽďěČĺăĹĚăĵĤćđ'žăžĚęĺŇăžŘăŷ■çŽďăŤřă■őéČĺăĹĚăĂĆăžŤćžĚęğĆăř\$print()ëř■ăŘęçŽďăĹ'Ś

```

bash % python3 search.py foo.txt bar.txt
usage: search.py [-h] -p pattern [-v] [-o OUTFILE] [--speed {fast,
→slow}]

```

```

        [filename [filename ...]]
search.py: error: the following arguments are required: -p/--pat

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt
filenames = ['foo.txt', 'bar.txt']
patterns   = ['spam', 'eggs']
verbose    = True
outfile     = None
speed       = slow

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt -o_
↪results
filenames = ['foo.txt', 'bar.txt']
patterns   = ['spam', 'eggs']
verbose    = True
outfile     = results
speed       = slow

bash % python3 search.py -v -p spam --pat=eggs foo.txt bar.txt -o_
↪results \
        --speed=fast
filenames = ['foo.txt', 'bar.txt']
patterns   = ['spam', 'eggs']
verbose    = True
outfile     = results
speed       = fast

```

```

    args = parser.parse_args()
    print('Searching for patterns in the following files:')
    for filename in args.files:
        with open(filename) as f:
            for line in f:
                for pattern in args.patterns:
                    if re.search(pattern, line):
                        print(f'Found {pattern} in {filename}')

```

## argparse

```

import argparse
parser = argparse.ArgumentParser()
parser.add_argument('files', help='Files to search in')
parser.add_argument('-p', '--pattern', help='Pattern to search for')
args = parser.parse_args()

```

```

import argparse
parser = argparse.ArgumentParser()
parser.add_argument('files', help='Files to search in')
parser.add_argument('-p', '--pattern', help='Pattern to search for')
parser.add_argument('-s', '--speed', help='Search speed (slow, fast)')
args = parser.parse_args()

```

```

parser.add_argument(dest='filenames', metavar='filename', nargs='*')

```

```

    parser.add_argument(dest='filenames', metavar='filename', nargs='*')
    parser.add_argument(dest='pattern', metavar='pattern', nargs='*')
    parser.add_argument(dest='speed', metavar='speed', nargs='*')

```

```
parser.add_argument('-v', dest='verbose', action='store_true',
                    help='verbose mode')
```

äyÑéíççŽďǺŔĈæŦŕæŎěǺŔŮäyÄäyĴǺ■ŦçŦñǺĀijǺžŭǺŕĒǺĒŭǺ■ŸǺĆĴǺyžǺyÄäyĴǺ■ŮçñēǺyšiiž

```
parser.add_argument('-o', dest='outfile', action='store',
                    help='output file')
```

äyÑéíççŽďǺŔĈæŦŕēŦŕ' æŸŎǺĒǺĒēōyǺ§ŖǺyĴǺŔĈæŦŕēĠǺđ'■ǺĠžçŎŕǺđ'ŽǺñǺiižŦǺžŭǺŕĒǺŎĈǺžñēŦǺ;ǺĴǺǺǺ  
required æǺĠǺŦŮēǺĴđ'žēŕēǺŔĈæŦŕēĠǺŕŦŦēǺǺĒǺĴL'äyÄäyĴǺĈŦŦ ǺŦŦ --pat  
ēǺĴđ'žǺyđ'äyĴǺŔĈæŦŦǺŦ■Ǻ;ćǺijŖēĈǺ;ǺŦŕǺ;ŦçŦĴǺĈ

```
parser.add_argument('-p', '--pat', metavar='pattern', required=True,
                    dest='patterns', action='append',
                    help='text pattern to search for')
```

ǺĴǺǺŦŎŕiižŦǺyÑéíççŽďǺŔĈæŦŕēŦŕ' æŸŎǺŎěǺŔŮäyÄäyĴǺĀijiižŦǺ;ĒæŸŦǺijŽǺŕĒǺĒŭǺŦŦǺŦŕēĈǺ;çŽďǺĒ

```
parser.add_argument('--speed', dest='speed', action='store',
                    choices={'slow', 'fast'}, default='slow',
                    help='search speed')
```

äyÄæŮēǺŔĈæŦŕēǺĴēǺžēćǺŦŦǺǺŎžŽiižŦǺ;ǺǺŕŦǺŦŕǺžēǺĴǺĠēǺŦ  
parser.parse() æŸžǺŦŦǺžĒǺĈ ǺŎĈǺijŽǺđ'ĴçŖĒ sys.argv  
çŽďǺĀijǺžŭēŦŦǺžđǺyÄäyĴçžŦǺđĴǺŎđǺĴŦǺĈ æŦŖǺyĴǺŔĈæŦŦǺĀijǺijŽēćñēōŦç;ŏæĴŖēŕēǺŎđǺĴŦǺy■  
add\_argument() æŸžǺŦŦçŽď dest ǺŦĈæŦŦŦǺŎžçŽďǺŦđæǺĠǺĀijǺĈ

ēŦŸǺ;ĴǺđ'ŽçĠ■ǺĒŭǺžŮæŸžǺŦŦŦēĈǺđŦǺŦǺ;Ǻžđ'ēǺŦēǺĴēǺžǺĈ  
ǺĴŦǺēĈiižŦǺ;ǺǺŦŕēĈ;ǺijŽæĴŦŦǺĴçŽďǺđ'ĴçŖĒ sys.argv æĴŮēǺĒǺ;ŦçŦĴ getopt  
ǺĴǺǺŮǺĈ Ǻ;ĒæŸŦŦijŦǺēĈæđĴǺ;ǺēĠĠçŦĴǺĴŦēĴçŽďæŸžǺijŦŦijŦǺŦĒǺijŽǺĠŦǺŦŦǺ;ĴǺđ'ŽǺĒŮǺ;ŽǺžççǺĀi  
argparse æĴǺǺŮǺŭšçžŦǺyŏǺ;ǺǺđ'ĴçŖĒæžĒǺĈ Ǻ;ǺǺŦŕēĈ;ēŦŸǺijŽççŦǺŦǺ;ŦçŦĴ  
optparse ǺžŦēĠçæđŦēǺĴēǺççŽďǺžççǺĀǺĈ ǺŦçŏǺ optparse ǺŦŦ argparse  
ǺĴĴǺĈŦiižŦǺ;ĒæŸŦŦŦēǺĒæŽŦ' ǺĒĴēŦŽiižŦǺžǺæ■đ' ǺĴĴæŸŦçŽďçĴŦǺžŦǺy■Ǻ;ǺǺžŦēŕēǺ;ŦçŦĴǺŎĈǺĈ

## 15.4 13.4 èŦŖēǺŦǺŮŮǺijžǺĠžǺŦĒçǺǺēŦŦǺĒŦçđ'ž

### éŮŏéçŸ

Ǻ;ǺǺĒžǺžĒǺyĴēďŽæĴŦŦijŦēŦŖēǺŦǺŮŮēĴǺēēǺäyÄäyĴǺŦĒçǺǺǺĈæ■đ'ēďŽæĴŦǺŦŦŦŦǺžđ' ǺžŦǺijŦçŽďŦiiž  
ēǺŦæŸŦŦĴǺēēǺǺijžǺĠžǺyÄäyĴǺŦĒçǺǺēŦŦǺĒŦçđ'žiižŦēŦŦçŦĴǺĴŮēĠǺŭšēŦŦŦǺĒēǺĈ

### èġçǺĒçæŸžǺēǺĴ

ēŦžǺŮŮǺǺžŦŦonçŽď getpass æĴǺǺŮǺ■çæŸŦǺ;ǺæĴ'ǺēĴǺēēǺçŽďǺĈǺ;ǺǺŦŦǺžēŦŦ'Ǻ;ǺǺĴĴē;žǺēĴ;  
ǺžŭǺyŦǺy■ǺijŽǺĴĴçŦĴǺĴŮçžĴŦŦǺžđæŸŦǺŦĒçǺǺǺĈäyÑéíçæŸŦǺĒŭǺ;ŦǺžççǺĀiiž





```
24
>>>
```

## èõíèõž

æIJL'ad'lad'ŽæŮzaijRæIeāĭ ŮçšëçzLčnřad' ġarRāžEijNāzŌerzāRŮçŌřacČāRŸeĠRāLřæL' ġeāNāžTāsČq  
ioctl() āĠ;æTřç■Lç■LāĀĆ äy■èēĠijNāyžāzĀāžLēēAāŌzčāTčl' ůēŁŽāžŽād' ■æIČçŽDāŁđæšTēĀNāy■æ

## 15.6 13.6 æL'ġeāNād'ŮéĆlāŚ;āzd'āžŮēŌŭāRŮāŏČçŽDēĭŠāĠž

### éŮóécŸ

äĭāæČşæL'ġeāNāyĀäyġad' ŮéĆlāŚ;āzd' āžŮāžēPythonā■ŮçņēäyšçŽDāĭčaijRēŌŭāRŮæL'ġeāNçzŞæđIJāĀ

### èġčāĒşæŮzæāĹ

äĭŁçTĭ subprocess.check\_output() āĠ;æTřāĀĆäĭNāēĆrijŽ

```
import subprocess
out_bytes = subprocess.check_output(['netstat', '-a'])
```

èŁŽæŏtāzččāAæL'ġeāNāyĀäyġæNĠāŏŽçŽDāŚ;āzd' āžŮāřEæL'ġeāNçzŞæđIJāžēäyĀäyġā■ŮēŁČā■Ůçņēäy  
āēĆæđIJāĭāēIJāēēAæŮĠæIJāĭčaijRēŁTāŽđrijNāŁāyĀäyġèġčāAæ■ēēġd'ā■şāRřāĀĆäĭNāēĆrijŽ

```
out_text = out_bytes.decode('utf-8')
```

āēĆæđIJēćnæL'ġeāNçŽDāŚ;āzd' āžēēġēŁēŽŭčāAēŁTāŽđrijNāřsaijŽæŁŽāĠžaijČāyŷāĀĆ  
äyNēġčçŽDāĭNā■Řæ■TēŌŭāĹrēTŽēřřāžŮēŌŭāRŮēŁTāŽđçāAijŽ

```
try:
    out_bytes = subprocess.check_output(['cmd', 'arg1', 'arg2'])
except subprocess.CalledProcessError as e:
    out_bytes = e.output          # Output generated before error
    code = e.returncode          # Return code
```

ézŸēŏd' æČĒāĒġāyNrijNcheck\_output() āžĒāžĒēŁTāŽdēĭŞāĒēāĹřæāĠāĠĒēĭŞāĠžçŽDāĀijāĀĆ  
āēĆæđIJāĭāēIJāēēAāŘNæŮŮæTŮēZEæāĠāĠĒēĭŞāĠžāŚNēTŽēřřēĭŞāĠžiijNāĭŁçTĭ stderr  
āŘĆæTrijŽ

```
out_bytes = subprocess.check_output(['cmd', 'arg1', 'arg2'],
                                     stderr=subprocess.STDOUT)
```

āēĆæđIJāĭāēIJāēēAçTġāyĀäyġēŮĒæŮŮæIJžāĹŮæĹæL'ġeāNāŚ;āzd' iijNāĭŁçTĭ timeout  
āŘĆæTrijŽ

```
try:
    out_bytes = subprocess.check_output(['cmd', 'arg1', 'arg2'],
    ↪ timeout=5)
except subprocess.TimeoutExpired as e:
    ...
```

éĀŽāyŷæĭēēōšīijŇāŚ;äzd'čŽĎæL'gëāŇäy■ēIJĀēēAä;£çŤlāĽrāžŤāśĆshellçŎřăċĈīijĽæŕŤæĈshāĀAbash  
 äyĀäyĽā■ŮçņēäyŝāĽŮēāĭāijŽēċŋāijăēĀŚçžŽāyĀäyĽā;ŎçžgçšžçžŝāŚ;äzd'īijŇæŕŤæĈ os.  
 execve() āĀĈāēĈæđIJā;ăæĈșēōĽāŚ;äzd'ēċŋāyĀäyĽshellæL'gëāŇīijŇāijăēĀŚāyĀäyĽā■ŮçņēäyŝāŔĈæŤŕīij  
 shell=True. æIJĽæŮŭāĀŽā;ăæĈșēēAPythonăŎžæL'gëāŇäyĀäyĽād'■æĭĈçŽĎshellāŚ;äzd'čŽĎæŮŭāĀŽē

```
out_bytes = subprocess.check_output('grep python | wc > out',
    ↪ shell=True)
```

ēIJĀēēAæŝĽăĎŔçŽĎæŸŕāIJĽshelläy■æL'gëāŇāŚ;äzd'āijŽā■ŸāIJĽāyĀăōŽçŽĎăōĽăĒĭēċŎēŽĽīijŇçĽžāĽ  
 èĤZæŮŭāĀŽāŔŕăžă;£çŤĭshlex.quote() āĢ;æŤŕæĭēēōšāŔĈæŤŕæ■ççăōçŽĎçŤlāŔŇāijŤçŤlāijŤēŭæĭēā

## ēōĭēōž

ä;£çŤĭcheck\_output() āĢ;æŤŕæŸŕæL'gëāŇād'ŮēĈĭāŚ;äzd'āžŭēŎŭāŔŮăĒŭēĤŤăŽđăĀijçŽĎæIJĀç  
 ä;EæŸŕīijŇăēĈæđIJā;ăēIJĀēēAŕŕžă■ŔēĤŽçĭŇăĀŽæŽŕăd'■æĭĈçŽĎăžđ'ăžŖīijŇæŕŤæĈççžŽăōĈăŔŖŝéĀAē;Ŗă  
 èĤZæŮŭāĀŽāŔŕçŽŕăŎēä;£çŤĭsubprocess.PopençšžăĀĈă;ŇăēĈīijŽ

```
import subprocess

# Some text to send
text = b'''
hello world
this is a test
goodbye
'''

# Launch a command with pipes
p = subprocess.Popen(['wc'],
    stdout = subprocess.PIPE,
    stdin = subprocess.PIPE)

# Send the data and get the output
stdout, stderr = p.communicate(text)

# To interpret as text, decode
out = stdout.decode('utf-8')
err = stderr.decode('utf-8')
```

subprocess æĭāāĭŮăŕžăžŎă;ĭēŤŮTTYçŽĎăđ'ŮēĈĭāŚ;äzd'äy■ăŔĽéĀĈçŤĭāĀĈ  
 ä;ŇăēĈīijŇă;ăäy■ēĈ;ä;£çŤlăōĈæĭēēĢlāĽlāŇŮāyĀäyĽçŤlăĽŭē;ŖăĒēăŕEçăĀçŽĎăžžăĽāīijĽæŕŤæĈăyĀäyĽs  
 èĤZæŮŭāĀŽīijŇă;ăēIJĀēēAä;£çŤlāĽŕçŋŋāyĽæŮžæĭāāĭŮăžEīijŇæŕŤæĈăŝžăžŎēŖŮăŔ■çŽĎ  
 expectăōŭăŮŔçŽĎăŭēăĒŭīijĽpexpectæĽŮçšžăīijçŽĎīijĽ

## 15.7 13.7 ád'■áLúæLÚèĀĔçğzâLíæÚĜäzúâSŇçZóâ;T

### éÚóécŸ

ä;ăæĈşèeAâd'■áLúæLÚçğzâLíæÚĜäzúâSŇçZóâ;TüjNă;EæŸřăRĹäy■æĈşèrĈçTĭshellâS;ăzd'ăĀĈ

### èğĉăEşæÚzæąĹ

shutil æĹąĹŮæIJĹ'ăĹLăd'Žă;£æ■űçŽDăĜ;æTřăRřăzèăd'■áLúæÚĜäzúâSŇçZóâ;TăĀĈă;£çTĭèŮæĪéé

```
import shutil

# Copy src to dst. (cp src dst)
shutil.copy(src, dst)

# Copy files, but preserve metadata (cp -p src dst)
shutil.copy2(src, dst)

# Copy directory tree (cp -R src dst)
shutil.copytree(src, dst)

# Move src to dst (mv src dst)
shutil.move(src, dst)
```

èĔŽăžZăĜ;æTřçŽDăRĈæTřéĈ;æŸřă■Ůçņăyşă;ĉăijRçŽDæÚĜäzúæLŮçZóâ;TăR■ăĀĈ  
ăžTăşĈér■ăzĹæĹæNşăžEçşzăijjçŽDUnixăŞ;ăzd'üjNăeĈăyĹéĪççŽDæşĹéĜĹéĈĹăĹEăĀĈ

ézŸèôd'æĈĒăEġăyNüjNăřzăžŌçņăRŮéŞ;æŌèèĀNăŮşèĔŽăžZăŞ;ăzd'ăd'ĎçRĒçŽDæŸřăŌĈæNĜăRŞçŽ  
ăĹNăeĈüjNăeĈădIJæŽRæÚĜäzúæŸřăŸĀăyĹçņăRŮéŞ;æŌëüjNéĈĉăzĹçZóæăĜæÚĜäzúăřEăijŽæŸřçņăRŮé  
ăeĈădIJă;ăăRĹæĈşăd'■ăĹŮçņăRŮéŞ;æŌëăIJnèznüjNéĈĉăzĹéIJăèeAæNĜăŏŽăĔşéTŏă■ŮăRĈæTř  
follow\_symlinks,ăeĈăyNüjŽ

ăeĈădIJă;ăæĈşăĪçTŽèĉnăd'■ăĹŮçZóâ;Tăy■çŽDçņăRŮéŞ;æŌëüjNăĈRèĔŽæăŮăĀŽüjŽ

```
shutil.copytree(src, dst, symlinks=True)
```

copytree() âRřăzèèŌ'ă;ăăIJăd'■ăĹŮèĔĜĈĹNăy■éĀĹæNĹ'æĀğçŽDăĪçTěæşRăžZæÚĜäzúæLŮçZóâ  
ă;ăăRřăzèæRŘă;ŽăyĀăyĹăĪçTěăĜ;æTřüjNăŌëăRŮăyĀăyĹçZóâ;TăR■ăSŇæÚĜäzúăR■ăĹŮèăĹă;IJăyžè;ŞăĔ

```
def ignore_pyc_files(dirname, filenames):
    return [name in filenames if name.endswith('.pyc')]

shutil.copytree(src, dst, ignore=ignore_pyc_files)
```

çTşăžŌăĪçTěæşRçğ■æĹąĹjRçŽDæÚĜäzúăR■æŸřăĹăyŸèğAçŽDüjNăŽăæ■d'ăyĀăyĹă;£æ■űçŽDăĜ;æ  
ignore\_patterns() âŮşçzRăNĒăRňăIJĪéĜNĪéĪçăĒEăĀĈă;NăeĈüjŽ

```
shutil.copytree(src, dst, ignore=shutil.ignore_patterns('*~', '*.pyc  
→'))
```

## èõléõž

ä;£çŦĭ shutil ād'■āLūæŨĜāzūāŠŇçZōā;ŦāžšāfŠçōĀā■ŦāžEçCzāRġāĀĆ  
äy■è£ĜĭjŇārzážŌæŨĜāzūāĒĈæŦræ■ōāfæAŕĭjŇcopy2() è£ZæāũçZĎāĜ;æŦŕāRĭèĈ;āŕ;èĜlāũsæIJĀād'ġ  
èõ£éŨōæŨūéŨŕ'āĀAāLZāzzæŨūéŨŕ'āŠŇæiĈéZŖè£ZāžZāšžæIJñāfæAŕāijZēcñāfĬçŦZĭjŇ  
ä;EæŸŕárzážŌæL'ĀæIJL'èĀĒāĀACLsāĀAèŦĎæžŖforkāŠŇāĒūāzŨæZŦ'æũsāCæñaçZĎæŨĜāzūāĒĈāfæA  
è£Zāyĭè£Ÿā;Ũā;ĬèŦŨāžŌāžŦāsĈæS■ä;IJçšççzçççšāđNāŠŇçŦĭæLūæL'ĀæŇæIJL'çZĎèõ£éŨōæiĈéZŖāĀĆ  
ä;āéĀZāyŷäy■āijZāŌžā;£çŦĭ shutil.copytree() āĜ;æŦŕæĭæL'ġeāŇçççççšāđ'Ĝāz;āĀĆ  
ā;Šāđ'ĎçRĒæŨĜāzūāŖ■çZĎæŨūāĀZĭjŇæIJĀāē;ä;£çŦĭ os.path  
äy■çZĎāĜ;æŦŕæĭççāōāfĬæIJĀād'ġçZĎāŖççğzæđ'■æĀġĭjĬçL'zāLŇæŸŕāŖŇæŨūèçAéĀĆçŦĭāžŌUnixāŠŇW  
ä;ŇāēĈĭjZ

```
>>> filename = '/Users/guido/programs/spam.py'
>>> import os.path
>>> os.path.basename(filename)
'spam.py'
>>> os.path.dirname(filename)
'/Users/guido/programs'
>>> os.path.split(filename)
('/Users/guido/programs', 'spam.py')
>>> os.path.join('/new/dir', os.path.basename(filename))
'/new/dir/spam.py'
>>> os.path.expanduser('~/' + 'guido/programs/spam.py')
'/Users/guido/programs/spam.py'
>>>
```

ä;£çŦĭ copytree() ād'■āLūæŨĜāzūāđ'žçZĎāyĀāyĭæçŸæL'ŇçZĎæŨōéçŸæŸŕárzážŌēŦZèŕŕçZĎād'Ĭ  
ä;ŇāēĈĭjŇāIJĀād'■āLūè£ĜĬŇāy■ĭjŇāĜ;æŦŕāRĭèĈ;āijZççŕāLŕæ■šāĬŖçZĎçñæŖūéSç;æŌēĭjŇāZāyŷæiĈéZ  
äyžāžEèġçĀEşè£Zāyĭè£ŨōéçŸĭjŇæL'ĀæIJL'ççŕāLŖçZĎæŨōéçŸāijZēcñæŦūéZEāLŕāyĀāyĭāLŨeāĭäy■āzūæL'S  
äyŇéĬæŸŕāyĀāyĭā;Ňā■ŖĭjZ

```
try:
    shutil.copytree(src, dst)
except shutil.Error as e:
    for src, dst, msg in e.args[0]:
        # src is source name
        # dst is destination name
        # msg is error message from exception
        print(dst, src, msg)
```

æçĆāđIJā;āæŖŖä;ZāĒşéŦōā■ŨāŖĆæŦŕ ignore\_dangling\_symlinks=True ĭjŇ  
è£ZæŨūāĀZ copytree() āijZāf;çŦŕæŌL'æŨāæŦĬçñæŖūéSç;æŌēāĀĆ

æIJñèLĆæijŦçđ'žçZĎè£ZāžZāĜ;æŦŕèĈ;æŸŕæIJĀāyŷèġAçZĎāĀĆäy■è£ĜĭjŇshutil  
è£ŸæIJL'æZŦ'ād'ZçZĎāŠŇād'■āLūæŦŕæ■ōçZŸāĒşçZĎæS■ä;IJāĀĆ  
āōÇçZĎæŨĜæaçā;ĬāĀijā;ŨāyĀçIJŇĭjŇāŖCèĀĆ Python documentation

## 15.8 13.8 aŁŻazzãŠNèġcãŌNã;ŠæaçæŮĠzú

### éŮóécŸ

ä;äeIJÄeëAãŁŻazzæŁŮèġcãŌNãÿyëġAæäijâijRçŽĐã;ŠæaçæŮĠzúiiijŁæfŤæĈ.tar,  
.tgzæŁŮ.zipiiijL

### èġcãEşæŮzæaŁ

shutil æŁaãIŮæNëæIJL'äyd'äyŁãĠ;æŤrãĀŤãĀŤ make\_archive() åŠN  
unpack\_archive() åŖræt'ġäyŁçŤÍãIJzãĀĈ äġNãĈiiijŽ

```
>>> import shutil
>>> shutil.unpack_archive('Python-3.3.0.tgz')

>>> shutil.make_archive('py33', 'zip', 'Python-3.3.0')
'/Users/beazley/Downloads/py33.zip'
>>>
```

make\_archive() çŽĐçññãžNäyŁãŖĈæŤræŸræIJşæIJŽçŽĐèġŞãĠzæäijâijRãĀĈ  
åŖrãžëã;ġçŤÍ get\_archive\_formats() èŌŮãŖŮæL'ÄæIJL'æŤræŤAçŽĐã;ŠæaçæäijâijRãŁŮèãŁãĀĈäġN

```
>>> shutil.get_archive_formats()
[('bztar', "bzip2'ed tar-file"), ('gztar', "gzip'ed tar-file"),
 ('tar', 'uncompressed tar file'), ('zip', 'ZIP file')]
>>>
```

### èŏlèöž

PythonèŸŸæIJL'ãĒŮãžŮçŽĐæŁaãIŮãŖrçŤÍæIëãd'ĐçŖEãd'Žçġ■ã;ŠæaçæäijâijRiiijŁæfŤæĈtarfile,  
zipfile, gzip, bz2iiijLçŽĐãžŤãšĈçzEèŁĈãĀĈ äy■èŸĠiiijNãĈæđIJã;ääžĒãžĒãŖŁæŸrëëAãŁŻazzæŁŮæŖŖãŖŮ  
åŖrãžëçŽŤ æŌëã;ġçŤÍ shutil äy■çŽĐèŸŽãžŽénŸãšĈãĠ;æŤrãĀĈ

èŸŽãžŽãĠ;æŤrëŸŸæIJL'ãġŁãd'ŽãĒŮãžŮéĀŁ'éãžiiijNçŤÍãžŌæŮëãŸŮæL'Şã■ŖãĀæćĐæĈĀãĀAæŮĠzú  
åŖĈèĀĈ shutilæŮĠzæaç

## 15.9 13.9 éĀŽèĠĠæŮĠzúãŖ■æşæeL'ġæŮĠzú

### éŮóécŸ

ä;äeIJÄeëAãEŽäyÄäyŁæŮL'ãŖŁãŁŖæŮĠzúæşæeL'ġæŞ■ä;IJçŽĐèĐŽæIJñiiijNãfŤæĈãŖzæŮëãŸŮã;Šæa  
ä;ääy■æĈşãIJÍPythonèĐŽæIJñäy■ërĈŤÍshelliiijNæŁŮèĀĒä;äëëAãŏđçŌŖãŸĀãžŽshelläy■èĈ;ãĀŽçŽĐãŁşèĈ

## èġċàEşæŮzæąĹ

æşæLĹæŮĠăzŭiijŃăŔŕăĴęŦĬos.walk() åĠjæŦŕiijŃăiĵăăyĂăyĹeăŭċžġçZŏăĴăŔ■çzZăŏČăĂĆ  
ăyŃéĬæŸŕăyĂăyĹăĴŃă■ŔŕiijŃăşæLĹċĹzăŏŽçŽDæŮĠăzŭăŔ■ăzŭċ■ŦăžŦæĹ'ĂæIJĹċņęăŔĹæĬăzŭċŽDæŮ

```
#!/usr/bin/env python3.3
import os

def findfile(start, name):
    for relpath, dirs, files in os.walk(start):
        if name in files:
            full_path = os.path.join(start, relpath, name)
            print(os.path.normpath(os.path.abspath(full_path)))

if __name__ == '__main__':
    findfile(sys.argv[1], sys.argv[2])
```

ăĬĬăŮŸeĐŽæIJăyžæŮĠăzŭfindfile.pyiijŃçĐŭăŔŎăIJĬăŦŦăzd'èăŃăy■æĹġèăŃăŏČăĂĆ  
æŃĠăŏŽăĬĬăġŃăşæLĹċçZŏăĴăžăŔĹăŔ■ăŮăĬJăyžăĴ■çĴăŔĆæŦŕiijŃăęCăyŃiijŽ

## èőĬèőž

os.walk() æŮzæşŦăyžæĹŦăzŋéA■ăŎEçZŏăĴăæŦiijŃ  
ăŕŔăŋăęŦZăĬëăyĂăyĹçZŏăĴŕiijŃăŏČăiijŽeŦŦăZđăyĂăyĹăyĹ'ăĬČçzĐŕiijŃăŃEăŔŋçŽyăŕzăžŎăşæLĹċçZŏăĴ  
ăžăăŔĹéČăyĹçZŏăĴăyŃéĬçŽDæŮĠăzŭăŔ■ăĬŮeăĬăĂĆ

ăŕzăžŎăŕŔăyĹăĬČçzĐŕiijŃăŔĬeIJăcĂăŦŃăyĂăyŃçZŏăăĠæŮĠăzŭăŔ■æŸŕăŔeăIJăŮĠăzŭăĬŮeăĬăy■  
os.path.join() âŔĹăzŭeŭŕăĴĐăĂĆ äyžăžEęĂăĬăE■ăĬGæĂĹçŽĐeŭŕăĴĐăŔ■ăŕŦăęĆ ./  
./foo//bar iijŃăĴęŦĬăžEăŔeăđ'Ůăyđ'ăyĹăĠjæŦŕæĬeăĬŏæ■ççzşæđIJăĂĆ çŋăăyĂăyĹæŸŕ  
os.path.abspath() ,ăŏČăŎeăŔŮăyĂăyĹeŭŕăĴĐŕiijŃăŔŕeČjæŸŕçŽyăŕzeŭŕăĴĐŕiijŃăIJăăŔŎeŦăŽđçzĬă  
çŋăăžŃăyĹæŸŕos.path.normpath() iijŃçŦĬăĬeēŦăŽđæ■căyŷeŭŕăĴĐŕiijŃăŔŕăžeēġċăEşăŔŃæŮIJăĬEă

ăŕçŏăęŦZăyĹeĐŽæIJŋçŽyăŕzăžŎUNIXăžşăŔŕăyĹéĬçŽĐăĴăĬăđ'ŽæşæLĹæĬeēŏşeęAçŏĂăŦăĴăĬăđ'Žŕiij  
ăzŭăyŦŕiijŃeŦŸeČĴăĴeĴzăĬçŽĐăĬăăĬăĬăŮăzŮçŽĐăĬşeČĴăĂĆ  
æĹŦăzŋăE■ăiijŦçđ'žăyĂăyĹăĴŃă■ŔŕiijŃăyŃéĬçŽĐăĠjæŦŕæĹŦşă■ŕæĹ'ĂæIJĹ'æIJăĬŦŦeĬŦăĬŦăžēĬĠççŽDæŮ

```
#!/usr/bin/env python3.3

import os
import time

def modified_within(top, seconds):
    now = time.time()
    for path, dirs, files in os.walk(top):
        for name in files:
            fullpath = os.path.join(path, name)
            if os.path.exists(fullpath):
                mtime = os.path.getmtime(fullpath)
                if mtime > (now - seconds):
                    print(fullpath)
```

```

if __name__ == '__main__':
    import sys
    if len(sys.argv) != 3:
        print('Usage: {} dir seconds'.format(sys.argv[0]))
        raise SystemExit(1)

    modified_within(sys.argv[1], float(sys.argv[2]))

```

aIJlæ■d' aG;æTřčŽDāšžçaĀázNäyŁiijŃä;ŁçTłos,os.path,globç■ŁçśzäijijæłāłŮiijŃä;ääřsèČ;ăôđçŎřæŽł  
 ăŔăŔăŔăĈăĈ5.11ăŔăŔăĈăĈ5.13ăŔăŔăĈăĈ■ŁçŽyăĖșçñăĖĈăĈĈ

## 15.10 13.10 əržāŔŮéĖ■ç;ŏæŮĜäzŮ

éŮŏécŸ

æĀŎæăŭèržāŔŮæŽŏéĀŽ.iniaĕijăijŔçŽDēĖ■ç;ŏæŮĜäzŮiijš

èğčăĖșçŮžæąŁ

configparser æłāłŮèČ;ĕcñçTłæłĕëržāŔŮéĖ■ç;ŏæŮĜäzŮăĈă;ŃăĕĆiijŃăĀĜĕŏçă;ăæIJL'ăĕĆăyŃç

```

; config.ini
; Sample configuration file

[installation]
library=%(prefix)s/lib
include=%(prefix)s/include
bin=%(prefix)s/bin
prefix=/usr/local

# Setting related to debug configuration
[debug]
log_errors=true
show_warnings=False

[server]
port: 8080
nworkers: 32
pid-file=/tmp/spam.pid
root=/www/root
signature:
=====
Brought to you by the Python Cookbook
=====

```

äyŃéłćæŸřäyĀäyłĕržāŔŮăŖăŔăŔăŮăĖŮäy■ăĀijçŽDă;Ńă■ŔiijŽ

```

>>> from configparser import ConfigParser
>>> cfg = ConfigParser()
>>> cfg.read('config.ini')
['config.ini']
>>> cfg.sections()
['installation', 'debug', 'server']
>>> cfg.get('installation', 'library')
'/usr/local/lib'
>>> cfg.getboolean('debug', 'log_errors')

True
>>> cfg.getint('server', 'port')
8080
>>> cfg.getint('server', 'nworkers')
32
>>> print(cfg.get('server', 'signature'))

\=====
Brought to you by the Python Cookbook
\=====
>>>

```

```

        cfg.write()

```

```

>>> cfg.set('server', 'port', '9000')
>>> cfg.set('debug', 'log_errors', 'False')
>>> import sys
>>> cfg.write(sys.stdout)

```

```

[installation]
library = %(prefix)s/lib
include = %(prefix)s/include
bin = %(prefix)s/bin
prefix = /usr/local

[debug]
log_errors = False
show_warnings = False

[server]
port = 9000
nworkers = 32
pid-file = /tmp/spam.pid
root = /www/root
signature =
    =====
    Brought to you by the Python Cookbook
    =====
>>>

```



## ëõléõž

éĚ■;õæŮĜäzũä;IJäyžäyÄçġ■āRrèræĀğāŁāē;çŽDæäijäijRiijNéIdäyýéĀĆçŦlāžŌ■YāĆlćlNāžRäy■;ç  
āIJlærRäyłéĚ■;õæŮĜäzũäy■iijNéĚ■;õæŦræ■õäijŽècñāŁēçžDiiJLærŦāēCä;Nā■Räy■çŽDāĀIInstallationā  
āĀIdebugāĀI āŠŇ āĀIserverāĀIiijL'āĀĆ ærRäyłāŁēçžDāIJlāĚũäy■æNĜāõŽārzāžŦçŽDāRĎäyłāRŸéĠRāĀ

āržāžŌāRrāõđçŌrāRŇæäüāŁšèC;çŽDēĚ■;õæŮĜäzũāŠŇPythonæžRæŮĜäzũæYræIJL'ā;Łād'ğçŽDäy■  
éēŮāĚĹiijNéĚ■;õæŮĜäzũçŽDēr■æšŦēēAæZŦ'èĠçŦsäžZiijNäyNéÍççŽDētNāĀijēr■āRēæYrç■L'æŦŁçŽDiiJ

```
prefix=/usr/local
prefix: /usr/local
```

éĚ■;õæŮĜäzũäy■çŽDāR■ā■ŮæYrāy■āNžāŁēād'ğārRāēŽçŽDāĀCä;NāēCiiJŽ

```
>>> cfg.get('installation', 'PREFIX')
'/usr/local'
>>> cfg.get('installation', 'prefix')
'/usr/local'
>>>
```

āIJlēğçæđRāĀijçŽDæŮüāĀŽiijNgetboolean() æŮzæšŦæšæL'çäzzä;ŦāRrēāNçŽDāĀijāĀCä;NāēC

```
log_errors = true
log_errors = TRUE
log_errors = Yes
log_errors = 1
```

æŁŮëõyēĚ■;õæŮĜäzũāŠŇPythonäžççāAæIJĀād'ğçŽDäy■āRŇāIJlāžŌiijNāõCāzũäy■æYrāzŌäyŁēĀŇ  
æŮĜäzũæYrāõŁ'ècĚäyĀäyłæŦŦ'ä;ŠècñēržāRŮçŽDāĀĆāēCæđIJççrāŁrāžEāRŸéĠRæŽŁæ■ciiJNāõCāõđéŽĚā  
ä;NāēCiiJNāIJlāyNéÍcēŁŽäyłéĚ■;õäy■iijNprefix āRŸéĠRāIJlā;ŁçŦlāõCçŽDāRŸéĠRāzNāL'■æŁŮäzNāŦ

```
[installation]
library=%(prefix)s/lib
include=%(prefix)s/include
bin=%(prefix)s/bin
prefix=/usr/local
```

ConfigParser æIJL'äyłāõžæYšècñāŁ;èġEçŽDçŁ'žæĀğæYrāõCèC;äyĀæñæfzāRŮād'ŽäyłéĚ■;õæŮ  
ä;NāēCiiJNāĀĠëõ;äyĀäyłçŦlæŁuāČRäyNéÍcēŁŽæäüāđĎēĀäžEäzŮäzñçŽDēĚ■;õæŮĜäzũiijŽ

```
; ~/.config.ini
[installation]
prefix=/Users/beazley/test

[debug]
log_errors=False
```

ēržāRŮēŁŽäyłæŮĜäzũiijNāõCārseC;ëüşäzNāL'■çŽDēĚ■;õāRŁāzũëŦuælēāĀCāēCiiJŽ

```
>>> # Previously read configuration
>>> cfg.get('installation', 'prefix')
```

```

'/usr/local'

>>> # Merge in user-specific configuration
>>> import os
>>> cfg.read(os.path.expanduser('~/.config.ini'))
['/Users/beazley/.config.ini']

>>> cfg.get('installation', 'prefix')
'/Users/beazley/test'
>>> cfg.get('installation', 'library')
'/Users/beazley/test/lib'
>>> cfg.getboolean('debug', 'log_errors')
False
>>>

```

azTçzEëgCårşäyN prefix aRÿéGRæYræAÕæauëçEçZÚaEüazŮçZyâEşâRÿéGRçZDrijNæfTæC  
 library çZDèç;ãoZâAijãÄC äžgçTşèçZçg■çzŞædIJçZDãŮşâZæYrâRÿéGRçZDæTzâEŽéGĞâRŮçZDæ  
 äjääRfäzëâCRäyNéIcèçZæâuaAŽerTéIÑiijZ

```

>>> cfg.get('installation', 'library')
'/Users/beazley/test/lib'
>>> cfg.set('installation', 'prefix', '/tmp/dir')
>>> cfg.get('installation', 'library')
'/tmp/dir/lib'
>>>

```

æIJâRÕèçYæIJLâç;LéG■èçAäyÄçCzèçAæşláæDRçZDæYrPythonâžüäy■èÇjæTfræNÄ.iniaŮGäzûâIJlä  
 çãoäçlâj;âauşçzRâRÇéYËäžEçconfigparseræŮGæaçäy■çZDèç■æşTèççæÇËäzëâRŁæTfræNÄçL'zæÄğãÄC

## 15.11 13.11 çzZçõÄâ■TèDŽæIJnăcđâŁăæŮëâ£ŮâŁşèÇj

### éŮóéçY

äjääyNæIJZâIJlèDŽæIJnăŞNçlNăžRäy■ârEçrŁæŮ■âçæAřâEŽâËæŮëâ£ŮâŮGäzûãÄC

### èğçâEşæŮzæaŁ

æL'Şâ■ræŮëâ£ŮæIJÄçõÄâ■TæŮžâijRæYrâjççTl logging ælââIŮâÄÇäçNæçCrijZ

```

import logging

def main():
    # Configure the logging system
    logging.basicConfig(
        filename='app.log',
        level=logging.ERROR
    )

```

```

# Variables (to make the calls that follow work)
hostname = 'www.python.org'
item = 'spam'
filename = 'data.csv'
mode = 'r'

# Example logging calls (insert into your program)
logging.critical('Host %s unknown', hostname)
logging.error("Couldn't find %r", item)
logging.warning('Feature is deprecated')
logging.info('Opening file %r, mode=%r', filename, mode)
logging.debug('Got here')

if __name__ == '__main__':
    main()

```

äyŁÉİcāZTāyŁæŮēāŁŮērÇçTīijŁcritical(), error(), warning(), info(), debug()īijLāžēēZāāŖæŮāijŖēāŁçd'žāyāŖŇçŽDāyēēĠçžgāŁnāĀĆ basicConfig() çŽD level āŖCæŤŖæŸŖāyĀāyŁēŁGæzd'āZīāĀĆ æLĀæIJŁçžgāŁnā;ŌāžŌæġd'çžgāŁnçŽDæŮēāŁŮūŁæAŖēČ;āijŽēcāŁçTēæŌLāĀĆ æŖŖāyŁloggingæ\$ā;IJçŽDāŖCæŤŖæŸŖāyĀāyŁæūŁæAŖāŮçñēāyšīijŇāŖŌēİcāEġēū\$āyĀāyŁæLŮād'ŽāyŁāŖC ædDēĀāæIJĀçZŁçŽDæŮēāŁŮūŁæAŖçŽDæŮūāĀZæŁSāznā;ŁçTīlāžE%æ\$ā;IJçñēæİēæāijāijŖāŇŮæūŁæA

ēŁŖēāŇēŁZāyŁçİŇāžŖāŖŌīijŇāIJŁæŮĠāzū app.log āyçŽDāEĀāōzāžTēŖēæŸŖāyŇēİcēŁZæūīijŽ

```

CRITICAL:root:Host www.python.org unknown
ERROR:root:Could not find 'spam'

```

āēČædIJā;āæČşæŤžāŖŸēŁŞāĠžçġLçžgīijŇā;āāŖŖāžēāŁōæŤž basicConfig() ēŖČçTīlāyçŽDāŖCæŤŖāĀĆāŁŇāēČīijŽ

```

logging.basicConfig(
    filename='app.log',
    level=logging.WARNING,
    format='%(levelname)s: %(asctime)s: %(message)s')

```

æIJĀāŖŌēŁŞāĠžāŖŸæŁŖāēČāyŇīijŽ

```

CRITICAL:2012-11-20 12:27:13,595:Host www.python.org unknown
ERROR:2012-11-20 12:27:13,595:Could not find 'spam'
WARNING:2012-11-20 12:27:13,595:Feature is deprecated

```

äyŁÉİcçŽDæŮēāŁŮēĠç;ōēČ;æŸŖçāñçijŮçāĀāŁŖçİŇāžŖāyçŽDāĀĆāēČæČædIJā;āæČşā;ŁçTīlēĠç;ōæŮŮ āŖŖāžēāČŖāyŇēİcēŁZæūāāŁōæŤž basicConfig() ēŖČçTīijŽ

```

import logging
import logging.config

def main():
    # Configure the logging system

```

```
logging.config.fileConfig('logconfig.ini')
...
```

álZázžäyÄäyłäyNéíCèŁŻæăũçŽĐæŮĠäzũĩijŇăŘ■ă■ŮăŘń logconfig.ini řijŽ

```
[loggers]
keys=root

[handlers]
keys=defaultHandler

[formatters]
keys=defaultFormatter

[logger_root]
level=INFO
handlers=defaultHandler
qualname=root

[handler_defaultHandler]
class=FileHandler
formatter=defaultFormatter
args=('app.log', 'a')

[formatter_defaultFormatter]
format=%(levelname)s: %(name)s: %(message)s
```

ăĕĆăđIJă;ăăĈşăŁăŤzéĚ■ç;őĩijŇăŔřăžĕçŽť æŎĕçijŮĕŁŠăŮĠäzũlogconfig.iniă■şăŔřăĂĆ

## èõlèõž

ăřĵčőăřzăžŎ logging æłăăİŮĕĂŇăũşæIJL'ăŁŁăđ'ŽæŽť éńŸçžġçŽĐĕĚ■ç;őéĂL'ėąžĩijŇ  
äy■ĕŁĠĕŁŻĕĠŇçŽĐæŮžæăŁăŕžăžŎčőĂă■ŤçŽĐçİŇăžŔăŞŇĕĐŽæIJňăũşçžŔĕũşăđ'şăžĖăĂĆ  
ăŔłăĈşăIJĕŕĈçŤłăŮĕăŁŮăŞ■ă;IJăL'■ăĚŁăL'ġĕăŇăyŇbasicConfig()ăĠ;æŤŕæŮžæşŤĩijŇă;ăçŽĐçİŇăžŔăŕşĕ

ăĕĆăđIJă;ăăĈşĕĖAă;ăçŽĐæŮĕăŁŮăŮŁăAŕăĖŽăĹŕăăĠăĠĖĖĖŤŽĕŕŕăy■ĩijŇĕĂŇăy■æŸŕæŮĕăŁŮăŮĠäzũ  
basicConfig() æŮŮăy■ăijăæŮĠäzũăŔ■ăŔĆæŤŕă■şăŔŕăĂĆăŁŇăĕĆĩijŽ

```
logging.basicConfig(level=logging.INFO)
```

basicConfig() äIJĴİŇăžŔăy■ăŔłĕÇĵĕćŇăL'ġĕăŇăyĂăňăăĂĆăĕĆăđIJă;ăçİ■ăŔŎăĈşăŤžăŔŸæŮĕă  
ăŕşĕIJăĕĖAăĚŁĕŎăŔŮ root logger řijŇçĐŮăŔŎçŽť æŎĕăŁăŤžăőĈăĂĆăŁŇăĕĆĩijŽ

```
logging.getLogger().level = logging.DEBUG
```

éIJĂĕĖAăijžĕŕĈçŽĐæŸŕæIJňĕŁĈăŔłăŸŕæijŤĈđ'žăžĖ logging  
æłăăİŮçŽĐăyĂăžŽăşžæIJňçŤłăşŤăĂĆ äőĈăŔŕăžĕăAŽæŽťăđ'ŽæŽť éńŸçžġçŽĐăőŽăĹăŮăĂĆ  
ăĚşăžŎăŮĕăŁŮăőŽăĹăŮăŮŮăyĂăyłăŁăĕĵçŽĐĕťĐăžŔăŸŕ [Logging Cookbook](#)



```

>>> import logging
>>> logging.basicConfig(level=logging.ERROR)

>>> import somelib
>>> somelib.func()
CRITICAL:somelib:A Critical Error!

>>> # Change the logging level for 'somelib' only
>>> logging.getLogger('somelib').level=logging.DEBUG
>>> somelib.func()
CRITICAL:somelib:A Critical Error!
DEBUG:somelib:A debug message
>>>

```

aIJlæfZéGÑrijNæāzæUëåfUëcñÉ■ç;õæLRäzËäzËë;ŞăGžERRORæLŮæZt'énYçžgăĹnçŽDæŭLæAřăĂ  
 äy■èfGrijNsomelibçŽDæUëåfUçžgăĹnècñă■TçNñéÉ■ç;õæLRăRřazëë;ŞăGždebugçžgăĹnçŽDæŭLæAřăĂ  
 âČRèfZæăuæZt'æTză■TçNñæĹaĹUçŽDæUëåfUëÉ■ç;õăřzăžŎërČerTæĹëèõşæYřă;ĹæŮză;ŁçŽDrijN  
 âZăäyžă;ăæUăéIJĀăŎzæZt'æTzăžă;TçŽDăĹĹsĀæUëåfUëÉ■ç;õăĀTăĀTăRĹéIJĀëAăfõæTză;ăæČşëAæZ

Logging HOWTO èřęçzEäzNçz■ăžEăëČă;TéÉ■ç;õæUëåfUăĹaĹUăŠNăĚŭäzŮæIJLçTĹæĹĂăŭgrijNăRřă

## 15.13 13.13 áódçŎřăyĂăyĹëóæUŭăZĹ

### éUőécŸ

ă;ăæČşëõřă;TçĹNăžRæL'gëaŃăd'ŽăyĹăzzăĹăæL'ĂèĹsèt'zçŽDæUŭéŮt'

### èğcăEşæŮzæaĹ

time æĹaĹUăNĚăRňă;Ĺăd'ŽăG;æTřæĹëæL'gëaŃëuşæUŭéŮt'æIJL'ăĚşçŽDăG;æTřăĂČ  
 âř;çõăăČă■d'rijNéĂŽăyăæĹSăžňăijŽăIJă■d'ăşžçăĂăžNăyĹădĐăĂăyĂăyĹæZt'énYçžgçŽDæŎăRčæĹæ

```

import time

class Timer:
    def __init__(self, func=time.perf_counter):
        self.elapsed = 0.0
        self._func = func
        self._start = None

    def start(self):
        if self._start is not None:
            raise RuntimeError('Already started')
        self._start = self._func()

    def stop(self):
        if self._start is None:
            raise RuntimeError('Not started')

```

```

        end = self._func()
        self.elapsed += end - self._start
        self._start = None

    def reset(self):
        self.elapsed = 0.0

    @property
    def running(self):
        return self._start is not None

    def __enter__(self):
        self.start()
        return self

    def __exit__(self, *args):
        self.stop()

```

ẽƒŽäyłçsżãõŽázL'ázEäyÄäyłãRřäzèècńçŦłæŁũæǻžæ■óéIJǺèçAǻRřǻŁłǻǺǻǻAłJæ■cǻŠŇéĜ■ç;õçŽǺèõǻǻ  
 ǻõČäijŽǻłJł elapsed ǻśđæǺğäy■èõřǻ;ŦæŦř'äyłæúŁèǺŮæŮúéŮř'ǻǺĆ  
 äyŇéłćæŸřäyÄäyłǻ;Ňǻ■RǻłæijŦčd'žæǺŬæǻüǻ;ƒçŦłǻõČüjŽ

```

def countdown(n):
    while n > 0:
        n -= 1

# Use 1: Explicit start/stop
t = Timer()
t.start()
countdown(1000000)
t.stop()
print(t.elapsed)

# Use 2: As a context manager
with t:
    countdown(1000000)

print(t.elapsed)

with Timer() as t2:
    countdown(1000000)
print(t2.elapsed)

```

## èõłèõž

æIJñèŁĆæRŘǻ;ŽázEäyÄäyłçóǺǻŦèǺŇǻõđçŦłçŽǺđçsžæłæǻõđçŎřæŮúéŮř'èõřǻ;ŦǻžèǻRŁèǺŮæŮúéõǻǻ  
 ǻŖŇæŮúǻžšæŸřǻřǻžǻ;ƒçŦłwithèř■ǻRèǻžèǻRŁäyŁäyŇæŮĜçõǻçŖĒǻŽłǻ■RèõõçŽǺäyÄäyłǻ;Łǻè;çŽǺäijŦčd'ž  
 ǻłJłèõǻæŮúäy■èçAèǺČèŽŚäyÄäyłǻžŦǻśĆçŽǺæŮúéŮř'ǻĜ;æŦřèŮóéçŸǻǺĆäyǺèŁŇæłèèř'ijŇ

ä;£çTítime.time() æLÚtime.clock() èóaçõÚçŽĐæUúéÚt' çš;ážæŽæŞ■ä;IJçşççzşçŽĐäy■ăRŃăij.  
èĂŃă;£çTítime.perf\_counter() âĜ;æTřăRřăžěçăôăĤă;£çTíçşççzşăyLéÍcæIJăçş;çăôçŽĐěóæUúăŽ.  
ăyLêřăžčăĂăy■çTśTimer çşžěőřă;TçŽĐæUúéÚt' æÝřéŠşěăĤæUúéÚt' ĩijŇăžúăŇĚăRňăžĚæL' ĂæIJL'ă  
ăĉCăđIJă;ăăRăcŞşěóaçõUèřěčŽçÍŇæL' ĂèLşet' zçŽĐCPUæUúéÚt' ĩijŇăžTěřă;£çTí time.  
process\_time() æĹăžčæŽĤĭjŽ

```
t = Timer(time.process_time)
with t:
    countdown(1000000)
print(t.elapsed)
```

time.perf\_counter() âŠŇ time.process\_time()  
éČ;ăijŽěĤăŽďăřRăTřă;ćăijRçŽĐçğŞæTřæUúéÚt' āĂĆ âóđéŽĚçŽĐæUúéÚt' āĂijæşăæIJL'ăžză;TăĐRăžL'ĭij.  
æŽt'ăđ' ŽăĚşăžŎěóæUúăŠŇæĂğěČ;ăĤĚăđRçŽĐă;Ňă■RěřăăRĆěĂĆ14.13ăřRêĤCăĂĆ

## 15.14 13.14 éŽŘăĹúăĚĚă■ŸăŠŇCPUçŽĐă;£çTíéĜŘ

### éUóécŸ

ă;ăæČşăřzăIJĬUnixçşççzşăyLéÍcěĤRěăŇçŽĐçĹŇăžRěô;ç;őăĚĚă■ŸæLÚCPUçŽĐă;£çTíéŽŘăĹúăĂĆ

### èğcăĚşæÚzæăĹ

resource əĤăăĬUèČ;ăRŇæUúæL'ğěăŇěĤăyđ'ăyĤăžzăăĤăăĂĆă;ŇăçĆĭijŇěçĂéŽŘăĹúCPUæUúéÚt' ĩij.

```
import signal
import resource
import os

def time_exceeded(signo, frame):
    print("Time's up!")
    raise SystemExit(1)

def set_max_runtime(seconds):
    # Install the signal handler and set a resource limit
    soft, hard = resource.getrlimit(resource.RLIMIT_CPU)
    resource.setrlimit(resource.RLIMIT_CPU, (seconds, hard))
    signal.signal(signal.SIGXCPU, time_exceeded)

if __name__ == '__main__':
    set_max_runtime(15)
    while True:
        pass
```

çÍŇăžRěĤRěăŇæUúĭijŇSIGXCPUăĤăăRúăIJăUúéÚt' ěĤĜæIJşæUúěćŇçTşæĹŘĭijŇçĐúăŘŎæL'ğěăŇăy.  
ěçĂéŽŘăĹúăĚĚă■Ÿă;£çTíĭijŇěô;ç;őăRřă;£çTíçŽĐăĂžăĚĚă■ŸăĂĭjă■şăRřĭijŇăçCăyŇĭjŽ



ǎĈRèfZæăũëö;ç;őăzĖăĖĖă■ŸéZŔăĹăăŔŎiĵŊċĹăŇăzŔèĹŔèąŇăĹŕăşşæIJĹ'ăđ'ŽăĵZăĖĖă■ŸæŮüăĵZăĹZ  
MemoryError ħĵĈăŷŷăĂĈ

āIɪæIɪnēŁĆä;Ŋā■Räy■iijŊsetrlimit() āĜ;æṬrēcñĆṬlæIēēō;ç;ōĆL'zāōZēĬDæzRäyŁēIēĆçŽDē;réŽR  
 ē;réŽRĀLūāỴrāyĀỵlāĀij̣iijŊā;ŞēŪEēĜĜēZāỵlāĀij̣çŽDæŪŪāĀZāŞ■ā;IjçşçzçşēĀZāỵyāij̣ZāRŚēĀĀỵĀỵl  
 çāñēŽRĀLūāỴrçṬlæIēāŊĜāōŽē;réŽRĀLūēĆ;ēō;āōZçŽDæIĀĀd'gāĀijaĀĆēĀZāỵyæIēēōşiijŊēŁZāỵlçṬşçşz  
 ār;çōaçāñēŽRĀLūāR̄rāzēāT̄zārRäyĀçĆçiiijŊā;EæỴræIĀāē;äy■ēēĀä;ŁçṬlçṬlæLūēŁZçIŊāŌzāŁōāT̄zāĀĆ

éIĲàèçAæşlæĐŔčŽĐæÝřæIJñèŁĆăĚĖăőžâRİēČ;éĂĆčŤlázÕUnixçşzçzşİijÑāẏüäȚă■ăļērAæL'ĂæIJL  
ærŦăęCăĹSăżnăIJłęJNërŦçŻĐæUũăĂŻīijÑăôÇcëČ;ăIJlLinuxăÿŁéĬcă■čăÿÿēŁŘěąŊīijNă;ĘæÝřăIJlOS  
XăÿŁă■t'ăÿ■ēČ;ăĂĆ

ä:äăČšéĂžēfĜēĎŽæIĴnăRřăLíæŧRèĝLăŽlăzúæL'SaijĂæŇĜăoŽčŽĎURLç:Šéat

webbrowser ælaaiUèĈěcňŤlæleăRrăLlăyĂăvylætRèğLăZlîijŇăzúăyŤăyŎăzşăRrăUăăEşăĂĈă;ŇăeĈă

ǎŏČäijZǎ;ŁčTlězYēōd'ætRègŁǎZlæL'SajjĀxNǧǎoŽc;ŚeātāĀČaeĆæđIjä;æfYæČśaržc;ŚeātæL'SajjĀxU

```
>>> # Open the page in a new browser window
>>> webbrowser.open_new('http://www.python.org')
True
>>>

>>> # Open the page in a new browser tab
>>> webbrowser.open_new_tab('http://www.python.org')
```

```
True
>>>
```

èfZæûâršâRřäzèæL'SâijÄäyÄäylæŮřçŽDætRègŁăZlçłŮârçæŁŮèĂĚæăĠç■iijNâRlèçAætRègŁăZlæT  
âçCæđIJä;ăæČšæŇĠăŏŽætRègŁăZlçšzăđNriijNâRřäzèä;ŁçTl webbrower.get()  
âĠ;æTřælææŇĠăŏŽæšRäyłçL'zâŏŽætRègŁăZlăĂCă;NâçCrijŽ

```
>>> c = webbrowser.get('firefox')
>>> c.open('http://www.python.org')
True
>>> c.open_new_tab('http://docs.python.org')
True
>>>
```

ârzäžŎæTřæŇAçŽDætRègŁăZlăR■çğrăLŮèălăRřæšéĚ'PythonæŮĠæaç <<http://docs.python.org/3/library/webbrowser.html>> '\_

## èŏlèŏž

âIJlèDŽæIJñäy■æL'SâijAætRègŁăZlæIJL'æŮŮăĂZâijŽă;ŁæIJL'çTlăĂCă;NâçCrijNæšRäyłèDŽæIJñæL  
ä;ăæČšăĤnéĂšæL'SâijÄäyÄäylæTŘègŁăZlælèçăŏăĤlăŏČăŮšçzRæ■čäyÿèĤRèăNăžĚăĂČ  
æLŮèĂĚæYřæšRäyłçłNăžRăžèHTMLç;ŠéatæâijâijRè;ŠăĠžæTřæ■ŏriijNă;ăæČšæL'SâijAætRègŁăZlæšççIJN  
äy■çŏăæYřäyŁélçăŠłçğ■æČĚăĤriijNă;ŁçTl webbrower ælăălŮéČ;æYřäyÄäyłçŏĂă■TăŏđçTlçŽDèğčăĤşă

## 16 çñňă■AăŽŽçnáïijŽætNèrTăĂAèrCèrTăŠNâijCăyÿ

èrTlèlNèĤYæYřă;ŁæçŠçŽDriijNă;ĤæYřèrCèrTrijšâršæšăçĆčăzŁæIJL'èŮčăžĤăĂČăžNăŏđæYřriijNăIJlPytl  
Contents:

### 16.1 14.1 æTŇèrTstdoutè;ŠăĠž

#### éŮŏéçY

ă;ăçŽDçłNăžRäy■æIJL'äyłæŮzæšTăijŽè;ŠăĠžăLřæăĠăĠĤè;ŠăĠžăy■riijLsys.stdoutriijL'ăĂČăžšâršæYřè  
ă;ăæČšăĤZăyłætŇèrTălèèrAæYŎăŏČriijŇçzŽăŏŽăyÄäyłè;ŠăĤëriijŇçZyăžTçŽDè;ŠăĠžèČ;æ■čäyÿæYłçđ'ză

#### èğčăĤşăŮzæăł

ă;ŁçTl unittest.mock ælăălŮäy■çŽD patch() âĠ;æTrijŇ  
ă;ŁçTlèŮălèélđăyÿçŏĂă■TrijNâRřäzèäyžă■TăyłætŇèrTălăæNš sys.stdout  
çDŮăRŎăZdæzZriijN âžŮäyTăy■ăžğçTšăđ'gèĠRçŽDăyt'æŮŮăRŮéĠRæLŮăIJlætŇèrTçTlă;ŇçŽt'æŎèæŽt'èIJ

ăIJăyžăyÄäyłă;Nă■RriijNæLŠăžňăIJl mymodule ælăălŮäy■ăŏžăzL'ăçCăyNăyÄäyłăĠ;æTrijŽ

```
# mymodule.py
```

```
def urlprint(protocol, host, domain):  
    url = '{}://{}.{}'.format(protocol, host, domain)  
    print(url)
```

```
    ézYëød'æČĚâEĕÿNâĚĚç;ôçŽĎ print ăĜ;æTŗăijŽăřĚë;ŞăĜzăŔSéĂĀăĹŕ sys.  
stdout ăĂĆ äyžăžĚæĭNërTĕ;ŞăĜžçIJşçŽĎăIJĹéĆĉéĜNĭijNă;ăăŔřăžěă;ĚçTĭăyĂăyĭæŽĚěžnăřzèşăĹăĹăæN  
ă;ĚçTĭ unittest.mock ăĹăăĭŬçŽĎ patch() æŬzæşTăŔřăžěă;ĹæŬză;ĚçŽĎăIJĹăĭNërTĕĚŔăqNçŽĎăyĹ  
ăžŭăyTă;ŞăĭNërTăôNăĹŔăŬăĂŽëĜĭăĹĹëĚTăŽĎăôČăžñçŽĎăŎşæIJĹçĹŭăĂĀăĂĆăyNéĭĉæYŕăřž  
mymodule ăĹăăĭŬçŽĎăĭNërTăžçăĂĭijŽ
```

```
from io import StringIO  
from unittest import TestCase  
from unittest.mock import patch  
import mymodule  
  
class TestURLPrint(TestCase):  
    def test_url_gets_to_stdout(self):  
        protocol = 'http'  
        host = 'www'  
        domain = 'example.com'  
        expected_url = '{}://{}.{}\n'.format(protocol, host, domain)  
  
        with patch('sys.stdout', new=StringIO()) as fake_out:  
            mymodule.urlprint(protocol, host, domain)  
            self.assertEqual(fake_out.getvalue(), expected_url)
```

## ëőĹëőž

```
urlprint() ăĜ;æTŗăŎĉăŔŬăyĹăyĭăŔĆæTŗĭijNăĭNërTăŬzæşTăĭjĂăĝNăĭjŽăĚĹëőç;ôăŕŔăyĂăyĭăĹă  
expected_url ăŔYĕĜŔăĚĉñëőç;ôăĹŔăŇĚăŔŕăăIJşæIJŽçŽĎë;ŞăĜžçŽĎăŬçĥăyşăĂĆ
```

```
unittest.mock.patch() ăĜ;æTŗĕćñçTĭă;IJăyĂăyĭăyĹăyNăŬĜçôăçŔĚăŽĭijNă;ĚçTĭ  
StringIO ăŕžèşăĹăĹăžçăŽĚ sys.stdout fake_out  
ăŔYĕĜŔăYŕăĹIJĹĕŕĕĚĚçĹNăy■ĕćnăĹŽăžççŽĎăĹăæNşăŕžèşăăĂĆ ăIJĭwith-  
ĕŕ■ăŔĕăy■ă;ĚçTĭăôČăŔřăžěăĹġĕăNăŔĎçġ■ăĉĂăşĕăĂĆă;Şwithĕŕ■ăŔĕççŞăĹşăŬĭijNpatch  
ăĭjŽăřĚăĹĂăĹĹăyĹĹĕĚăĂĉăđ■ăĹŕăĭNërTăĭjĂăĝNăĹ■çŽĎçĹŭăĂĀăĂĆ  
ăĹĹăyĂçĆzéIJĂĕĕĂăşĭăĎŔçŽĎăYŕăşŔăžŽăŕžPythonçŽĎCăĹĹăşTăŔŕĕČ;ăĭjŽăĚ;çTĕăŎĹ  
sys.stdout çŽĎĚĚ■ç;ôăžNçŽĹăŎĉăĚŽăĚĕăĹŕăăĜăĜĚë;ŞăĜzăy■ăĂĆ  
éŽŔăžŎçŕĜăžĔĭijNăIJĹĕĹăy■ăĭjŽăŭĹăŔĹăĹŕĕĚăŬzéĹççŽĎĕşĕġĭijNăôČăĂĆçTĭăžŎçŕPythonăžçăĂă  
ăĕČăđIJă;ăçIJşçŽĎĚIJĂĕĕĂăĹĹCăĹĹăşTăy■ă■TĕŎŭĹ/OĭijNă;ăăŔřăžěăĚĹăĹŞăĭjĂăyĂăyĭăyŕăŬăŬĜăžŭ  
ăŽĹăđŽăĚşăžŎă■TĕŎŭăžĕă■Ŭçĥăyşă;ăăĭjŔă■TĕŎŭĹ/OăŞN StringIO  
ăŕžèşăĕŕŭăŔĆĕYĔ5.6ăŕŔĕĹCăĂĆ
```

## 16.2 14.2 aIJaTãĖCætNërTäy■czZärzèsæL'SèaëäyA

### éUóécY

ä;ääEŽçŽDã■TãĖCætNërTäy■éIJÄèeAçzZæŃGăőŽçŽDärzèsæL'SèaëäyArijŃ  
çTlãlëæŮ■élĀăőČăznãIJlætNërTäy■çŽDæIJšæIJZëaŃäyziijLærTæCrijNæŮ■élĀècnerČçTlãŮüçŽDãRCæt

### èğcãEşæŮzæaL

unittest.mock.patch() aĜ;æTřãRřecncTlãlëèğcãEşæŁZäyIéŮóécYãĀC  
patch() èŁYãRřecncTlã;IJäyÄäyIëčĚéčřãZlãĀÄyŁäyNæŮĜçõaçŘEãZlãLŮã■TçNňã;ŁçTliijNär;çõaázŮ  
ä;ŃäçCrijNäyNéIcæYřäyÄäyIärEăőČă;ŠãAŽècĚéčřãZlã;ŁçTlçŽDä;Ńã■RijŽ

```
from unittest.mock import patch
import example

@patch('example.func')
def test1(x, mock_func):
    example.func(x)          # Uses patched example.func
    mock_func.assert_called_with(x)
```

ăőČèŁYãRřäzèècnã;ŠãAŽäyÄäyŁäyNæŮĜçõaçŘEãZliijŽ

```
with patch('example.func') as mock_func:
    example.func(x)          # Uses patched example.func
    mock_func.assert_called_with(x)
```

æIJÄãRŮrijNä;äèŁYãRřäzèæL'ŃãLlçŽDä;ŁçTlăőCæL'SèaëäyArijŽ

```
p = patch('example.func')
mock_func = p.start()
example.func(x)
mock_func.assert_called_with(x)
p.stop()
```

ăçCădIJãRřèČ;çŽDëriijNä;äèČ;ăd'šãRăăLăèčĚéčřãZlãŃäyŁäyNæŮĜçõaçŘEãZlãlëççZăd'ŽäyIärzès

```
@patch('example.func1')
@patch('example.func2')
@patch('example.func3')
def test1(mock1, mock2, mock3):
    ...

def test2():
    with patch('example.patch1') as mock1, \
        patch('example.patch2') as mock2, \
        patch('example.patch3') as mock3:
        ...
```

## ěőléőž

patch() æŔŕăŕŭăŷĂăŷłăŭšă■ŸăĬłŕžèšăçŽĎăĚłêŭŕăĬăŔ■ĭĭŷŇăŕĚăĚŭăŽĚă■căŷžăŷĂăŷłăŰŕçŽĎă  
ăŎšăĬêçŽĎăĀĭĭăĭŷŽăĬĬêçĚéēŕăŽĬăĜĭæŦŕăĬŰăŷĹăŷŇăŰĜçŏăçŔĚăŽĬăŏŇăĬŔăŔŎêĜĬăĹăĹăĀčăđ■ăŽđăĬěă  
ézŸêŏđ'ăĈĚăĚŷŷŇĭĭŷŇăĹ'ĂăĬĬăĀĭĭăĭŷŽéčŇ MagicMock áŏđăĬŇăŽĚăžčăĂĈăĬŇăĚĈĭĭŷŽ

```
>>> x = 42
>>> with patch('__main__.x'):
...     print(x)
...
<MagicMock name='x' id='4314230032'>
>>> x
42
>>>
```

ăŷ■êĚĜĭĭŷŇăĭăŕŕăžèéĂŽĚĚĜçžŽ patch() æŔŔăĬŽçŇăžŇăŷłăŔĈăŦŕăĬêŕĚăĀĭĭăŽĚă■căĹŔăžžăĭŦ

```
>>> x
42
>>> with patch('__main__.x', 'patched_value'):
...     print(x)
...
patched_value
>>> x
42
>>>
```

èçŇçĹĬăĬěăĭĬăŷžăŽĚă■căĀĭĭçŽĎ MagicMock áŏđăĬŇêĈĭăđ'šăĹăăŇšăŕŕêŕĈçĹĬăŕžèšăăŖŇăŏđăĬŇăĂ  
ăžŰăžŇêŕŕăĭŦăŕžèšăçŽĎăĭçĹĬăĤăăĀŕăžŭăĚăĚŏŷăĭăăĹġêăŇăŰ■êĹĂăĈĂăšĕĭĭŷŇăĬŇăĚĈĭĭŷŽ

```
>>> from unittest.mock import MagicMock
>>> m = MagicMock(return_value = 10)
>>> m(1, 2, debug=True)
10
>>> m.assert_called_with(1, 2, debug=True)
>>> m.assert_called_with(1, 2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File ".../unittest/mock.py", line 726, in assert_called_with
    raise AssertionError(msg)
AssertionError: Expected call: mock(1, 2)
Actual call: mock(1, 2, debug=True)
>>>

>>> m.upper.return_value = 'HELLO'
>>> m.upper('hello')
'HELLO'
>>> assert m.upper.called

>>> m.split.return_value = ['hello', 'world']
>>> m.split('hello world')
```

```

['hello', 'world']
>>> m.split.assert_called_with('hello world')
>>>

>>> m['blah']
<MagicMock name='mock.__getitem__()' id='4314412048'>
>>> m.__getitem__.called
True
>>> m.__getitem__.assert_called_with('blah')
>>>

```

äyÄeLñæIëèöšijÑeŁŻäzZæŞ■ä;IJäijŽäIJläyÄäyIa■TäĖČætNërTäy■ăōÑæLRăĂĆă;NăeĆijNăAĞëö;ä;

```

# example.py
from urllib.request import urlopen
import csv

def dowprices():
    u = urlopen('http://finance.yahoo.com/d/quotes.csv?s=@^DJI&f=s11
↳ ')
    lines = (line.decode('utf-8') for line in u)
    rows = (row for row in csv.reader(lines) if len(row) == 2)
    prices = { name:float(price) for name, price in rows }
    return prices

```

æ■čäyÿæIëèöšijÑeŁŻäyIaĞ;æTřäijŽä;ŁçTl urlopen() äzŎWe-  
bäyŁéIcèŎuăRŮæTřæ■ăōăzŭëğçædŘăôČăĂĆ äIJIa■TäĖČætNërTäy■ijNă;ăăRřäzëçzŽăôČäyÄäyIécĎăĖLăôŽ

```

import unittest
from unittest.mock import patch
import io
import example

sample_data = io.BytesIO(b'''\
"IBM",91.1\r
"AA",13.25\r
"MSFT",27.72\r
\r
''')

class Tests(unittest.TestCase):
    @patch('example.urlopen', return_value=sample_data)
    def test_dowprices(self, mock_urlopen):
        p = example.dowprices()
        self.assertTrue(mock_urlopen.called)
        self.assertEqual(p,
                          {'IBM': 91.1,
                           'AA': 13.25,
                           'MSFT' : 27.72})

if __name__ == '__main__':

```

```
unittest.main()
```

æIñä;Ñäy■iijÑä;■äzÖ example æÍaálUäy■çŽĐ urlopen()  
 åĜ;æTřècñäyÄäyÍæÍaæNşáržèsææŽfäzçiiĴ ēřæáržèsäiĵŽefTāŽdäyÄäyÍaÑĚäRñætĴNērTæTřæ■ócŽĐ  
 ByteIO().

æŸæIJLäyÄçĆzĳĳŃâIJlæLŠèæëyAæŮüæĹŚázñä;£çŦlăẼE example.  
 urlopen æĹëăžçæẼ£ urllib.request.urlopen äĂĆ  
 âĴšă;ăăĹLZăžžèæëyAçŽDæŮüăĂŽĳĳŃă;ăăŦĒéæžă;£çŦlăŮĆăžñăIJlăĳŦĒërŦăžčçăAăy■çŽDăŦ■çğŕăĂĆ  
 çŦśăžŮæŦŦĒërŦăžčçăAă;£çŦlăẼE from urllib.request import urlopen ,éĆčăžĹ  
 dowprices() âĴ;æŦŦ äy■ă;£çŦlçŽD urlopen() âĴ;æŦŦăŮđéŽĒyŦĹŕśă;■ăžŮ  
 example æĹăăĹŮăžĒĂĆ

æIJñèŁĆăōđēZĚäyŁăŔlæYřărz unittest.mock ælqaiUčZDäyĂæñætĚăřlë;Đæ■ćăĂĆ  
æZťăđ'ŽæZťénYčžgčZĐčL'zæĂgĭijNěřůăŔĆěĂĆ ăőYăŮzăŮĞăăç

16.3 14.3 aJá■TāĖĈætNërTäy■ætNërTāiĈaÿÿæĈĖaĖt

éŮőécÿ

ä:äăČšăĖŽăyłætNërTçTlĭă;NăĭăăĠĖçăoçŽĎăĹđ'æŨ■æšŘăyłaijCăyŷăYrăŘeēcăăĹŽăĠzăĂĆ

èġčǎẸșæŮźæąŁ

árżăžŎaijĆăyŷçŽDætŊerŤăŔŕăĭġçŦĭ      assertRaises()      æŰzæşŤăĂĆ  
 äĭŊăeĈiijŊăeĈădĬJăĭăeĈşætŊerŤăşŔăyĭăĜĭ;æŤŕăĽZăĜzăĖ  
 aijĆăyŷiijŊăĈŔăyŊeĭçèĖŽăuăăĖZiijŽ      ValueError

```
import unittest

# A simple function to illustrate
def parse_int(s):
    return int(s)

class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        self.assertRaises(ValueError, parse_int, 'N/A')
```

æCædIJa:æCşætNerTajCâyçZĐaĖuä;ŞaAiijijNéIJAèeAçTlälRäRead'ŮäyÄçg■ŮzæşTijZ

```
import errno

class TestIO(unittest.TestCase):
    def test_file_not_found(self):
        try:
            f = open('/file/not/found')
        except IOError as e:
            self.assertEqual(e.errno, errno.ENOENT)
```

```
else:
    self.fail('IOError not raised')
```

## ěóľěőž

`assertRaises()` æŮžæšŤäÿžæŧŇërŤäijČäÿÿâ■ŸâIJlæĀğæŔŔä;ŽäžEäÿÄäÿłçōĀä;ŁæŮžæšŤāĀĆäÿÄäÿłäÿÿëğAçŽĎéŽüéŸsæŸŕæLŇāLlāŌžèŁZèqŇäijČäÿÿæčĀæŧŇāĀĆæŕŤæČiijŽ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        try:
            r = parse_int('N/A')
        except ValueError as e:
            self.assertEqual(type(e), ValueError)
```

èŁŽçğ■æŮžæšŤçŽĎeŮöécŸâIJlāžŌāōČä;ŁāōžæŸŦéAŮæijŔāĚüāžŮæČĚāĚiijŇæŕŤæČæšæIJL'äzzä;éČčāžŁä;äèŸä;ŮéIJĀèçAāčđāŁāāŔēāđ'ŮçŽĎæčĀæŧŇèŁĞčlŇiijŇāçČäÿŇéłçèŁŽæäüijŽ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        try:
            r = parse_int('N/A')
        except ValueError as e:
            self.assertEqual(type(e), ValueError)
        else:
            self.fail('ValueError not raised')
```

`assertRaises()` æŮžæšŤäijŽād'ĎçŔĚæL'ĀæIJL'çžĚŁČiijŇāŽāæ■d'ä;āāžŤērēä;ŁçŤlāōČāĀĆ

`assertRaises()` çŽĎäÿÄäÿłçijžçČzæŸŕāōČæŧŇäÿ■āžĚäijČäÿÿāĚüā;ŦçŽĎāĀijæŸŕād'ŽārSāĀĆäÿžāžĚæŧŇërŤäijČäÿÿāĀijijŇāŔŕāžēä;ŁçŤl

`assertRaisesRegex()` æŮžæšŤiijŇāōČāŔŕāŔŇæŮüæŧŇërŤäijČäÿÿçŽĎā■ŸâIJlāžēāŔĚéĀŽèŁĞæ■čāŁŽäijŔāŇzéĚ■äijČäÿÿçŽĎā■Ůçñäÿšēāłçđ

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        self.assertRaisesRegex(ValueError, 'invalid literal .*',
                               parse_int, 'N/A')
```

`assertRaises()` āŠŇ `assertRaisesRegex()`

èŁŸæIJL'äÿÄäÿłāōžæŸŦsæŁçŤçŽĎāIJŕæŮžāršæŸŕāōČāžñèŁŸèČ;èçŇā;ŦāAžäÿŁäÿŇæŮĞçōaçŔĚāŽlā;ŁçŤl

```
class TestConversion(unittest.TestCase):
    def test_bad_int(self):
        with self.assertRaisesRegex(ValueError, 'invalid literal .*
→'):
            r = parse_int('N/A')
```

ä;Ěä;āçŽĎæŧŇërŤæŮL'āŔĚāŁŕād'ŽäÿlæL'ğēāŇæ■čēłđ'çŽĎæŮüāĀŽèŁŽçğ■æŮžæšŤāršä;ŁæIJL'çŤlāžĚ





```
import unittest
import os
import platform

class Tests(unittest.TestCase):
    def test_0(self):
        self.assertTrue(True)

    @unittest.skip('skipped test')
    def test_1(self):
        self.fail('should have failed!')

    @unittest.skipIf(os.name=='posix', 'Not supported on Unix')
    def test_2(self):
        import winreg

    @unittest.skipUnless(platform.system() == 'Darwin', 'Mac_
→specific test')
    def test_3(self):
        self.assertTrue(True)

    @unittest.expectedFailure
    def test_4(self):
        self.assertEqual(2+2, 5)

if __name__ == '__main__':
    unittest.main()
```

æĊædIJä;ääIJÍMacäyŁèĤŘèąNèĤŽæőłäzččăAĭijNă;ăăijŽă; ŪăĤŕăĊăyNè;ŠăĠzĭijŽ

```
bash % python3 testsample.py -v
test_0 (__main__.Tests) ... ok
test_1 (__main__.Tests) ... skipped 'skipped test'
test_2 (__main__.Tests) ... skipped 'Not supported on Unix'
test_3 (__main__.Tests) ... ok
test_4 (__main__.Tests) ... expected failure

-----
↪--
Ran 5 tests in 0.002s

OK (skipped=2, expected failures=1)
```

## ěőłěőž

skip() ěĊĚěřăŽłĊ;ěċŋĤłăĬăĤ;ĤĤă\$Řăyłă;ăăy■ăĤšěĤŘèąNċŽĎăĤNĕŕĤăĂĊ  
skipIf() ăŠŇ skipUnless() ăŕžăžŌă;ăăŔăĤĤšăIJă\$ŘăyłĊŁ'žăőŽăžšăŔŕăĤŪPythonĊŁ'ĤăIJăĤŪăĚŪ  
ă;ĤĊĤĬ@expectedĊŽĎăĎ'sĕt'ěċĊĚěřăŽłăĬăăĠĕőŕĕĊăžŽĊăőăőŽăijŽăĎ'sĕt'ěĊŽĎăĤNĕŕĤĭijNăžŭăyĤăŕžĕĤ  
ăĤ;ĊĤĤăŪžăšĤĊŽĎěĊĚěřăŽłĤŸăŔŕăžěěċŋĤłăĬăěěĊĚěĕŕăĤŕăyĤăĤNĕŕĤĊšzĭijNăŕĤăĊĭijŽ

```
@unittest.skipUnless(platform.system() == 'Darwin', 'Mac specific_
↪tests')
class DarwinTests(unittest.TestCase):
    pass
```

## 16.6 14.6 ăĎ'ĎĊŘĖăĎ'ŽăyłăijĊăyŷ

### éŪőěĊŸ

ă;ăăIJŁ'ăyĂăyłăžċčăAĊŁ'ĠăőłăŔŕĕĊ;ăijŽăŁŽăĠžăĎ'Žăyłăy■ăŔNċŽĎăijĊăyŷĭijNăĂŌăăŭăĤ'ěĊ;ăy■

## ěġĊăĖšăŪžăăĤ

æĊædIJä;ăăŔŕăžěĊĤłă■ĤăyłăžċčăAăĬŪăĎ'ĎĊŘĖăy■ăŔNċŽĎăijĊăyŷĭijNăŔŕăžěăŕĖăăőĊăžăĤăĤăĚăyĂă

```
try:
    client_obj.get_url(url)
except (URLError, ValueError, SocketTimeout):
    client_obj.remove_url(url)
```

ajllefZayla.Na.Ray.NijNaECceUay.azza;TayAaylajCayyaRSCTsaUueCijZaeL'geaN  
remove\_url() æUæsaTaaC æCædIJa;æCsaazæUay.æsaRaylajCayyeLZeaNay.aRNcZDad'DcREijNa  
except er.aReay.NijZ

```
try:
    client_obj.get_url(url)
except (URLError, ValueError):
    client_obj.remove_url(url)
except SocketTimeout:
    client_obj.handle_url_timeout(url)
```

alLad'ZcZDajCayyajZaeIJL'asCcZgaEsçszijNarzažOefZcg.æCEaEijNa;aaRfeC;ajLçTiaõCazncZDa

```
try:
    f = open(filename)
except (FileNotFoundError, PermissionError):
    pass
```

aRrazecneG.aEZayzijZ

```
try:
    f = open(filename)
except OSError:
    pass
```

OSError æYr FileNotFoundError aŠN PermissionError  
ajCayycZDaşçszaaC

## eoieoz

ar;çoaad'DcREad'ZaylajCayyaIJneznazuæsaazAazLçL'zaolçZDijNay.æLGa;aaRfazea;LçTl  
as aEsçTaa.UæleeOua.UecnaLZaGzajCayycZDajTçTlijZ

```
try:
    f = open(filename)
except OSError as e:
    if e.errno == errno.ENOENT:
        logger.error('File not found')
    elif e.errno == errno.EACCES:
        logger.error('Permission denied')
    else:
        logger.error('Unexpected error: %d', e.errno)
```

efZayla.Na.Ray.NijN e aRYeGRæNGaRSayAaylecnæLZaGzcZD OSError  
ajCayyaoda.NaAC efZaylaIJJa;æCsaZt'efZayAæ.eaLEadReLZaylajCayycZDæUuaAZajZa;LaeIJL'çTlij

aRNæUueL'YeAæslæDRçZDæUuaAZ except er.aReæYreažazRæcAæseçZDijNçnnayAaylaNzeE.  
ajaaRfazea;LaõzaYşçZDædDeAaad'Zayl except aRNæUuaNzeE.çZDæCEa;ciijNærTæCijZ

```
>>> f = open('missing')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'missing'
>>> try:
...     f = open('missing')
... except OSError:
...     print('It failed')
... except FileNotFoundError:
...     print('File not found')
...
It failed
>>>
```

FileNotFoundError er ðáRæáúæšæIJL æL'gæaŃçŽDăŎšăZăæYř  
 OSError æŽt'äyÄeLñijŃăŎČăRřăŃzéĚ FileNotFoundEror äijCăyÿijŃ  
 äžŎæYřăřsæYřçññäyÄäyŃăŃzéĚ ŽDăĂĆ äIJlërČërŤçŽDæŰuăĂŽijŃăæČăđIJă;ăărzæšRăyŃçL'zăŏŽăijCăyÿ  
 ä;ăăRřăžéĂŽeĚGăšççIJŃerëäijCăyÿçŽD \_\_mro\_\_ äšdæĂgæŃeăŃnéĂšætŤRègĹăĂĆærŤæČijŽ

```
>>> FileNotFoundError.__mro__
(<class 'FileNotFoundError'>, <class 'OSError'>, <class 'Exception'>
↳,
 <class 'BaseException'>, <class 'object'>)
>>>
```

äyLéŃcálŰeăŃäyăăzză;ŤäyĂäyŃçŽt'ăĹř BaseException çŽDçşzéČ;èČ;èçŃçŤlăžŎ  
 except er ðáRæăĂĆ

## 16.7 14.7 æŃŤèŎuæL'ĂæIJL'äijCăyÿ

### éŰŏécŸ

æĂŎæăuæŃŤèŎuăžçăĂäyŃçŽDæL'ĂæIJL'äijCăyÿijš

### èğčăEşæŰzæăĹ

æČşèçĂæŃŤèŎuæL'ĂæIJL'çŽDăijCăyÿijŃăRřăžéçŽt' æŎčæŃŤèŎu Exception  
 äŃşăRřijŽ

```
try:
...
except Exception as e:
...
    log('Reason:', e)           # Important!
```

èĚŽăyŃlărĚäijŽæŃŤèŎuėŽd'ăžĚ SystemExit äĂĂ KeyboardInterrupt  
 äšŃŃ GeneratorExit äžŃăd'ŰçŽDæL'ĂæIJL'äijCăyÿăĂĆ

æĈæđIĴajæĤŸæĈſæ■TèŌüèĤŽäyL'äyĴajĈäyŷiijNärE  
BaseException ā■şâRřăĂĆ

Exception

æŤžæĹŘ

## èóíèőž

æ■TèŌüæL'ĂæIJL'āijĈäyŷeĂŽäyŷæŸřĉŤśäžŌĉÍNăžRăŚŸăIJĴæſŘăžŽăđ'■æĬCæŞ■ăĴIĴă■ăžüăy■èĈĵèő  
æĈæđIĴajăăy■æŸřăĴĴçEăĤĈçŽĐăžžiiijNèĤŽăžſæŸřĉijŪăEŽăy■æŸſĕřĈĕřŤăžĉĉăAĉŽĐăyĂăyĴĉôĂă■ŤæŪ

æ■ĉăŽăăĈæĈ■d'iiijNăĈæđIĴajăéĂL'æŅŦ'æ■TèŌüæL'ĂæIJL'āijĈäyŷiijNéĈĉăžĴăIJĴæſŘăyĴăIJræŪžiiijLă  
æĈæđIĴajăæſqæIJL'èĤŽăăüăĂžiiijNæIJL'æŪăăĂŽăĴăĴIJNăĴrăijĈäyŷæL'Şă■ræŪăăRřèĈĵæŸyăy■ĉĬĂăđ't'èĐ

```
def parse_int(s):  
    try:  
        n = int(v)  
    except Exception:  
        print("Couldn't parse")
```

èřŤĉĬĂèĤRëqNèĤŽăyĴăĴĴ;æŤřiiijNĉžŞăđIĴæĈäyNiiijŽ

```
>>> parse_int('n/a')  
Couldn't parse  
>>> parse_int('42')  
Couldn't parse  
>>>
```

èĤŽæŪăăĂŽăĴăăřſăijŽæNăăđ't'æĈſiiijŽăĂIJèĤŽăŚNăŽđăžNăŤĴiiijſăĂĬ  
ăĂĴăĈăĴăăĈRăyNéĬĉèĤŽăăüèĈ■ăEŽèĤŽăyĴăĴĴ;æŤřiiijŽ

```
def parse_int(s):  
    try:  
        n = int(v)  
    except Exception as e:  
        print("Couldn't parse")  
        print('Reason:', e)
```

èĤŽæŪăăĂŽăĴăăèĈĵèŌüăRŪăĈäyNèĴŞăĴžiiijNæNĜæŸŌăžEæIJL'äyĴĉijŪĉĬNéŤŽérriijŽ

```
>>> parse_int('42')  
Couldn't parse  
Reason: global name 'v' is not defined  
>>>
```

ăĴĴæŸŌæŸĴiiijNăăăžŤĕřăřĴăRřèĈĵăřEăijĈäyŷăđ'ĐĉŘEăŽĴăôŽăžL'ĉŽĐĉſĴăĴEăyĂăžŽăĂĆ  
ăy■èĤĴiiijNèĈAæŸřăĴăăĤĤéăžæ■TèŌüæL'ĂæIJL'āijĈäyŷiijNĉăôăĤĴăL'Şă■ræ■ĉĉăôĉŽĐĕřĴăŪ■ăĤăæAřăĴŪă

## 16.8 14.8 aŁZazžèGłáoŻázL'ajCây

### éÚóécŸ

ajlájædĐázžçŽĐázŤčŤlćlNázRây■iijNä;æČšarEázŤasCaijCâyãÑĚèčĚæLŘèGłáoŻázL'çŽĐajCây

### èğčăĚşæŮzæąŁ

aŁZazžæŮřçŽĐajCâyãŁçóĀ■ŤāĀŤāĀŤáoŻázL'æŮřçŽĐçšziijNèol'áoČçžgæL'fèGł  
Exception iijLæLŮèĀĚæŸřazžä;ŤäyĀäyłâušā■ŸajlčŽĐajCâyçšzadNiiJL'āĀĆ  
äŁNäeĆriijNäeCædIJä;äçijŮāĚZç;ŚçzIJçŽyāĚşçŽĐćlNázRriijNä;āāRřèČ;äijŽáoŻázL'äyĀāzŽçšzaiijjæCâyNç

```
class NetworkError(Exception):  
    pass  
  
class HostnameError(NetworkError):  
    pass  
  
class TimeoutError(NetworkError):  
    pass  
  
class ProtocolError(NetworkError):  
    pass
```

çĐuāŘŌçŤlæLũarsāRřazžæČŘéĀŽäyýéČčæuā;ŁçŤlæŻázŽaijCâyãžĚriijNä;NäeĆriijŽ

```
try:  
    msg = s.recv()  
except TimeoutError as e:  
    ...  
except ProtocolError as e:  
    ...
```

### èółéőž

èGłáoŻázL'ajCâyçšzazŤerëæĀzæŸřçžgæL'fèGłāĚĚç;ŏçŽĐ Exception  
çšziijN æLŮèĀĚæŸřçžgæL'fèGłéČčazŽæIJnèžnârsæŸřazŮ Exception  
çžgæL'fèĀNæIěçŽĐçšzāĀĆ ar;çóææL'ĀæIJL'çšzāRŊæŮuāžšçžgæL'fèGł BaseException  
riijNä;Ěajāy■āžŤerëä;ŁçŤlæŻäyłāšžçšzæIěáoŻázL'æŮřçŽĐajCâyãĀĆ BaseException  
æŸřäyžçšžçššéĀĀāGžaijCâyýèĀNāŁçŤŹçŽĐriijNærŤæČ KeyboardInterrupt æLŮ  
SystemExit äžæāRŁāĚuāzŮéČčazŽaijŽçžŽázŤčŤlāRŚéĀĀāĚāRũèĀNéĀĀāGžçŽĐajCâyãĀĆ  
āŽāæ■d'riijNæ■ŤèŮuēŁZázŽaijCâyýæIJnèžnæšqāzĀāzLæĐRázL'āĀĆ  
èŁZæuāçŽĐerriijNāĀGāeČä;äçžgæL'f BaseException āRřèČ;äijŽārijeGt'ä;äçŽĐèGłáoŻázL'ajCâyãy■ā

ajlćlNázRây■ajŤāĚèèGłáoŻázL'ajCâyãRřazžæ;Łä;Ůā;äçŽĐazççāĀæŽt'āĚuāRřerřæĀgriijNèČ;äyĚæ  
èŁŸæIJL'äyĀçg■èö;èöææŸřarĚèGłáoŻázL'ajCâyýéĀŽèŁGçžgæL'ŁçžĐāRŁètuæIěāĀĆajlād'■æIČazŤčŤlćlN  
ä;ŁçŤlāšžçšzæIěāŁĚçžĐāRĐçg■ajCâyçšzazžæŸřäŁæIJL'çŤlčŽĐāĀĆāŏČāRřazžèol'çŤlæLũæ■ŤèŮuäyĀā

```
try:
    s.send(msg)
except ProtocolError:
    ...
```

ä;äè£YèĈ;æ■TèŌuæŽt'äd'gèNĈăŽt'çŽDăijĈăyÿiijNăřsăĈRăyNéIcé£ŽæăüiijŽ

```
try:
    s.send(msg)
except NetworkError:
    ...
```

ăĕĈădIJă;ăăĈşăŏŽăZL'çŽDăŪrăijĈăyÿéĜ■ăEŽăžE \_\_init\_\_() æŪzæşTijN  
çăŏăfIă;ăä;£çTlăL'ĂæIJL'ăRĈăTřerĈçTl Exception.\_\_init\_\_() iijNă;NăĕĈiijŽ

```
class CustomError(Exception):
    def __init__(self, message, status):
        super().__init__(message, status)
        self.message = message
        self.status = status
```

çIJNăyLăŌzæIJL'çĈZăĕĜæĂřijNăy■è£ĜExceptionçŽDézYèŏd'èaNăyžæYřæŌăRŪæL'ĂæIJL'ăijăéĂŞç  
.args âşdăĂğăy■. â;Lăd'ŽăĖüăzŪăĜ;æTřăžŞăŞNéĈlăLEPythonăžŞézYèŏd'æL'ĂæIJL'ăijĈăyÿéĈ;ă£Ėéăza  
.args âşdăĂğřijN âŽăæ■d'ăĕĈădIJă;ăă£;çTřăžEĕ£ŽăyĂæ■řijNă;ăăijŽăRŞçŌřæIJL'ăžŽæŪăăĂŽă;ăăŏŽăž  
ăyžăžEăijTĈd'ž .args çŽDă;£çTl iijNăĕĈèŽşăyNăyNéIcé£ŽăyIă;£çTlăEĖç;ŏçŽD Run-  
timeError'ăijĈăyÿçŽDăžd'ăžŞăijŽerIijNă æşlăĎRçIJNraiseer■ăRăy■ă;£çTlçŽDăRĈăTřăyIăTřăYřæĂŌăă

```
>>> try:
...     raise RuntimeError('It failed')
... except RuntimeError as e:
...     print(e.args)
...
('It failed',)
>>> try:
...     raise RuntimeError('It failed', 42, 'spam')
... except RuntimeError as e:
...
...     print(e.args)
...
('It failed', 42, 'spam')
>>>
```

ăĖşăžŌăLŽăžžèĜlăŏŽăZL'ăijĈăyÿçŽDăŽt'äd'Žă£ăæAřijNăřuăRĈăĖĂĈ'PythonăŏYæŪzæŪĜăæăĕ  
<<https://docs.python.org/3/tutorial/errors.html>>‘\_



## 16.9 14.9 æ■TèÕuāijCāyŷāRÕæLZāGzāRēad'ŪçŽDāijCāyŷ

### éUóécŸ

äjäæČšæ■TèÕuāyĀäyĭāijCāyŷāRÕæLZāGzāRēad'ŪäyĀäyĭāy■āRŊçŽDāijCāyŷijŊāRŊæŪüēŸŷāŪāIJ

### èğčāEşæŪzæqĹ

äyžāžEéŞçæŌēāijCāyŷijŊāŷçTĪ raise from èŕ■āRēæĭēāzçæŽŷçōĀā■TçŽD raise  
èŕ■āRēāĀĆ āōČāijŽèōſ'äjäāRŊæŪüāŷĪçTŽāyď'äyĭāijCāyŷçŽDāŷæAſāĀĆäŷŊāæČŕijŽ

```
>>> def example():
...     try:
...         int('N/A')
...     except ValueError as e:
...         raise RuntimeError('A parsing error occurred') from e
→e
...
>>> example()
Traceback (most recent call last):
  File "<stdin>", line 3, in example
ValueError: invalid literal for int() with base 10: 'N/A'
```

äyĹēĪççŽDāijCāyŷæŸŕāyŊēĪççŽDāijCāyŷāžğçTŷçŽDçZſ'æŌēāŌşāZāijŽ

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example
RuntimeError: A parsing error occurred
>>>
```

āIJĹāZđæžŕāy■āRŕāzēcIJŊāĹŕijŊāyď'äyĭāijCāyŷēČŷēçŋæ■TèÕuāĀĆ  
èçAæČšæ■TèÕüēŷæāüçŽDāijCāyŷijŊāŷāāRŕāzēäŷçTĪäyĀäyĭçōĀā■TçŽD except  
èŕ■āRēāĀĆ äy■ēŷçŕijŊāŷæŸŷāRŕāzēēĀŽēŷGæşççIJŊāijCāyŷāržēsāçŽD \_\_cause\_\_  
āśđæĀğæĪēēŷşēyĭāijCāyŷēŞçāĀĆäŷŊāæČŕijŽ

```
try:
    example()
except RuntimeError as e:
    print("It didn't work:", e)

    if e.__cause__:
        print('Cause:', e.__cause__)
```

āŷŞāIJĪ except āĪŪäy■āRĹæIJĹāRēad'ŪçŽDāijCāyŷēçŋæLZāGzāŪüāijŽāŕijèGſ'äyĀäyĭéŽŖèŪŔçŽDā

```
>>> def example2():
...     try:
...         int('N/A')
```

```

...     except ValueError as e:
...         print("Couldn't parse:", err)
...
>>>
>>> example2()
Traceback (most recent call last):
  File "<stdin>", line 3, in example2
ValueError: invalid literal for int() with base 10: 'N/A'

```

ǎIJlǎd'ĐçŘĚäyŁēřřaijĆăyŷçŽĐæŮúăĂŽiijŃăŔęǎd'ŮăyĂăyłaijĆăyŷăŔŚçŦšăžĚiijŽ

```

Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example2
NameError: global name 'err' is not defined
>>>

```

ēfŽăyłăŹŃă■Řăy■iijŃă;ăăŔŃæŮűēŬăŹŮăžĚăyđ'ăyłaijĆăyŷçŽĐăŹăAřřiijŃă;ĚăŸřăŹăijĆăyŷçŽĐēğčēfŽăŮűăĂŽiijŃNameErrorăijĆăyŷçēŹă;IJăyžçłŃăžŔăIJăçžŁăijĆăyŷçēŹăŁŽăĜžiiijŃēĂŃăy■ăŸřă;■ăžŬ

ăęČădIJiijŃă;ăăČšăŹ;çŦčæŬŔăijĆăyŷēŹ;iiijŃăŔřă;ŹçŦŦ raise from None:

```

>>> def example3():
...     try:
...         int('N/A')
...     except ValueError:
...         raise RuntimeError('A parsing error occurred') from _
↳None
...
>>>
example3()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 5, in example3
RuntimeError: A parsing error occurred
>>>

```

èőléőž

ǎIJlěőŹēőăăžččăAæŮűiijŃăIJlǎŔęǎd'ŮăyĂăył except äžččăAăIŮăy■ă;ŹçŦŦ raise  
 èř■ăŔēçŽĐăŮűăĂŽă;ăēęAçŁ'žăŁŃăŕŔăŹčăžĚăĂĆăđ'ğăđ'ŽăŦŕăČĚăĚăyŃiijŃēŹçğ■  
 raise èř■ăŔēēČ;ăžŦēŕēēŹăŦžăĹŔ raise from èř■ăŔēăĂĆăžšăŕšăŸŕēŕt'ă;ăăžŦēŕă;ŹçŦŦăyŃēŹčēŹçğ

```

try:
...
except SomeException as e:
    raise DifferentException() from e

```

ēfŽăăűăĂŽçŽĐăŬšăžăŸřă;ăăžŦēŕēăŸŹçđ'žçŽĐăŕĚăŬšăžăēŹ;ăŬčēŹăŹăŹăĂĆ

äzšåršæYřèrt'iijÑDifferentException æYřçZt'æÕëäzÕ SomeException  
èa■çTšèĀNæIëāĀĆ èŁŻçg■āĔšçşzāRřäzèäzÕāZđæžřçzŞæđIJäy■çIJNāĠzæIëāĀĆ

æçĆæđIJä;āāČRäyNéIćèŁZæūāĀEZäzčçāAīijNā;āāz■çDūāijŽā;ŮāĹřäyĀäyĹéŞ;æÕëāijCāyÿiijN  
äy■èŁĠèŁZäyĹāzūæşæIJL'ā;ĹäyĒæŽřçŽĐèrt'æYÕèŁZäyĹāijCāyÿéŞ;āĹřāzTæYřāĒĒéČĹāijCāyÿèŁYæYřæŞ

```
try:  
    ...  
except SomeException:  
    raise DifferentException()
```

ā;Şā;āā;ŁçTĹ raise from èr■āRëçŽĐèrīijNārşā;ĹäyĒæēŽçŽĐèāĹæYÕæŁZāĠzçŽĐæYřçñnāzNāyĹā  
æIJĀāRÕäyĀäyĹā;Nā■Räy■éŽŘèŮRāijCāyÿéŞ;āĹæAřāĀĆ  
ār;çōæēŽŘèŮRāijCāyÿéŞ;āĹæAřāy■āĹ'äžÕāZđæžřīijNāRÑæŮūāōČäzşäyčād'säzĒā;Ĺād'ŽæIJL'çTĹçŽĐèrt'  
äy■èŁĠäyĠzNçŽĒāzşç■L'īijNæIJL'æŮūāĀZāRĹāŁçTŽéĀĆā;ŞçŽĐāĹæAřāzşæYřā;ĹæIJL'çTĹçŽĐāĀĆ

## 16.10 14.10 éĠæŮřæŁZāĠžèćnæ■TèÕūçŽĐāijCāyÿ

éŮóécY

ā;āāIJläyĀäyĹ except āĪŮäy■æ■TèÕūāzĒäyĀäyĹāijCāyÿiijNçÕřāIJæČşéĠæŮřæŁZāĠžāōČāĀĆ

èğčāĒşæŮzæāĹ

çōĀā■TçŽĐä;ŁçTĹäyĀäyĹā■TçNñçŽĐ rasie èr■āRëā■şāRřīijNā;NāçCīijŽ

```
>>> def example():  
...     try:  
...         int('N/A')  
...     except ValueError:  
...         print("Didn't work")  
...         raise  
...  
  
>>> example()  
Didn't work  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
  File "<stdin>", line 3, in example  
ValueError: invalid literal for int() with base 10: 'N/A'  
>>>
```

èóĹèőž

èŁZäyĹéŮóécYéĀZäyÿæYřā;Şā;æēIJĀèçAāIJĹæ■TèÕūāijCāyÿāRÕæŁ'gèāNæşŘäyĹæŞ■ā;IJīijĹæřTæçCè  
äyĀäyĹā;ĹäyÿèğAçŽĐçTĹæşTæYřāIJĹæ■TèÕūæŁ'ĀæIJL'āijCāyÿçŽĐād'ĐçŘĒāZĹäy■īijŽ

```

try:
    ...
except Exception as e:
    # Process exception information in some way
    ...

    # Propagate the exception
    raise

```

## 16.11 14.11 èŁŞăĜžè■ēăŞĹăĖăæĀŗ

éŮőéćŸ

äĵääŸŊæIJžèĜĥăűşçŽĐćĬŊăžŔèĈĭçŦşăĹŔè■ēăŞĹăĖăæĀŗĭĵĹăŗŦăēĈăžşăĭĵĈçĹ'žăĀğăĹŮăĭçŦĬéŮőéćŸ

èğĉăĖşăŮžăæĹĹ

èēĀēĭŞăĜžăŸĀăŸĹè■ēăŞĹăűĹăĀŗĭĵŊăŔŗăĭçŦĬĭ warning.warn()  
ăĜĭăŦŗăĀĈăĭŊăēĈĭĵŽ

```

import warnings

def func(x, y, logfile=None, debug=False):
    if logfile is not None:
        warnings.warn('logfile argument deprecated',
↳DeprecationWarning)
    ...

```

warn() çŽĐăŔĈăŦŗăŸŗăŸĀăŸĹè■ēăŞĹăűĹăĀŗăŊăŸăŸĹè■ēăŞĹçşžĭĵŊè■ēăŞĹçşžăĪĴ'ăēĈăŸŊăĜă  
DeprecationWarning, SyntaxWarning, RuntimeWarning, ResourceWarning, æĹŮ FutureWarn-  
ing.

ăržè■ēăŞĹçŽĐăđ'ĐçŔĖăŔŮăĖşăžŎăĭăăēĈăĭŦēŦŔēăŊèğĉēĜĹăŽĹăžēăŔĹăŸĀăžŽăĖŮăžŮéĖ■çĭŏăĀĈ  
ăĭŊăēĈĭĵŊăēĈăđĪăĭăăĭçŦĬ-W all éĀĹéăžăŎžēŦŔēăŊPythonĭĵŊăĭăĭĵŽăĭŮăĹŔăēĈăŸŊçŽĐēĭŞăĜžĭĵŽ

```

bash % python3 -W all example.py
example.py:5: DeprecationWarning: logfile argument is deprecated
  warnings.warn('logfile argument is deprecated',
↳DeprecationWarning)

```

éĀŽăŸŸăĬèőőĭĵŊè■ēăŞĹăĭĵŽēĭŞăĜžăĹŔăăĜăĜĖēŦŽēŗŗăŸĹăĀĈăēĈăđĪăĭăăĈşèőşè■ēăŞĹēĭŋă■ăŸŸă  
-W error éĀĹéăžĭĵŽ

```

bash % python3 -W error example.py
Traceback (most recent call last):
  File "example.py", line 10, in <module>
    func(2, 3, logfile='log.txt')

```

# èóíèőž

ä: ɪäy zä Rēad' Ūäy Ääylä EĖç: ǫä Ğ: æ Trāz Šç ŽDē■ ēä ŠLä: ɛç Tlā: Nā■ Riiŋ Nāy Nélcäi: Tçd' žäz Eäy Ääylä šçä

éZÿèóð' æĈĖāĖtāyNriiĴāzūüä■æŸrāL' ĀæIJL'è■ēāSŁæūŁæAřéĈ; äijŽāĠžçŌřāĀĈ-W  
 éĀL'ēāzēĈ; æŌġāŁūē■ēāSŁæūŁæAřçŽĎē, ŠāĠžāĀĈ -W all  
 äijŽē, ŠāĠžāL' ĀæIJL'è■ēāSŁæūŁæAřiiĴN-W ignore āf;çTēāŌL'æL'ĀæIJL'è■ēāSŁiiĴN-W  
 error āřEē■ēāSŁē;ñæ■ćāŁŘāijCāyŷāĀĈ āŘēād' ŪāyĀçg■ēĀL'æN'riiĴNā;āēŸYāŘrāzēā;ŁçTī  
 warnings.simplefilter() āĠ;æŢřæŌġāŁūē, ŠāĠžāĀĈ always  
 āŘćæŢřāijŽēŌ' æL'ĀæIJL'è■ēāSŁæūŁæAřāĠžçŌriiĴN` ignore  
 āf;çTēēřCāL'ĀæIJL'çŽĎē■ēāSŁiiĴNerror āřEē■ēāSŁē;ñæ■ćāŁŘāijCāyŷāĀĈ

16.12 14.12 èřČèřŤašžæIňčŽĎćÍNăžŔat'ŕæžČéŤŽèřř

éŮőécŸ

ä;äcŽĐćíNǎžRát'PæžČǎŘŎèrěæĂŎæăuăŎžèrČèrTǎoČiijš

# èğčå£şæŮźæąŁ

æĈcædIJä;ăçŽĐđĺNăžRăZăăyžæşŘăýłaijCăyÿëĂŇat'ĲæžČiiJÑefŘèąŃ  
python3 -i someprogram.py                      ǎRfáL'gèaŇčôĂ■TçŽĐěřČerTăĂĆ

-i éĀL'ėążăŔřèđl'ćÍŇăžŔçzŞæİşăŔŌæL'ŞăijĂăyĂăyġăžd'ăžŞăijŔshellăĂĈ  
çĎŭăŔŌăjăăŕřşēĈjæşēçIJŇçŎŕăĈĈijŇăĭŇăēĈijŇăĂĜëđĭăjăæIJL'ăyŇéĬçŻĎăžĉăĂijŻ

```
# sample.py

def func(n):
    return n + 10

func('Hello')
```

èĤŔëąŇ python3 -i sample.py äijŻæIJL'çşzäijijăēĈăyŇçŻĎēĭŞăĜzŭijŻ

```
bash % python3 -i sample.py
Traceback (most recent call last):
  File "sample.py", line 6, in <module>
    func('Hello')
  File "sample.py", line 4, in func
    return n + 10
TypeError: Can't convert 'int' object to str implicitly
>>> func(10)
20
>>>
```

ăēĈăđIJăjăçIJŇăy■ăĤŕăyĤéĬçēĤZăăŭçŻĎijŇăŔŕăžēăIJĬćÍŇăžŔăt'ĭæžĈăŔŌæL'ŞăijĂPythonçŻĎŕĈŕĭ

```
>>> import pdb
>>> pdb.pm()
> sample.py(4) func()
-> return n + 10
(Pdb) w
  sample.py(6) <module>()
-> func('Hello')
> sample.py(4) func()
-> return n + 10
(Pdb) print n
'Hello'
(Pdb) q
>>>
```

ăēĈăđIJăjăçŻĎăžĉăĂæL'ĂăIJĬçŻĎçŎŕăĈĈăĭĤéŽĭēŎŭăŔŬăžd'ăžŞshellijĤăŕŤăēĈăIJĭăşŔăyġăIJ■ăĤă  
éĂŽăyŕăŔŕăžēă■ŤēŎŭăijĈăyŕăŔŎēĜĭăŭşæL'Şă■ŕēŭşēyġăĤăăĂŕăĂĈăĭŇăēĈijŻ

```
import traceback
import sys

try:
    func(arg)
except:
    print('**** AN ERROR OCCURRED ****')
    traceback.print_exc(file=sys.stderr)
```

èēĂæŸŕăjăçŻĎćÍŇăžŔăşşæIJL'ăt'ĭæžĈijŇēĂŇăŔġăŸŕăžĝçŤşăžĖăyĂăžZăjăçIJŇăy■ăēĜĈçŻĎçzŞăđĤ

ä;ääIJlæDšâĖt'ëüčçŽDâIJræŮzæŔSâĖĕäyÄäyN print () èŕ■âŔëäzšæŸŕäyĭäy■éŤŽçŽDëÄL'æNĭ'ãÄĆ  
äy■èŕĜiijNĕeAæŸŕä;ææL'ŠçŏŮèŕŽæâũâAŽiijNæIJL'äyÄäzŽârRæĽÄâũĝâŔŕäzëäyŏâĽĭ'ä;ääÄĆ  
éĕŮâĖĽiijNtraceback.print\_stack () âĜ;æŦŕäijŽä;äçĭNâžŔèŕŔëaŇâĽŕéCčäyĭçČžçŽDæŮũâÄŽâĽŽ

```
>>> def sample(n):  
...     if n > 0:  
...         sample(n-1)  
...     else:  
...         traceback.print_stack(file=sys.stderr)  
...  
>>> sample(5)  
File "<stdin>", line 1, in <module>  
File "<stdin>", line 3, in sample  
File "<stdin>", line 3, in sample  
File "<stdin>", line 3, in sample  
File "<stdin>", line 3, in sample  
File "<stdin>", line 3, in sample  
File "<stdin>", line 5, in sample  
>>>
```

âŔëâd'ŮiijNä;æèŕŸâŔŕäzëâČŔäyNéĭcèŕŽæâũä;ŕçŦĭ  
âIJläzzä;ŦâIJræŮzæL'NâĽĭçŽDâŔŕâĽĭèŕČerŦâŽĭiijŽ pdb.set\_trace()

```
import pdb  
  
def func(arg):  
...  
    pdb.set_trace()  
...
```

ä;ŠçĭNâžŔæŕŦè;Čâd'ĝèAŇä;äæČšërČerŦæŐĝâĽüætAçĭNâžëâŔĽâĜ;æŦŕâŔCæŦŕçŽDæŮũâÄŽèŕŽäyĭâ  
äĭNâeČiijNäyÄæŮçerČerŦâŽĭäijÄâĝNèŕŔëaŇiijNä;ääŕsèČ;âd'šä;ŕçŦĭ  
print æĭèèĜCætŇâŔŸéĜŔâÄijæĽŮæŦšâĜzæšŔäyĭâŠ;äzd'ærŦæç  
æĭèèŐũâŔŮèŕ;èŸĭäŕæAŕãÄĆ w

## ëŏĭëŏž

äy■èeAârEërČerŦäijDçŽDèŕĜäžŐâd'■æĭCâNŮâÄĆäyÄäzŽçŏÄâ■ŦçŽDëŦŽèŕŕâŔĭeIJÄeAèĝCâršçĭNâ  
âŏdëŽĖçŽDëŦŽèŕŕäyÄèĽnæŸŕääEæâĽçŽDæIJÄâŔŐäyÄëaŇãÄĆ  
ä;ääIJläijÄâŔSçŽDæŮũâÄŽiijNâžšâŔŕäzëâIJlä;äeIJÄeAerČerŦçŽDâIJræŮzæŔSâĖĕäyÄäyN  
print () âĜ;æŦŕæĭèerĽæŮ■äŕæAŕiijĽâŔĭeIJÄeAæIJÄâŔŐâŔSäyČçŽDæŮũâÄŽâĽäéŽd'èŕŽäžŽæL'Šâ■ŕ

èŕČerŦâŽĭçŽDäyÄäyĭäyÿèĝAçŦĭæšŦæŸŕèĝCætŇnæšŔäyĭâũšçzŔât'ĭæžČçŽDâĜ;æŦŕäy■çŽDâŔŸéĜŔâÄ  
çšëeAšæÄŐæâũâIJläĜ;æŦŕât'ĭæžČâŔŐèŕŽâĖèerČerŦâŽĭæŸŕäyÄäyĭä;ĽæIJL'çŦĭçŽDæĽÄèČ;ãÄĆ

ä;Šä;äæČšèĝçâĽŮäyÄäyĭeĭäyÿâd'■æĭCçŽDçĭNâžŔiijNâžŦâsČçŽDæŐĝâĽüèÄžè;Šä;ääy■æŸŕä;ĽæyĖ  
æŔSâĖĖ pdb.set\_trace () èŕŽæâũçŽDèŕ■âŔëâršâ;ĽæIJL'çŦĭläžEãÄĆ

âŏdëŽĖäyĽiijNçĭNâžŔäijŽäyÄçŽt'èŕŔëaŇâĽŕççŕâĽŕ set\_trace()  
èŕ■âŔëä;■ç;ŏiijNçDũâŔŐçnNéĭ'ñèŕŽâĖèerČerŦâŽĭãÄĆ çDũâŔŐä;ääŕsâŔŕäzëâAžæŽt'âd'ŽçŽDäžNâžEãÄĆ

æĈædĪJä;ää;ŁçŦĪIDEæĪěăAŽPythonăijĂăRŠĭijŇěĂŽăÿÿIDEéĈ;ăijŽæRŘă;ŻeĠăũşçŽDërĈërŦăZĪăĪěă  
æZĭ'ăd'ŽěŁZæŰzéĪçŽDăŁæAŁăRăRăžěăRĈèĂĈă;ăă;ŁçŦĪçŽDIDEæL'ŇăĒŇăĂĈ

## 16.13 14.13 çŻŽă;ăçŽDĈĪŇăžRăAŽæĂĝèĈ;ætŇërŦ

éŰóéĈŸ

ă;ăæĈşætŇërŦă;ăçŽDĈĪŇăžRěŁRěăŇæL'ĂeĹşèť'żçŽDæŰűéŰť'ăżűăAŽæĂĝèĈ;ætŇërŦăĂĈ

èĝĉăEşæŰzæąĹ

æĈædĪJä;ăăRĪæŸřçŮĂă■ŦçŽDæĈşætŇërŦăÿŇă;ăçŽDĈĪŇăžRæŦť'ă;ŞeĹşèť'żçŽDæŰűéŰť'ĭijŇ  
éĂŽăÿÿă;ŁçŦĪUnixæŰűéŰť'ăĠ;æŦřăřşěăŇăžĒĭijŇăřŦăeĈĭijŽ

```
bash % time python3 someprogram.py
real 0m13.937s
user 0m12.162s
sys 0m0.098s
bash %
```

æĈædĪJä;ăeŁŸéĪJĂeĕAăÿĂăÿĪĈĪŇăžRăRĎăÿĪçzEèĹĈçŽDèřççzEăĹăŚĹĭijŇăRăřžěă;ŁçŦĪ  
cProfile æĪăăĪŰĭijŽ

```
bash % python3 -m cProfile someprogram.py
      859647 function calls in 16.016 CPU seconds

Ordered by: standard name

ncalls  tottime  percall  cumtime  percall_
→filename:lineno(function)
      263169    0.080    0.000    0.080    0.000 someprogram.
→py:16(frange)
        513    0.001    0.000    0.002    0.000 someprogram.
→py:30(generate_mandel)
      262656    0.194    0.000    15.295    0.000 someprogram.py:32(
→<genexpr>)
         1    0.036    0.036    16.077    16.077 someprogram.py:4(
→<module>)
      262144   15.021    0.000    15.021    0.000 someprogram.py:4(in_
→mandelbrot)
         1    0.000    0.000    0.000    0.000 os.py:746(urandom)
         1    0.000    0.000    0.000    0.000 png.py:1056(_readable)
         1    0.000    0.000    0.000    0.000 png.py:1073(Reader)
         1    0.227    0.227    0.438    0.438 png.py:163(<module>)
        512    0.010    0.000    0.010    0.000 png.py:200(group)
...
bash %
```



äy■ēfGéĀŽāyāČĚāEĭtæYřāzNāžŎēfŽāyd'äylæđAçñřāzNéŮt'āĀĆærTāęĆä;ăăüşçzŘçšěéAŞăzččăĀèĚĬ  
årzāžŎēfŽāžŽāĠ;æTřčŽDæĀġèČ;ætNērTřijNāŘřāzčă;ĤčTlāyĀäylčōĀă■TčŽDčĚēčřāŽlřijŽ

```
# timethis.py

import time
from functools import wraps

def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.perf_counter()
        r = func(*args, **kwargs)
        end = time.perf_counter()
        print('{}.{} : {}'.format(func.__module__, func.__name__,
        ↪end - start))
        return r
    return wrapper
```

èēAä;ĤčTlēfŽāylčēĚēčřāŽlřijNāŘlēIJĀèēAārEăĚŮæTč;ōăIJlă;ăèēAèfŽèqNæĀġèČ;ætNērTčŽDăĠ;æT

```
>>> @timethis
... def countdown(n):
...     while n > 0:
...         n -= 1
...
>>> countdown(10000000)
__main__.countdown : 0.803001880645752
>>>
```

èēAætNērTæšŘāylăzččăĀăĬŮēfŘèqNæŮŮēŮt'řijNă;ăăŘřāzčăōŽăžL'ăyĀäylăyLăyNæŮĠčōaçŘĚăŽlřijN

```
from contextlib import contextmanager

@contextmanager
def timeblock(label):
    start = time.perf_counter()
    try:
        yield
    finally:
        end = time.perf_counter()
        print('{} : {}'.format(label, end - start))
```

äyNéĬcæYřă;ĤčTlēfŽāylăyLăyNæŮĠčōaçŘĚăŽlčŽDă;Nă■ŘřijŽ

```
>>> with timeblock('counting'):
...     n = 10000000
...     while n > 0:
...         n -= 1
...
counting : 1.5551159381866455
```

```
>>>
```

```
    ȧřžžŌætNērTȧŁŁārRçŽĐžččāAçL'ĠæōțēfRèaŊæĂğèĈ;ııjŊä;ŁçTÍ      timeit
æłāāıŪāıjŽāŁŁæŪžāŁııjŊäŁŊæĈııjŽ
```

```
>>> from timeit import timeit
>>> timeit('math.sqrt(2)', 'import math')
0.1432319980012835
>>> timeit('sqrt(2)', 'from math import sqrt')
0.10836604500218527
>>>
```

```
    timeit āıjŽæL'ğèaŊçññäyĂäyłāRĈæTřäy■ēr■āRē100äyĠæñāāžžēōaçōŪèŁRèaŊæŪúéŪť āĂĈ
çññāžŊäyłāRĈæTřäyŸřèŁRèaŊæıNērTāžŊāL■ēĚ■ç;ōçŌřāçĈāĂĈæçĈæđIJā;āæĈşæTžāRŸāŁıçŌřæL'ğèaŊæñ
āRřžžæĈRāyŊéłçèŁZæāūèōŁç;ō number āRĈæTřçŽĐāĀııjŽ
```

```
>>> timeit('math.sqrt(2)', 'import math', number=10000000)
1.434852126003534
>>> timeit('sqrt(2)', 'from math import sqrt', number=10000000)
1.0270336690009572
>>>
```

## èőłèőž

```
    āıŞæL'ğèaŊæĂğèĈ;ætNērTçŽĐæŪūāĂŽııjŊéIJāèèAæşłæĐRçŽĐæŸřā;æŌŭāRŪçŽĐçzŞæđIJéĈ;æŸřæ
time.perf_counter() āĠ;æTřāıjŽāIJłçzŽāōŽāžşāRřäyŁèŌŭāRŪæIJĀénŸçş;āžèçŽĐèōææŪūāĀıjāĂĈ
äy■ēŁĠııjŊāōĈāž■çĐűēŁŸæŸřāşžžŌæŪúéŞşæŪúéŪťııjŊäŁŁad'ŽāZächt'āāıjŽā;şāŞ■āŁřāōĈçŽĐçş;çāōāžēı
āçĈæđIJā;āāržžŌæL'ğèaŊæŪúéŪť æŽť æĐşāĒť èűčııjŊä;ŁçTÍ    time.process_time()
æłæžžæŽŁāōĈāĂĈäŁŊæĈııjŽ
```

```
from functools import wraps
def timethis(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        start = time.process_time()
        r = func(*args, **kwargs)
        end = time.process_time()
        print('{}.{} : {}'.format(func.__module__, func.__name__,
    ↪end - start))
        return r
    return wrapper
```

```
    æIJĀāRŌııjŊāçĈæđIJā;āæĈşèŁZèaŊæŽť æūsāĒèçŽĐæĂğèĈ;āŁĒæđRııjŊéĈçāžŁä;æéIJĀèèAèřèçzĒéŸı
time    āĂĀtimeit    āŞŊāĒūāžŪçŽyāĒşæłāāıŪçŽĐæŪĠæaçāĂĈ
èŁZæāūā;āāRřžžèçRĒèğçāŞŊāžşāRřçŽyāĒşçŽĐāūōāıjĈāžæāRĒäyĀāžŽāĒūāžŪéŽúéŸşāĂĈ
èŁŸāRřžžæāRĈèĂĈ13.13ārRèŁĈäy■çŽyāĒşçŽĐäyĂäyłāŁZāžžèōææŪūāŽłçşzçŽĐäŁŊā■RāĂĈ
```

## 16.14 14.14 aŁăéĀşçÍŃăžŘèĚŘèąŃ

éŮóécŸ

ä;ăçŽĎçÍŃăžŘèĚŘèąŃăđ'łæĚćiiŃŃă;ăæČşăIJăy■ă;ŕçŤlăđ'■ăiĆæŁĂæIJræŦăęĈCæLŦ'ăśŦæĹŮJITçijŮ

èğĉăEşşæŮzæąĹ

ăĚşăžŮçÍŃăžŘăijŸăŃŮçŽĎçŋăyĂăyłăĜĚăĹŽæŸŦăĂIJăy■ëęAăijŸăŃŮăĂIriŷŃçŋăžŃăyłăĜĚăĹŽæŸŦăĚĆăđIJă;ăçŽĎçÍŃăžŘèĚŘèąŃçijŞæĚćiiŃŃéęŮăĚĹă;ăă;Ůă;ŕçŤlă14.13ăřŘèĹĆçŽĎæŁĂæIJŦăĚĹăŦŦăăŮĈèĚŽè

éĀžăyŷæİëëőşă;ăăijŽăŦŦşçŮŦă;ăă;ŮçÍŃăžŘăIJăŦŦşæŦŦăĜăăyłçĈ■çĆăIJŦæŮzèĹşèŦ'žăžĚăđ'ğéĜŦæŮŮéăŦŦăęĈăĚĚă■ŸçŽĎæŦŦæ■őăđ'ĎçŘĚă;łçŮŦăĈăyĂæŮęă;ăăőŽă;■ăĹŦèĚŽăžŽçĆziiŃŃă;ăăŦŦăŦŦăžă;ŕçŤlăyŮ

ă;ŕçŤlăĜ;æŦŦ

ă;Ĺăđ'ŽçÍŃăžŘăŦŦŸăĹŽăijĂăğŃăijŽă;ŕçŤlăPythonèŦ■ēĹăĚŽăyĂăžŽçőĂă■ŦèĎŽæIJŋăĂĆă;ŞçijŮăĚŽèĎŽæIJŋçŽĎæŮŮăĂŽiiŃŃéĂžăyŷăžăăČŦăžĚăĚŽæŦŦæŮăçžŞæđĎçŽĎăžççăAriŷŦŦăŦăęĈiiŃŽ

```
# somescript.py

import sys
import csv

with open(sys.argv[1]) as f:
    for row in csv.reader(f):

        # Some kind of processing
        pass
```

ă;ĹăŦŦşæIJĹăžžçşëęĂşiiŃŃăČŦèĚŽæăăăőŽăžĹăIJăĹăŦŦşĂëŃăŦŦçŽĎăžççăĂëĚŦèąŢëŦăİëëęĂŦŦăŦŦşëĚŽçğ■ēĂşăžăăăăijĆæŸŦçŦşăžŮăşĂëĈĹăŦŦŸéĜŦăŦŦăĚĹăşĂăŦŦŸéĜŦçŽĎăăđçŮŦăŮăijŦiiŷĹă;ŕçŤlăşĂëĈăăŽăă■đ'iiŃŃăęĈăđIJă;ăæČşèŮŦçÍŃăžŘèĚŘèąŢăŽŦăŦŦăžŽiiŃŢăŦŦēIJăëęĂăŦŦēĎŽæIJŋèŦ■ăŦŦăŦŦă;ăĚăăĜ;æŦŦ

```
# somescript.py
import sys
import csv

def main(filename):
    with open(filename) as f:
        for row in csv.reader(f):
            # Some kind of processing
            pass

main(sys.argv[1])
```

éĂşăžęçŽĎăăőăijĆăŦŮăĚşăžŮăăđéŽĚèĚŦèąŢçŽĎçÍŃăžŘiiŃŃăy■ēĜăăžăă■őçžŦŦŦŦiiŃŃă;ŕçŤlăĜ;æŦŦŦ30%çŽĎăĂğëĈ;æŦŦă■ĜæŸŦăĹăyŷëğĂçŽĎăĂĆ

ăŦ;ăŦŦŦëĈ;ăŮžăŮĹăśđăĂğëőĚéŮő

æŕŔäyÄæñä;ƒçŦłçĆż(.)æŞ■ä;IJçñæİèèõŒÉŮõásđæĀğçŽĐæŮüāĀŽäijŽäyęæİééİād'ŮçŽĐäijĂéŦĂāĀ  
ãoČäijŽęęăŔŚçŁ'záoŽçŽĐæŮzæşŦiijŊærŦăęĆ      \_\_getattr\_\_()      ăŞŊ  
\_\_getattr\_\_() iijŊèŒŽăžZæŮzæşŦäijŽèŒŽëăŊăŮăËyæŞ■ä;IJæŞ■ä;IJăĂĆ

éĂŽăyÿä;ăăŔŕäzëä;ƒçŦł      from module import name  
èŒŽæăüçŽĐäŕijăĚëă;ćäijŔiijŊäzëăŔĹä;ƒçŦłçżŚáoŽçŽĐæŮzæşŦăĂĆ  
ăĀĞèõ;ă;ăæIJŁ'ăęĆăyŊçŽĐăžçăĀçŁ'ĠăõŦiijŽ

```
import math

def compute_roots(nums):
    result = []
    for n in nums:
        result.append(math.sqrt(n))
    return result

# Test
nums = range(1000000)
for n in range(100):
    r = compute_roots(nums)
```

ăIJăĹŚăžñæIJzăŽłäyŁéİćæŦŊèŦŦçŽĐæŮüāĀŽiijŊèŒŽäyłçİŊăžŔèŁsèt'zăžĚăđ'ğæęĆ40çğŚăĂĆçŎŕăIJă  
compute\_roots() ăĠ;æŦŕăęĆăyŊiijŽ

```
from math import sqrt

def compute_roots(nums):

    result = []
    result_append = result.append
    for n in nums:
        result_append(sqrt(n))
    return result
```

ăŒôæŦzăŔŎçŽĐçŁĹæIJñèŒŔëăŊæŮüéŮŦ'ăđ'ğæęĆæŸŕ29çğŚăĂĆăŦŕäyĂäy■ăŔŊăžŊăđ'ĐăŕŝæŸŕæŮĹé  
çŦł      sqrt()      äžçæŽŒäžĚ      math.sqrt()      ăĂĆ      The result.  
append()      æŮzæşŦèćnètŊçżŽäyĂäyĹăsĂéĆĹăŔŸéĠŔ      result\_append  
iijŊçĐüăŔŎăIJăĚĚéĆĹă;łçŎŕăy■ă;ƒçŦłăőČăĂĆ

äy■èŒĠiijŊèŒŽăžZæŦzăŔŸăŔĹæIJŁ'ăIJăđ'ğęĠŔéĠŮăđ'■ăžçăĀăy■æŁ'■æIJŁ'æĐŔăžŁ'iijŊærŦăęĆă;łç  
ăŽăæ■đ'iijŊèŒŽăžZäijŸăŊŮăžşăŔĹæŸŕăIJăşŔăžŽçŁ'záoŽăIJŕæŮzæŁ'■ăžŦèŕëèćnä;ƒçŦłăĂĆ

### çŔĚëğçăsĂéĆĹăŔŸéĠŔ

ăžŊăĹ'■æŔŔèŒĠiijŊăsĂéĆĹăŔŸéĠŔăijŽærŦăĚĹăsĂăŔŸéĠŔèŒŔëăŊæŦşăžęăŦăăĂĆ  
ărzăžŎéćŚçZăęõŒÉŮõçŽĐăŔ■çğŕiijŊéĂŽèŒĠăŕĚèŒŽăžZăŔ■çğŕăŔŸæĹŔăsĂéĆĹăŔŸéĠŔăŔŕäzëăŁăęĂşçİŊă  
ă;ŊăęĆiijŊçIJŊăyŊăžŊăĹ'■ărzăžŎ compute\_roots() ăĠ;æŦŕèŒŽëăŊăŒôæŦzăŔŎçŽĐçŁĹæIJñiijŽ

```
import math

def compute_roots(nums):
    sqrt = math.sqrt
```

```
result = []
result_append = result.append
for n in nums:
    result_append(sqrt(n))
return result
```

åIJléŹäyłçŁŁæIJñäy■īijŃsqrt äzŎ match æłaiŮëcñæŃłăĜżázúæŤłăĚëazĖäyĂäyłāsĂéĈłāŔŸéĜŔ  
æĖĈăđIJă;ăĖŋŖëąŃëŋŹäyłäzčçăĀīijŃăđ' ġæĖĈêŁŝet' z25çġŖīijŁărzäžŎăzŃăŁ'■29çġŖăŔŁæŸřäyĂäyłæŤzèŁŹ  
ëŋŹäyłéćłăđ' ŮçŹĐăŁăéĂŝăŎŝăŹăæŸřăŹăäyžărzäžŎăsĂéĈłāŔŸéĜŔ sqrt  
çŹĐăŝëæŁ;çĖĀăŋăžŎăĚłāsĂăŔŸéĜŔ sqrt

ărzäžŎçŝzäy■çŹĐăŝđæĂġëôŋéŮôăžŝăŔŃæăüëĂĈçŤłäžŎèŋŹäyłăŎŝçŖĖăĂĈ  
éĂŹäyŷæłëôŝīijŃæŝëæŁ;æŝŖäyłăĀijærŤăĖĈ self.name  
ăijŹærŤëôŋéŮôăyĂäyłāsĂéĈłāŔŸéĜŔëĖĀăĚăyĂăžŹăĂĈ åIJłăĖĚéĈłăłçŮŖäy■īijŃăŔřăžëărĖæŝŖäyłéIJăĚ

```
# Slower
class SomeClass:
    ...
    def method(self):
        for x in s:
            op(self.value)

# Faster
class SomeClass:
    ...
    def method(self):
        value = self.value
        for x in s:
            op(value)
```

éĀłăĚăy■ăŋĚëĖĀçŹĐăŁ;èŝă

ăzză;ŤăŮŭăĂŹă;Ŗă;ăă;ŋçŤłéćłăđ' ŮçŹĐăđ' ĐçŖĖăŝĈīijŁærŤăĖĈèĈĚëĉŕăŹłăĂĀăŝđæĂġëôŋéŮôăĂĀăŔ  
ærŤăĖĈçIJŃäyŃăĖĈăyŃçŹĐèŋŹäyłçŝzīijŹ

```
class A:
    def __init__(self, x, y):
        self.x = x
        self.y = y
    @property
    def y(self):
        return self._y
    @y.setter
    def y(self, value):
        self._y = value
```

çŎŕăIJléŹëąŃäyĂäyłçôĂăŤăŤŃërŤīijŹ

```
>>> from timeit import timeit
>>> a = A(1,2)
```

```
>>> timeit('a.x', 'from __main__ import a')
0.07817923510447145
>>> timeit('a.y', 'from __main__ import a')
0.35766440676525235
>>>
```

āRrāzēçIJNāLrīijNēōēŮōāsđæĀğycZÿærTāsđæĀğxèĀNēlĀæĒcçZDäy■æ■cäyĀçCzçCzīijNād'gæçCael  
æçCædIJä;āāIJlæDRæĀğēC;çZDēfīijNēCčāzLārśéIJĀēçAēG■æŪrāōæğEäyNārřāzŌyçZDāsđæĀğēōēŮōāz  
æçCædIJæšqæIJL'āfĒēçAīijNārřā;ççTlçōĀā■TāsđæĀğāRğāĀC  
æçCædIJāzĒāzĒæYrāZāyZāĒŪāzŪçijŪçlNēr■ēlĀēIJĀēçAā;ççTlgetter/setteraĠ;æTṛāřsāŌzāfōæTzāzççāAēç

ä;ççTlāEĒç;ōçZDāōzāZl

āEĒç;ōçZDæTṛæ■ōçszādNærTāçCā■ŪçñçäyśāĀAāĒCçzDāĀAālŪēālāĀAēZEāRlLāšNā■ŪāEÿēC;æYr  
æçCædIJä;āæCšēĠāūsāōđçŌræŪrçZDæTṛæ■ōçzSæđDīijLærTāçCēS;æŌēāLŪēālāĀAāzšēqæāšç■L'īijL'īijN  
ēCčāzLēçAæCšāIJlæĀğēC;äyLē;ç;āLrāEĒç;ōçZDēĀšāzēāGāāzŌāy■ārřēC;īijNāZāæ■d'īijNēfYæYrāzŪāzŪ

ēAāfāĒ■āLZāzZāy■āfĒēçAçZDæTṛæ■ōçzSæđDæLŪād'■āLū

æIJL'æŪūāĀZçlNāzRāSŸæCšæY;æSĒäyNīijNæđDēĀāyĀāzZāzūæšqæIJL'āfĒēçAçZDæTṛæ■ōçzSæđ

```
values = [x for x in sequence]
squares = [x*x for x in values]
```

āzšēōyēfZēGNçZDæCšæçTṛæYrēçŪāĒLārEäyĀāzZāĀijæTūēZEāLrāyĀäyāLŪēālāy■īijNçDūāRŌā;ççT  
äy■ēfĠīijNçñāyĀäyāLŪēālāōNāĒlæšqæIJL'āfĒēçAīijNārRrāzēçōĀā■TçZDāCRāyNēlçēfZæāūāEZīijZ

```
squares = [x*x for x in sequence]
```

äyŌæ■d'çZyāĒšīijNēfYēçAæšlæDRäyNēCčāzZārřPythonçZDāĒšāznæTṛæ■ōæIJzālŪēfĠāzŌāAṚæL'g  
æIJL'āzZāzZāzūæšqæIJL'ā;Lāç;çZDçRĒēğçæLŪāfāāzzPythonçZDāĒĒā■YælāādNīijNæzēçTl  
copy.deepcopy() āzNçszçZDāĠ;æTṛāĀC ēĀZāyāIJlēfZāzZāzççāAäy■æYrāRrāzēāŌzæŌL'ād'■āLūæS

ēōlēōž

āIJlāijYāNŪāzNāL■īijNæIJL'āfĒēçAāĒLçāTçl'ūāyNā;ççTlçZDçōŪæçTāĀC  
ēĀL'æNl'äyĀäyāād'■ælČāzēäyž O(n log n) çZDçōŪæçTṛæAærTā;āāŌzērCæTt'äyĀäyāād'■ælČāzēäyž  
O(n\*\*2) çZDçōŪæçTṛæL'ĀāyçælēçZDæĀğēC;æRṚā■ĠēçAād'gā;Ūād'ZāĀC

æçCædIJä;æğL'ā;Ūā;æçYæYrā;ŪēfZēāNāijYāNŪīijNēCčāzLērūāzŌæTṛ'ā;šēĀCēZSāĀC  
ä;IJāyZāyĀēLñāĠEāLZīijNāy■ēçAārřçlNāzRçZDærRāyĀäylēCīāLĒēç;āŌzāijYāNŪ,āZāyZēfZāzZāfōæTz  
ä;āāzTēřēäySæšlāzŌāijYāNŪāzğçTšæĀğēC;ççSūécLçZDāIJræŪzīijNærTāçCāĒēēCīā;ççŌrāĀC

ä;äçfYēçAæšlæDRā;ōārRāijYāNŪçZDçzSæđIJāĀCā;NāçCēĀCēZSāyNēlçāLZāzZāyĀäyā■ŪāEÿçZDæ

```
a = {
    'name' : 'AAPL',
    'shares' : 100,
    'price' : 534.22
}
```

```
b = dict(name='AAPL', shares=100, price=534.22)
```

āRŌēlcāyĀçg■āEŽæşTæŽt'çŏĀæt' AäyĀžZījLā;āy■ēIJĀēēAāIJlāĒşēTŏā■ŪāyLē;ŞāĒēāijTāRūrijLāā  
āy■ēēĠijNāēCædIJā;āārEēēZāy'd'āyīāzççāAçL'ĠæŏtēēZēāNāĀgēČ;æŧNērTārżærTæŪūrijNāijZāRŞçŌrā;ç  
dict() çŽDæŪzāijRāijZæĒcāzE3āĀ■āĀC çIJNāLrēēZāyīijNā;āæYrāy■æYrāEIJL'āĒşāLlāēLāL'ĀæIJL'ā;  
dict() çŽDāzççāAēČ;æZēæ■cæL'RçñnāyĀçg■āĀC āy■ād' şīijNēAīæYŌçŽDçlNāžRāSŸāRlāijZāĒşæşlāzŪ

āēCædIJā;āçŽDāijYāNŪēēAæşCærTē;ČénYīijNāEIJnēLCçŽDēēZāzŽçŏĀ■TæLĀæIJræzaēūşāy■āžEīij  
ā;NāēČīijNPyPyāūēēlNāYrPythonēgçēGLāZlçŽDāRēād' ŪāyĀçg■āŏđçŌrīijNāŏČāijZāLēædRā;āçŽDçlNāž  
āŏČæIJL'æŪāāZēČ;ædĀād' gçŽDæRRā■GæĀgēČ;īijNēĀZāyāRfāzēæŌēēLSCāzççāAçŽDēĀşāzēāĀC  
āy■ēēĠāRræČIJçŽDæYrīijNāLrāEžēēZæIJnāzēā;■çīijNPyPyēēYāy■ēēČ;āŏNāĒlæTræNĀPython3.  
āZāæ■d'īijNēēZāyīæYrā;āārEālēēIJĀēēAāŌžçāTçl'ūçŽDāĀCā;āēēYāRfāzēēĀČēZŞāyNNumbaāūēēlNīijN  
NumbaēYrāyĀāyīāIJlā;āā;ççTlēcĒēērāZlālēēĀL'æNl'PythonāĠ;æTŕēēZēāNāijYāNŪæŪūçŽDāLlāēĀAçijŪ  
ēēZāzŽāĠ;æTŕāijZā;ççTlLLVMēēçijŪērSæLŔæIJnāIJræIJzāZlçāAāĀCāŏČāRŔæāūāRfāzēædĀād' gçŽDæR  
ā;EæYrīijNēūşPyPyāyĀāūīijNāŏČārżāzŌPython 3çŽDæTræNĀçŌrāIJlēYāAIJçTŕZāIJlāŏđēlNēYūæŏtāĀC

æIJāāRŌēLŞāijTçTlJohn Ousterhoutēŕ'ēēĠçŽDērlā;IJāyžçzşār;īijZāĀIJæIJāāē;çŽDæĀgēČ;āijYāNŪ  
çŽt'āLŔā;āçIJşçŽDēIJĀēēAāijYāNŪçŽDæŪūāZāE■āŌzēĀČēZŞāŏČāĀCçāŏāflā;āçlNāžRæ■ççāŏçŽDēēRē

## 17 çññā■AāžTçñāīijŽCér■ēlĀæL'fāsT

æIJñçñāçlĀçIJijāžŌāzŌPythonēŏēēŪŏCāzççāAçŽDēŪŏēēYāĀCēŏyād'ŽPythonāEēç;ŏāžŞæYŕçTlCāEž  
ēŏēēŪŏCæYŕēŏl'PythonçŽDārżçŌræIJL'āžŞēēZēāNāžd'āžŞāyĀāyīēē■ēēAçŽDçzDæLŔēČlāLēāĀC  
ēēZāzşæYrāyĀāyīā;Şā;āēlçāyt'āžŌPython 2 āLŔ Python 3æL'fāsTāzççāAçŽDēŪŏēēYāĀC  
ēēZ;çDŪPythonæRRā;ZāžEāyĀāyīāzēæşZçŽDçijŪçlNAPIīijNāŏđēZēāyLæIJL'ā;Lād'ZæŪzæşTælēād'DçRĒ  
çZyærTērTāZ;ççZāGzārżāžŌærRāyĀāyīāRŕēČ;çŽDāūēāEūāLŪæLĀæIJŕçŽDēŕççēEāRČēĀČīijN  
æLŞāzLēGççTlçŽDæYræYŕēZEāy■āIJlāyĀāyīārRçL'ĠæŏtçŽDC++āzççāAīijNāzēāRlāyĀāžZæIJL'āzçēāæ.  
ēēZāyīçŽŏæāGæYræRRā;ZāyĀçşzāLŪçŽDçijŪçlNāēlæfēīijNæIJL'çzŔēlNçŽDçlNāžRāSŸāRfāzēæL'fāsTē

ēēZēĠNæYræLŞāznārEāIJlād'gēČlāLēççYçş■āy■āūēā;IJçŽDāzççāAīijŽ

```
/* sample.c */_method
#include <math.h>

/* Compute the greatest common divisor */
int gcd(int x, int y) {
    int g = y;
    while (x > 0) {
        g = x;
        x = y % x;
        y = g;
    }
    return g;
}

/* Test if (x0,y0) is in the Mandelbrot set or not */
int in_mandel(double x0, double y0, int n) {
```





## 17.1 15.1 ä;£çŦÍctypesèóÉúÓCäzççäA

### éúóécŸ

ä;äæIJL'äyÄäzŹCäG;æŦräüšçzŦècñçijŦèrSäLŦräĖšäznâzŞæLŦÜDLLäy■äÄCä;äâyŦæIJZäŦräzëä;£çŦÍçŹ  
èĀŦäy■çŦÍçijŦäĖŹéíäð'ŦçŹŹDCäzççäAæLŦŦä;£çŦÍçŹññäyLæŦŦæLŦ'äŦŦäüëäĖŦäÄC

### èğçäEşæŦzæqL

ärzäzŦéIJÄèæAèrŸçŦÍCäzççäAçŹDäyÄäzŦärŦçŹŹDèŦóécŸiijŦéÄŹäyŦä;£çŦÍPythonæäGäĖEäzŞäy■çŹ  
ctypes æláäIŦärsèüšäð'şäzĖäÄC èæAä;£çŦÍ ctypes iijŦä;äèĖŦäĖĖLèæAçäóäĖIä;äèæAèóéŦŦóçŹŹDCäzççä  
iijLäŦŦæäüçŹŹDæðüæðDäÄA■Ŧäð'gärŦäÄAçijŦèrSäŦÍç■L'iijLçŹŹDæşŦäyŦäĖšäznâzŞäy■äzĖäÄC  
äyŹäzĖè£ŹèäŦæIJñèLŸçŹŹDæijŦŦçd'zïijŦäŦGèó;ä;äæIJL'äyÄäyŦäĖšäznâzŞäŦ■ä■ŦäŦŦ  
libsamplē.so iijŦéŦŦéíççŹŹDäĖĖäöžärsæŦŦŦ5çnäazŦçz■éŦläĖEéCçæäüäÄC  
äŦèäð'Ŧè£ŸäŦGèó;è£ŹäyŦ libsample.so æŦŦGäzŦècñæŦŦç;óäŦŦä;■äzŦ sample.  
py æŦŦGäzŦçŹŹyärŦŦçŹŹDçŹŦä;Ŧäy■äzĖäÄC

èæAèóéŦŦóçŹŹDè£ŹäyŦäG;æŦŦäzŞiijŦä;äèæAäĖĖLæðDäzŹäyÄäyŦäŦéèçĖäóŸçŹŹDPythonæláäIŦiijŦäæCäyŦé

```
# sample.py
import ctypes
import os

# Try to locate the .so file in the same directory as this file
_file = 'libsamplē.so'
_path = os.path.join(* (os.path.split(__file__)[:-1] + (_file,)))
_mod = ctypes.cdll.LoadLibrary(_path)

# int gcd(int, int)
gcd = _mod.gcd
gcd.argtypes = (ctypes.c_int, ctypes.c_int)
gcd.restype = ctypes.c_int

# int in_mandel(double, double, int)
in_mandel = _mod.in_mandel
in_mandel.argtypes = (ctypes.c_double, ctypes.c_double, ctypes.c_
→int)
in_mandel.restype = ctypes.c_int

# int divide(int, int, int *)
_divide = _mod.divide
_divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
→POINTER(ctypes.c_int))
_divide.restype = ctypes.c_int

def divide(x, y):
```

```

    rem = ctypes.c_int()
    quot = _divide(x, y, rem)

    return quot, rem.value

# void avg(double *, int n)
# Define a special type for the 'double *' argument
class DoubleArrayType:
    def from_param(self, param):
        typename = type(param).__name__
        if hasattr(self, 'from_' + typename):
            return getattr(self, 'from_' + typename)(param)
        elif isinstance(param, ctypes.Array):
            return param
        else:
            raise TypeError("Can't convert %s" % typename)

    # Cast from array.array objects
    def from_array(self, param):
        if param.typecode != 'd':
            raise TypeError('must be an array of doubles')
        ptr, _ = param.buffer_info()
        return ctypes.cast(ptr, ctypes.POINTER(ctypes.c_double))

    # Cast from lists/tuples
    def from_list(self, param):
        val = ((ctypes.c_double)*len(param))(*param)
        return val

    from_tuple = from_list

    # Cast from a numpy array
    def from_ndarray(self, param):
        return param.ctypes.data_as(ctypes.POINTER(ctypes.c_double))

DoubleArray = DoubleArrayType()
_avg = _mod.avg
_avg.argtypes = (DoubleArray, ctypes.c_int)
_avg.restype = ctypes.c_double

def avg(values):
    return _avg(values, len(values))

# struct Point { }
class Point(ctypes.Structure):
    _fields_ = [('x', ctypes.c_double),
                ('y', ctypes.c_double)]

# double distance(Point *, Point *)
distance = _mod.distance

```

```
distance.argtypes = (ctypes.POINTER(Point), ctypes.POINTER(Point))
distance.restype = ctypes.c_double
```

æĈædIJäYÄÄLGæ■çäyÿijNä;äärſäRräzēāLæ; ;āzūā;ŁçTléGŃéIcāōZāZLçZĐCāG;æTřāžEāĀĆä;NāēĆ

```
>>> import sample
>>> sample.gcd(35,42)
7
>>> sample.in_mandel(0,0,500)
1
>>> sample.in_mandel(2.0,1.0,500)
0
>>> sample.divide(42,8)
(5, 2)
>>> sample.avg([1,2,3])
2.0
>>> p1 = sample.Point(1,2)
>>> p2 = sample.Point(4,5)
>>> sample.distance(p1,p2)
4.242640687119285
>>>
```

## ěőlěőž

æIJnārRèŁĆæIJL'ā;Łād'ŽāĀijā;ŮæŁŚāzněřęçzEęőlěőžçŽDāIJræŮzāĀĆ  
éęŮāĒLæYřāržāžŌCāŠŃPythonāzččāAäyĀęŮāL'ŠāŃĒçŽDēŮőćYÿijNāēĈædIJä;āāIJlä;ŁçTÍ  
ctypes æIěěőŁēŮőçijŮērSāRŌçŽĐCāzččāAÿijŃ éĆčāzŁéIJĀēęAçāōāŁēŁZāyĹāĒŚāznāzŠæT;āIJÍ  
sample.py æĹāāĹŮāRŃäyĀäyĹāIJræŮzāĀĆ äyĀçg■āRrēČ;æYřārEçTŠæĹRçŽD .  
so æŮGāzūæT;ç;őāIJlěęAä;ŁçTÍlāōČçŽĐPythonāzččāAāRŃäyĀäyŁçZōā;TäyNāĀĆ  
æŁŚāznāIJÍ recipeāĀTsample.py äy■ā;ŁçTÍ \_\_file\_\_  
āRŸēĜRæĹēæšççIJNāōČēćnāōL'ēčĒçŽDä;■ç;őÿijŃ çDūāRŌæđDēĀäyĀäyĹæŃGāRŠāRŃäyĀäyŁçZōā;Täy■  
libsamle.so æŮGāzūçŽDēŮrā;ĐāĀĆ

æĈædIJCāG;æTřāžŠęćnāōL'ēčĒāLrāĒŮāzŮāIJræŮzÿijŃéĆčāzŁā;äārſēęAāŁōæTřçZyāžTçŽDēŮrā;ĐāĀ  
æĈædIJCāG;æTřāžŠāIJlä;æIJzāŽĹāyŁēćnāōL'ēčĒāyžāyĀäyĹæāGāĜEāžŠāžEÿijŃ  
éĆčāzŁāRřāzēā;ŁçTÍ ctypes.util.find\_library() āG;æTřāĹēæšēæL'çÿijŽ

```
>>> from ctypes.util import find_library
>>> find_library('m')
'/usr/lib/libm.dylib'
>>> find_library('pthread')
'/usr/lib/libpthread.dylib'
>>> find_library('sample')
'/usr/local/lib/libsample.so'
>>>
```

äyĀæŮęä;ăçšēęAŠāžEĆāG;æTřāžŠçŽDä;■ç;őÿijŃéĆčāzŁārſāRřāzēāĈRāyŃéIcēŁZæāūā;ŁçTÍ  
ctypes.cdll.LoadLibrary() æĹēāLæ; ;āōČÿijŃ āĒŮäy■ \_path  
æYřāāGāĜEāžŠçŽDāĒlēŮrā;ĐÿijŽ

```
_mod = ctypes.cdll.LoadLibrary(_path)
```

āĠjæTřāžŠěcñāŁæj;āŘŌrijNā;æĪĀēęAçijŪāEŻāĠāyġlēr■āRēæĪēæŘŘāRŪčŁ'zāōŽčŽDčņēāRūāzūæNč  
ārśāČRāyNēĪcēŁZāyġāzččāAçŁ'ĠæōṭāyĀæāūijŽ

```
# int in_mandel(double, double, int)
in_mandel = _mod.in_mandel
in_mandel.argtypes = (ctypes.c_double, ctypes.c_double, ctypes.c_
    ↳int)
in_mandel.restype = ctypes.c_int
```

āĪġēŁŻæōṭāzččāAāy■ijN. argtypes āśđæĀġæYřāyĀāyġāĒČčzDrijNāNĒāRñāzEæšRāyġāĠjæTřčŽDč  
ēĀN .restype ārśæYřčŽyāžTčŽDēŁTāZđčśzādNāĀČ ctypes  
āōŽāzŁ'āžEāđ'ġēĠRčŽDčśzādNāržēsajijŁārTāēČc\_double, c\_int, c\_short, c\_floatč■L'ijL'ijN  
āžčēāġāžEāržāžTčŽDcæTřæ■ōčśzādNāĀČāēČæđĪā;āæČšēōġPythonēČ;āđ'šāijāēĀŠæ■čçāōčŽDāRCæTřčśz  
éČčāzŁēŁZāžŽčśzādNč■;āŘ■čŽDčzŠāōŽæYřā;ŁēĠ■ēęAçŽDāyĀæ■ēāĀČāēČæđĪā;āæšqæĪĪēŁZāzŁāAŽiij  
ēŁYāRřēČ;āijŽārijēĠræTř'āyġēġčēĠāZġēŁZčĪNāNČæŌŁāĀČ  
ā;ŁčTġctypesæĪĪ'āyĀāyġēžžčČēČčŽDāĪřæŪzæYřāŌščTščŽDcāžččāAā;ŁčTġčŽDæĪřēr■āRřēČ;ēũ\$Pytho  
divide() āĠjæTřæYřāyĀāyġā;Łāē;čŽDā;Nā■RrijNāōČēAŽēŁĠāyġāRČæTřēŽd'āžēāRēāyĀāyġāRČæTř  
ār;čōāēŁZæYřāyĀāyġā;ŁāyġēġAçŽDcæŁĀæĪřijNā;EæYřāĪĪPythonāy■ā■t'āy■čšēēAšæĀŌæāūāyĒæŽřčŽ  
āġNāēČrijNā;āāy■ēČ;āČRāyNēĪcēŁZæāūčōĀā■TčŽDāAŽiijŽ

```
>>> divide = _mod.divide
>>> divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
    ↳POINTER(ctypes.c_int))
>>> x = 0
>>> divide(10, 3, x)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ctypes.ArgumentError: argument 3: <class 'TypeError': expected LP_
    ↳c_int
instance instead of int
>>>
```

ārśčŌŪēŁZāyġēČjæ■čçāōčŽDāūēā;ĪijNāōČāijŽēŁĪāR■PythonāržāžŌæTř'æTřčŽDāy■āRřæŽt'æTřzāŌšā  
āržāžŌæūŁ'āŘŁāĪræNĠēŚŁčŽDāRCæTřrijNā;æĀŽāyġēĪĀēęAāĒŁæđDāžzāyĀāyġēŽyāžTčŽDctypesāržēsā

```
>>> x = ctypes.c_int()
>>> divide(10, 3, x)
3
>>> x.value
1
>>>
```

āĪġēŁŻēĠNrijNāyĀāyġ ctypes.c\_int āōđā;NēcñāŁZāžzāzūā;ĪāyžāyĀāyġæNĠēŚĪēcñāijāēŁZāŌzā  
ēũ\$æŽōēĀŽPythonæTř'ā;čāy■āRŊčŽDæYřrijNāyĀāyġ c\_int  
āržēsāqæYřāRřāžēēcñāŁōæTřčŽDāĀČ .value āśđæĀġāRřēcñčTġāēēēŌūāRŪæŁŪæŽt'æTřēŁZāyġāĀijāĀČ

āržāžŌēČčāžZāy■āČRPythončŽDcērČčTġrijNēĀŽāyġāRřāžēāEŻāyĀāyġārRčŽDāNĒēēĒāĠjæTřāĀČ  
ēŁŻēĠNrijNāēŁsāžnēōġ divide() āĠjæTřēĀŽēŁĠāēČčzDāēēēŁTāZđāyđ'āyġčzššæđĪijž

```
# int divide(int, int, int *)
_divide = _mod.divide
_divide.argtypes = (ctypes.c_int, ctypes.c_int, ctypes.
    ↳POINTER(ctypes.c_int))
_divide.restype = ctypes.c_int

def divide(x, y):
    rem = ctypes.c_int()
    quot = _divide(x, y, rem)
    return quot, rem.value
```

avg() āĢæTṙāRLæYṙāyĀäylæŨṙçŽĎæŇŚæLYāĀĆCāzččāAæIJ\$æIJZæŌēāRŪāLṚāyĀäylæŇĜēŚLāŠ  
 ä;EæYṙijŇāIJPythonäy■īijŇæLŠāznāēĒēāzēĀČēŽŚēēZāylēŪōēēYīijŽæTṙçzĎæYṙāTēijšāōCæYṙāyĀäylāL  
 ēēYæYṙ array ælāāIŪäy■çŽĎäyĀäylæTṙçzĎijšēēYæYṙāyĀäyl numpy  
 æTṙçzĎijšēēYæYṙērt æLĀæIJLēČ;æYṙijš āōēēŽĒäyLīijŇäyĀäylPythonāAIJæTṙçzĎāĀIæIJLāđ'Žçġ■ā;čā

DoubleArrayType āijTçd'žāzEæĀŌæāuāđ'DçRĒēēZçġ■æČĒāEṭāĀĆ  
 āIJlēēZāylçszāy■āōŽāzLāzEāyĀäylā■TāylæŪzæşT from\_param() āĀĆ  
 ēēZāylæŪzæşTçŽĎēġSēL'sæYṙæŌēāRŪāyĀäylā■TāylāRCæTṙçDūāRŌārEāĒūāRŠāyNē;ñæ■cāyžāyĀäylāRĪ  
 īijLæIJnā;Nāy■æYṙāyĀäyl ctypes.c\_double çŽĎæŇĜēŚLīijLāĀĆ  
 āIJ from\_param() äy■īijŇā;āāRṙāzēāAŽāzzā;Tā;āæČşāAŽçŽĎāžNāĀĆ  
 āRCæTṙçŽĎçszādNāR■ēēcāæRṚāRŪāĠzælēāzūēēcñçTlāžŌāLEāRSāLṚāyĀäylæZt'āĒūā;şçŽĎæŪzæşTāy■āŌ  
 ā;NāēČīijŇāēCāđIJāyĀäylāLŪēālēēcñāijāēĀŠēēĠælēīijŇēČčāzL typename āṙsæYṙ list  
 īijŇ çDūāRŌ from\_list æŪzæşTēēcñēṙČçTlāĀĆ

āṙzāžŌāLŪēālāSŇāĒČçzĎijŇfrom\_list æŪzæşTārEāĒūē;ñæ■cāyžāyĀäyl ctypes  
 çŽĎæTṙçzĎāržēšāāĀĆ ēēZāylçIJNāyLāŌzæIJLçČzāēĠæĀīijŇäyNēĪēāLŠāznā;ēçTlāyĀäylāzđ'āžSāijRā;N  
 ctypes æTṙçzĎijŽ

```
>>> nums = [1, 2, 3]
>>> a = (ctypes.c_double * len(nums))(*nums)
>>> a
<__main__.c_double_Array_3 object at 0x10069cd40>
>>> a[0]
1.0
>>> a[1]
2.0
>>> a[2]
3.0
>>>
```

āṙzāžŌæTṙçzĎāržēšāīijŇfrom\_array() æRṚāRŪāzTāsČçŽĎāEĒā■YæŇĜēŚLāzūārEāĒūē;ñæ■cāyž  
 ctypes æŇĜēŚLāržēšāāĀĆā;NāēČīijŽ

```
>>> import array
>>> a = array.array('d', [1, 2, 3])
>>> a
array('d', [1.0, 2.0, 3.0])
>>> ptr_ = a.buffer_info()
>>> ptr
4298687200
```

```
>>> ctypes.cast(ptr, ctypes.POINTER(ctypes.c_double))
<__main__.LP_c_double object at 0x10069cd40>
>>>
```

from\_ndarray() æijTçd'žāžEāržāžŌ numpy æTřčzDčŽDè;ñæ■ćæŠ■ā;IJāĀĆ  
 éĀŽèĚĜāōŽāzL DoubleArrayType çšžāžūāIJĪ avg() çšžādNç■;āR■āy■ā;ĚçTlāōČiijN  
 éĆčāžLēfŽāyġ;æTřāřsēČ;æŌēāRŪād'Žāyġāy■āR■NçŽDčšžæTřčzDè;ŠāĒēāžEiijŽ

```
>>> import sample
>>> sample.avg([1, 2, 3])
2.0
>>> sample.avg((1, 2, 3))
2.0
>>> import array
>>> sample.avg(array.array('d', [1, 2, 3]))
2.0
>>> import numpy
>>> sample.avg(numpy.array([1.0, 2.0, 3.0]))
2.0
>>>
```

æIJñèŁĆæIJĀāRŌāyĀéČlāLEāRŠā;āæijTçd'žāžEæĀŌæūād'DčŘEāyĀāyġčōĀā■TčŽDČčzŠæđDāĀĆ  
 āržāžŌčzŠæđDā;ŠiijNā;āāRlēIJĀēēAāČRāyNēlčēfZæāūčōĀā■TčŽDāōŽāzL'āyĀāyġčšžiiijNāNĒāRñčŽyāžTč

```
class Point(ctypes.Structure):
    _fields_ = [('x', ctypes.c_double),
                ('y', ctypes.c_double)]
```

āyĀæŪēçšžècñāōŽāzL'āRŌiijNā;āārsāRfāžēāIJlçšžādNç■;āR■āy■āLŪēĀĒæYřēIJĀēēAāōđā;NāNŪçzŠ

```
>>> p1 = sample.Point(1, 2)
>>> p2 = sample.Point(4, 5)
>>> p1.x
1.0
>>> p1.y
2.0
>>> sample.distance(p1, p2)
4.242640687119285
>>>
```

æIJĀāRŌāyĀāžZārRçŽDæRŘçd'žiiijŽāēČæđIJā;āæČšāIJlPythonāy■ēōēŪōāyĀāžZārRçŽDCāĜ;æTřiiij  
 ctypes æYřāyĀāyġā;LæIJL'çTlçŽDāĜ;æTřāžŠāĀĆ āřčōāāēČæ■d'iijNāēČæđIJā;āæČšēēAāŌžēōēŪōāyĀ  
 Swig (15.9èŁČāijŽēōšāĹr) æLŪ CythoniijL15.10èŁČiijL'āĀĆ

āržāžŌād'ġādNāžŠçŽDēōēŪōæIJL'āyġāyžēēAēŪōēčYiijNçTšāžŌctypesāzūāy■æYřāōNāĒlēĜlāLāNŪr  
 éĆčāžLā;āārsāfĒēāzēŁsēt'žād'ġēĜRæŪūēŪr'ælēçijŪāĒZæL'ĀæIJL'çŽDčšžādNç■;āR■iijNārsāČRā;Nā■Rāy  
 āēČæđIJāĜ;æTřāžŠād'šād'■āIČiijNā;āēfYā;ŪāŌzçijŪāĒZā;Lād'ŽārRçŽDāNĒēēēĀĜ;æTřāŠNæTřæNĀçšž  
 āRēād'ŪiijNēZd'ēlđā;āāūšçzRāōNāĒlçš;éĀŽāžEæL'ĀæIJL'āžTšĆçŽDCæŌēāRčçzEēŁČiijNāNĒæNñāEēā■  
 éĀŽāyāyĀāyġā;LārRçŽDāžčçāAçijžēZūāĀAēōēŪōēūLçTŃæLŪāĒūāzŪçšžāiijēTŽēřfārsēČ;ēōl'Pythonçl

ā;IJāyž ctypes çŽDāyĀāyġæZēāžçiiijNā;āēfYārřāžēēĀČēZŠāyNCFfiāĀČCFfiæRŘā;ZāžEā;Lād'Žç

ä;EæYřä;ŁçTÍCèř■æşTāzūæTřæŇAæŽt'ād'ŽénYčžğŽĐCäzččăAçşzăđŇăĂĆ  
ăĹrăEŻēfZăIĴnāzeäyžæ■ćijŇCFFIēfYæYřäyĂäyŁçŻyărzè;ČæŮřçŽĐăũēčĹŇijŇ  
ä;EæYřăŏČçŽĐăťAèąNāžææ■čĹJĹăfŇéĀşăyĹă■GăĂĆçTŽèGşèfYæIJĹăIJĹèŏĹèŏzăIJĹPythonăřEæĹēçŽĐçĹ

## 17.2 15.2 çŏĂă■TçŽĐCæL'ĹăsTăĹăĹİŮ

### éŮŏécY

ă;ăăČşăy■ă;ĹēĹăăĚŮăzŮăũēăĚŮijŇçŽt'æŎēă;ŁçTÍPythonçŽĐæL'ĹăsTăĹăĹİŮæĹēçijŮăEŻăyĂăžZçŏĂă■Tç

### èğčăEşæŮzæăĹ

ărzăžŎçŏĂă■TçŽĐCäzččăAĵijŇăđĐăžžăyĂäyĹèĠăŏŽăzĹæL'ĹăsTăĹăĹİŮæYřă;ĹăŏzæYşçŽĐăĂĆ  
ă;IJăyžçŇŇăyĂæ■ćijŇă;ăēIJăēçAçăŏăĹă;ăçŽĐCäzččăAæIJĹăyĂäyĹæ■čçăŏçŽĐăđ't'æŮGăžŮăĂĆă;ŇăçĆij

```
/* sample.h */

#include <math.h>

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

éĂžăyăæĹēŏşijŇēfZăyĹăđ't'æŮGăžŮēçĂărzăžTăyĂäyĹăũşçzŘēçŇă■TçŇŇçijŮērSēfĠçŽĐăžşăĂĆ  
æIJĹăžEēfZăžZijŇăyŇēĹæĹSăžŇăijTçđ'žăyŇçijŮăEŻæL'ĹăsTăĠă;æTřçŽĐăyĂäyĹçŏĂă■Tă;Ňă■ŘijŽ

```
#include "Python.h"
#include "sample.h"

/* int gcd(int, int) */
static PyObject *py_gcd(PyObject *self, PyObject *args) {
    int x, y, result;

    if (!PyArg_ParseTuple(args, "ii", &x, &y)) {
        return NULL;
    }
    result = gcd(x,y);
    return Py_BuildValue("i", result);
}

/* int in_mandel(double, double, int) */
```

```

static PyObject *py_in_mandel(PyObject *self, PyObject *args) {
    double x0, y0;
    int n;
    int result;

    if (!PyArg_ParseTuple(args, "ddi", &x0, &y0, &n)) {
        return NULL;
    }
    result = in_mandel(x0,y0,n);
    return Py_BuildValue("i", result);
}

/* int divide(int, int, int *) */
static PyObject *py_divide(PyObject *self, PyObject *args) {
    int a, b, quotient, remainder;
    if (!PyArg_ParseTuple(args, "ii", &a, &b)) {
        return NULL;
    }
    quotient = divide(a,b, &remainder);
    return Py_BuildValue("(ii)", quotient, remainder);
}

/* Module method table */
static PyMethodDef SampleMethods[] = {
    {"gcd", py_gcd, METH_VARARGS, "Greatest common divisor"},
    {"in_mandel", py_in_mandel, METH_VARARGS, "Mandelbrot test"},
    {"divide", py_divide, METH_VARARGS, "Integer division"},
    { NULL, NULL, 0, NULL}
};

/* Module structure */
static struct PyModuleDef samplemodule = {
    PyModuleDef_HEAD_INIT,

    "sample",          /* name of module */
    "A sample module", /* Doc string (may be NULL) */
    -1,                /* Size of per-interpreter state or -1 */
    SampleMethods       /* Method table */
};

/* Module initialization function */
PyMODINIT_FUNC
PyInit_sample(void) {
    return PyModule_Create(&samplemodule);
}

```

ẽĖAçzŠăõŽēfŽăyŁæLŕăŝTăĹăăİŮijŇăČŘăyŇéİcēfŽăăŭăĹŽăzžăyĂăyĹ      setup.py  
 æŮĜăzŭijŽ



```
# setup.py
from distutils.core import setup, Extension

setup(name='sample',
      ext_modules=[
          Extension('sample',
                  ['pysample.c'],
                  include_dirs = ['/some/dir'],
                  define_macros = [('FOO', '1')],
                  undef_macros = ['BAR'],
                  library_dirs = ['/usr/local/lib'],
                  libraries = ['sample']
                  )
      ])
)
```

äyžžæEæđĎāžžæIĲāçzŁçŽĎāĠæŦřāžŠiijŇŅāŔléIĲāçőĀă■ŦçŽĎäĲçŦŦí python3  
buildlib.py build\_ext --inplace āŠĳāzd'āŇšāŔŕiijŽ

```
bash % python3 setup.py build_ext --inplace
running build_ext
building 'sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
prototypes
-I/usr/local/include/python3.3m -c pysample.c
-o build/temp.macosx-10.6-x86_64-3.3/pysample.o
gcc -bundle -undefined dynamic_lookup
build/temp.macosx-10.6-x86_64-3.3/pysample.o \
-L/usr/local/lib -lsample -o sample.so
bash %
```

æCäyLæL' Åçd' žiiŋŇăőČăijŽăĹZăzzăyĂăyĹăŖ■ă■ŮăŖŋ sample.so  
çŽăĐăĚăžăŋăžăŠăĂăČă; ŠēcŋçijŮērŠăŖŮőijŇă;ăăŖsēc;ăŖĚăőČă;IJăyžăyĂăyĹăĹăĹăŮăŖijăĚăēēfŽăĹăăžĚijŽ

```
>>> import sample
>>> sample.gcd(35, 42)
7
>>> sample.in_mandel(0, 0, 500)
1
>>> sample.in_mandel(2.0, 1.0, 500)
0
>>> sample.divide(42, 8)
(5, 2)
>>>
```

æĈæđIĴä;äæŸřáIĴíWindowsæIĴžăZlăyŁéIĉăřlĕřŦēſZăžZæ■ēēld’iijŅăRřēĈ;aijŽēAĜăLřăŘĎĉg■ĎŔăĉĈ  
PythonĉŽĎăžŅēſZăLŭăLĒăRſēĂžăyŷä;ĤĉTlăžĒMicrosoft Visual StudioæIĕæđĎăžžăĂĈ  
ăyžăžĒēōl’ēſZăžZăL’l’ăſŦēĈ;æ■ăyŷăŭēă;IĴiijŅă;ăēIĴăĕēAă;ĤĉTlăRŅăăŭăLŨăĒiĵăōžĉŽĎăŭēăĒŭăIĕĉijŨē  
ăRĈēĂĈĉŽŷăžŦĉŽĎ PythonæŨĜăæĉ

## èòléôž

ǎIJǎřǐerTǎzzǎ;TǎL'NǎEŽǎL'ǎsTǎzNǎL'■īījNǎIJAǎē;èċ;ǎĒLǎRĈèĀĈǎyNPythonǎŪGǎçǎy■ċŽD  
ǎL'ǎsTǎŠNǎtNǎĒĒPythonēġċēĠǎŽĬ. PythonċŽDCǎL'ǎsTǎPIǎĬLǎd'gīījNǎIJlèfZéĠNǎT'ǎylǎŌžèōšēfǎ  
ǎy■ēfĠǎrzǎžŌǎIJAǎǎyǎfĈċŽDēĈlǎLēēfYǎYǎRǎřzēèólēôžǎyNċŽDǎĀĈ

ēēŪǎĒLīījNǎIJlǎL'ǎsTǎlǎǎIŪǎy■īījNǎ;ǎǎEŽċŽDǎĠ;ǎTǎřēĈ;ǎYǎRǎĈRǎyNéIċēfZǎǎūċŽDǎyǎǎylǎēZóēǎ

```
static PyObject *py_func(PyObject *self, PyObject *args) {  
    ...  
}
```

PyObject ǎYǎyǎǎylēĈ;ēǎlċd'žǎzzǎ;TPythonǎřzēšǎċŽDCǎTǎřǎ■ōċšǎdNǎĀĈ  
ǎIJǎyǎǎylēNŸċžġǎsĈéIċīījNǎyǎǎylǎL'ǎsTǎĠ;ǎTǎřǎsǎYǎyǎǎylǎŌēǎRŪǎyǎǎylPythonǎřzēšǎ  
īījLǎIJĬ PyObject \*argsǎy■īījLǎĒĈċŽDǎžūēfTǎŽDǎyǎǎylǎŪrPythonǎřzēšǎċŽDCǎĠ;ǎTǎřǎĀĈ  
ǎĠ;ǎTǎřċŽD self ǎRĈǎTǎřǎřzǎžŌċōǎǎ■TǎċŽDǎL'ǎsTǎĠ;ǎTǎřǎsǎǎIJL'ēċnǎ;fċTǎlǎLīījN  
ǎy■ēfĠǎēĈǎdIJǎ;ǎǎĈšǎōžǎžL'ǎŪrċŽDċšǎēLŪēǎĒǎYǎRǎCǎy■ċŽDǎřzēšǎċšǎdNċŽDēřǎřsēĈ;ǎf'ǎyǎLċTǎlǎIJ  
ēĈǎžL self ǎřsēĈ;ǎīījTǎTǎlēĈǎyǎlǎōdǎ;NǎžEǎĀĈ

PyArg\_ParseTuple() ǎĠ;ǎTǎřēċnċTǎlǎēǎřEPythonǎy■ċŽDǎǎīē;ǎǎēĈǎLǎRĈǎyǎǎřzǎžTēǎlċd'žǎĀĈ  
ǎōĈǎRŪǎyǎǎylǎēNĠǎōžē;ŠǎĒēǎǎīǎīRċŽDǎǎīǎīRǎNŪǎ■Ūċņǎyšǎ;IJǎyžē;ŠǎĒēīījNǎřTǎēĈǎIJǎǎlǎ  
ǎRŪǎǎūēfYǎIJL'ǎ■YǎT'ē;ǎǎēĈǎRŌċžŠǎdIJċŽDCǎRŸēĠRċŽDǎIJǎlǎǎĀĈ  
ǎēĈǎdIJē;ŠǎĒēċŽDǎǎīǎy■ǎNžēĒēfZǎylǎēǎīǎīRǎNŪǎ■ŪċņǎyšīījNǎřǎījZǎLZǎĠzǎyǎǎylǎījCǎyǎžūēfT  
ēǎŽēfĠǎēĈǎēšēǎžūēfTǎŽDNULLīījNǎyǎǎylǎRǎLēĀĈċŽDǎījCǎyǎījZǎIJlǎēĈTǎlǎžċǎǎy■ēċnǎēLZǎĠzǎĀĈ

Py\_BuildValue() ǎĠ;ǎTǎřēċnċTǎlǎēǎžǎē■ōCǎTǎřǎ■ōċšǎdNǎLZǎžžPythonǎřzēšǎĀĈ  
ǎōĈǎRŪǎyǎǎylǎēǎīǎīRǎNŪǎ■ŪċņǎyšǎēlǎēNĠǎōžēIJšǎIJZċšǎdNǎĀĈ  
ǎIJlǎL'ǎsTǎĠ;ǎTǎřǎy■īījNǎōĈēċnċTǎlǎēēfTǎŽdċžŠǎdIJċžPythonǎĀĈ  
Py\_BuildValue() ċŽDǎyǎǎylċL'ǎǎǎYǎřǎōĈēĈ;ǎdDǎžǎZt'ǎLǎād'■ǎIĈċŽDǎřzēšǎċšǎdNīījNǎřTǎēĈ  
ǎIJpy\_divide() ǎžċǎǎy■īījNǎyǎǎylǎ;Nǎ■RǎījTċd'žǎžEǎǎŌǎǎūēfTǎŽDǎyǎǎylǎēĈċŽDǎĀĈǎy■ēfĠ

```
return Py_BuildValue("i", 34); // Return an integer  
return Py_BuildValue("d", 3.4); // Return a double  
return Py_BuildValue("s", "Hello"); // Null-terminated UTF-8 string  
return Py_BuildValue("(ii)", 3, 4); // Tuple (3, 4)
```

ǎIJlǎL'ǎsTǎlǎǎIŪǎžTēĈīījNǎ;ǎǎījZǎRŠċŌrǎyǎǎylǎĠ;ǎTǎřǎlīījNǎřTǎēĈǎIJNēLĈǎy■ċŽD  
SampleMethodsēǎlǎĀĈēfZǎylēǎlǎRǎřzēǎLŪǎĠžCǎĠ;ǎTǎřǎǎPythonǎy■ǎ;fċTǎlċŽDǎRǎ■ŪǎǎǎǎŪGǎē  
ǎL'ǎǎIJL'ǎlǎǎIŪēĈ;ēIJǎēēǎǎNĠǎōžēfZǎylēǎlīījNǎžǎyǎžǎōĈǎIJlǎlǎǎIŪǎLǎġNǎNŪǎŪūēēǎēċnǎ;fċTǎlǎL

ǎIJAǎRŌĈŽDǎĠ;ǎTǎPyInit\_sample() ǎYǎřǎlǎǎIŪǎLǎġNǎNŪǎĠ;ǎTǎřīījNǎ;EēřēǎlǎǎIŪċņǎyǎǎyl  
ēfZǎylǎĠ;ǎTǎřċŽDǎyžēēǎǎūēǎ;IJǎYǎřǎIJlèġċēĠǎŽlǎy■ēšǎlǎēNǎlǎǎIŪǎřzēšǎĀĈ

ǎIJAǎRŌǎyǎǎylēēǎĈĈēIJǎēēǎǎRǎǎĠžǎēlēīījNǎ;fċTǎCǎĠ;ǎTǎřǎlǎēL'ǎsTǎPythonēēǎēĀĈēŽSċŽDǎž  
īījLǎōdēZēǎylīījNĈ APIǎNēǎRǎžEēūēēfĠ500ǎylǎĠ;ǎTǎřīījL'ǎĀĈǎ;ǎžTēēēǎǎēIJNēLĈǎ;ŠǎǎžǎYǎyǎy  
ǎŽt'ǎd'ŽēNŸċžǎēEǎōžīījNǎRǎřzēĈIJNĈIJN PyArg\_ParseTuple() ǎŠN  
Py\_BuildValue() ǎĠ;ǎTǎřċŽDǎŪGǎçīījN ċDŪǎRŌēfZǎyǎǎēǎL'ǎsTǎījǎǎĀĈ

## 17.3 15.3 çijÚâĖZæL'ŕâšŦâĜ;æŦŕæŠ■ä;IJæŦŕçzĎiijŊâŖŕèĈ;æŸŕèċnarrayæŭaŭæĹŮçšzäijijl

### éŮóéćŸ

ä;äæĈšçijŮâĖZäyÄäyŦCæL'ŕâšŦâĜ;æŦŕæŭæŠ■ä;IJæŦŕçzĎiijŊâŖŕèĈ;æŸŕèċnarrayæŭaŭæĹŮçšzäijijl  
äy■æŦĜiijŊä;äæĈšèŕŕ'ä;äçŽĎâĜ;æŦŕæŽŦ'âĹæĖÄŽçŦŦiijŊèÄŊäy■æŸŕéŠĹâræšŖäyŦçL'žâŕŽçŽĎžšæL'ÄçŦ

### èĝĉâĖşæŮzæaĹ

äyžâZĖèĈ;èŕŕ'æŖŕæŕŮâšŊâĎ'ĎçŖĖæŦŕçzĎâĖŮæIJL'âŖŕçğžæd'■æÄĝiijŊä;äæIJÄèçAä;ŦçŦŦâĹŖ  
*Buffer Protocol* . äyŊéĬæŸŕäyÄäyŦæL'ŊâĖZçŽĎCæL'ŕâšŦâĜ;æŦŕä;Ŋâ■ŖiijŊ  
çŦŦæŭæŖŕæŕŮâŦŕçzĎâŦŕæ■ŕâžŭèŖĈçŦŦæIJŋçŋâiijÄçŖĜéĈŦâĹĖçŽĎ avg(double  
\*buf, int len) âĜ;æŦŕiijŽ

```
/* Call double avg(double *, int) */
static PyObject *py_avg(PyObject *self, PyObject *args) {
    PyObject *bufobj;
    Py_buffer view;
    double result;
    /* Get the passed Python object */
    if (!PyArg_ParseTuple(args, "O", &bufobj)) {
        return NULL;
    }

    /* Attempt to extract buffer information from it */

    if (PyObject_GetBuffer(bufobj, &view,
        PyBUF_ANY_CONTIGUOUS | PyBUF_FORMAT) == -1) {
        return NULL;
    }

    if (view.ndim != 1) {
        PyErr_SetString(PyExc_TypeError, "Expected a 1-dimensional array
↪");
        PyBuffer_Release(&view);
        return NULL;
    }

    /* Check the type of items in the array */
    if (strcmp(view.format, "d") != 0) {
        PyErr_SetString(PyExc_TypeError, "Expected an array of doubles
↪");
        PyBuffer_Release(&view);
        return NULL;
    }

    /* Pass the raw buffer and size to the C function */
    result = avg(view.buf, view.shape[0]);
```

```
/* Indicate we're done working with the buffer */
PyBuffer_Release(&view);
return Py_BuildValue("d", result);
}
```

äyÑéíçæĹŚäzñæijŦçd'žäyÑèŁŻäyĹæĹ'āśŦāĠ;æŦræŸræĈä;Ŧāũëä;ĬçŽĎiižŽ

```
>>> import array
>>> avg(array.array('d', [1, 2, 3]))
2.0
>>> import numpy
>>> avg(numpy.array([1.0, 2.0, 3.0]))
2.0
>>> avg([1, 2, 3])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'list' does not support the buffer interface
>>> avg(b'Hello')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Expected an array of doubles
>>> a = numpy.array([[1., 2., 3.], [4., 5., 6.]])
>>> avg(a[:, 2])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: ndarray is not contiguous
>>> sample.avg(a)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Expected a 1-dimensional array
>>> sample.avg(a[0])

2.0
>>>
```

## ëóíëőž

ārEäyÄäyĹæŦŦçžĎāržèšāijāçžŽCāĠ;æŦrāŦŦèĈ;æŸŦäyÄäyĹæĹ'āśŦāĠ;æŦrāŦŽçŽĎæĬĬÄäyÿèġAçŽĎä  
ā;Ĺād'ŽPythonāžŦçŦĬĬNāžŦiiŦNāžŦāž;āĈŦād'ĎçŦŦēāĹŦçġŦā■ēōāçŦŦiiŦŦēĈ;æŸŦāšžāžŦŦēŦŸæĀġèĈ;çŽĎ  
éĀŽèŁĠçijŦāĒžèĈ;æŦŦāŦŦāžŦāš■ä;ĬæŦŦçžĎçŽĎäžççāĬiiŦŦä;āāŦŦäžèçijŦāĒžā;Ĺāē;çŽĎāĒijāōžèŁŽāžž  
èĀŦäy■æŸŦāŦŦèĈ;āĒijāōžä;äēĠāũšçŽĎäžççāĬāĀĈ

äžççāAçŽĎāĒšéŦŦçĈžāĬĬäžŦ PyBuffer\_GetBuffer() āĠ;æŦŦāĀĈ  
çžŽāōžäyÄäyĹäžæĎŦçŽĎPythonāržèšāijŦNāōĈāijžèŦŦçĬĬāāŦžèŦŦāŦŦāžŦāšĈāĒēā■ŸāŦæĀŦiiŦŦāōĈçŦāā  
1. āijāçžŽ PyBuffer\_GetBuffer() çŽĎçĹ'žæŦŦæāĠāŦŦçžŽāĠžāžĒæĹ'ĀéĬĬAçŽĎāĒēā■ŸçijŦāĒšçšžāç  
ä;ŦāēĈiiŦŦPyBUF\_ANY\_CONTIGUOUS èāĬd'žæŸŦäyÄäyĹæĀŦçžççŽĎāĒēā■ŸāŦžāššāĀĈ

āržāžŦæŦŦçžĎāĬāā■ŦèĹĈā■ŦçŦäyšāŦŦāŦŦäžŦçšžāijijāržèšāēĀŦŦēĬĬiiŦŦäyÄäyĹ  
Py\_buffer çžšæĎĎä;ŦāŦēāŦŦāžĒæĹ'ĀēĬĬĹ'āžŦāšĈāĒēā■ŸçŽĎāŦæĀŦāĀĈ

ãõÇãÑĖãŖñäyÄäyĭæÑĠãŖŚãĖĖã■ŸãĬŖãĬÄãÄAãd' ġãŖŖãÄAãĖČçŧ' äãd' ġãŖŖãÄAãæãĭãĭŖãŚÑãĖüãžŮčžĖĖĤ

```
typedef struct bufferinfo {
    void *buf; /* Pointer to buffer memory */
    PyObject *obj; /* Python object that is the owner */
    Py_ssize_t len; /* Total size in bytes */
    Py_ssize_t itemsize; /* Size in bytes of a single item */
    int readonly; /* Read-only access flag */
    int ndim; /* Number of dimensions */
    char *format; /* struct code of a single item */
    Py_ssize_t *shape; /* Array containing dimensions */
    Py_ssize_t *strides; /* Array containing strides */
    Py_ssize_t *suboffsets; /* Array containing suboffsets */
} Py_buffer;
```

æĬñĖĤĆäy■ĭĭñĖĤŚãžñãŖĭãĖŞãşĭæŌĖãŖŮäyÄäyĭãŖÑçşĭăžĕæŧõçĆžæŧŖæŧŖçžĎäĭĬäyžãŖĆæŧŖãĤĤ  
ĖĕAãĕÄãŞĖãĖČçŧ' äæŸŖãŖĕæŸŖäyÄäyĭãŖÑçşĭăžĕæŧõçĆžæŧŖĭĭñãŖĭĖĬÄĖĤÑĖŖA  
format äşđæÄġæŸŖäy■æŸŖã■ŮçñĕäyşãĬĭđãĬĭ. ĖĤŽäyĭãžşæŸŖ struct  
æĭããĭŮçŧĭæĭĕĕĭĭŮçãAãžÑĖĤŽãĬŮæŧŖæ■õçŽĎãĤĤ ĖÄŽäyĭæĭĖĖõşĭĭñformat  
ãŖŖãžĕæŸŖãžžãĭŧãĖĭãőž struct æĭããĭŮçŽĎæãĭãĭŖãŖŮŮã■Ůçñĕäyşĭĭñ  
ăžŮäyŧãĕĤæĎĬæŧŖçžĎãÑĖãŖñãžĖČçžŞæĎĎçŽĎĖŖĭãõČãŖŖãžĕãÑĖãŖñãĎ' ŽäyĭãĬĭãĤĤ  
äyÄæŮĕæĤŚãžñãŮşçžŖĕãĖãĖŽăžĖãžŧãşĤçŽĎçĭĭşã■ŸãŖñžãĖæAŖĭĭñĖĕČãŖĭĖĬÄĖĖAçõÄã■ŧçŽĎãŖĖãõČãĭĭ  
ãĖĎĖŽĖäyĭĭĭñĖĤŚãžñäy■ãĤĖãÑĖãĤĤæŸŖæĤŌæăŮçŽĎæŧŖçžĎçşžãĎÑãĤŮĖÄĖãĖČæŸŖĕĕñãžĂãžĤăžşãĤŽ  
ĖĤŽăžşæŸŖäyžãžĂãžĤĖĤŽäyĭãĤĤ;æŧŖĕČĭãĖĭãőž array æĭããĭŮãžşĖČĭãĖĭãőž numpy  
æĭããĭŮäy■çŽĎæŧŖçžĎãžĖãĤĤ

ãĬĭĖĤŧãŽđæĬÄçžĤĤçžŞæĎĬäžŖãĤĭĭññãžŧãşĤçŽĎçĭĭşãĖŞãŖñžĖġĖãžĭ;ăĤĖĖãžãĭ;Ĥçŧĭ  
PyBuffer\_Release() ĖĠĤæŧŧ;ĖŌĤãĤĤ äžŖãĤĤÄžĖĖĖAĖĤŽäyÄæ■ĖæŸŖäyžãžĖĖČĭ;æ■ĕçãõçŽĎçõçŖĖĖĤ

ãŖŖãæŮĭĭĭñĖĤĬñĖĤĆãžşăžĖãžĖãŖĭæŸŖæĭĭŧçđ' žăžĖæŌĖãŖŮæŧŖçžĎçŽĎäyÄäyĭãŖŖçŽĎãžĕçãAçĤĤĤãĖŮ  
ãĕĤæĎĬäĭ;çĬĬşçŽĎĖĖAãĎ' ĎçŖĖæŧŖçžĎĭĭññãĭ;ããŖŖĖČĭ;ãĭĭžĕĕŖãĤŖãĎ' Žçžŧ' æŧŖæ■õãAãĎ' ġæŧŖæ■õãAãäy■ãĭ  
ĖČçãžĤŖãŖşãĭŮãŌžã■ĕæŽŧ' ĖñŸçžġçŽĎäyĬĖĖĤăžĖãĤĤ;ăĖĬÄĖĖAãŖĤĖĖČãĖŮŸæŮžæŮĠæãĕăĖĖĖŮãŖŮæŽŧ

ãĕĤæĎĬäĭ;ăĖĬÄĖĖAçĭĭŮãĖŽæŮĤŖãŖĤãĤŖæŧŖçžĎãĎ' ĎçŖĖçŽĎãĎ' ŽäyĭæĤĤ' äşŧĭĭñĖĕČçãžĤĖÄŽĖĤĠCytho

## 17.4 15.4 ãĬĬĤĤæĤĤ'äşŧæĭããĭŮäy■æŞ■ãĭĬĖŽŖãĭçæŖĠĖĤĤ

### ĖŮĖĖČŸ

ăĭ;ăæĬĤĤ'äyÄäyĭæĤĤ'äşŧæĭããĭŮĖĬÄĖĖAãĎ' ĎçŖĖČçžŞæĎĎäĭ;Şäy■çŽĎæŖĠĖĤĤĭĭñ  
ăĭĖæŸŖãĭ;ããŖĤäy■æČşæŽŧ' ĖĬşçžŞæĎĎäĭ;Şäy■ăžžãĭŧãĖĖĖČĭçžĖĤĤçžžPythonãĤĤ

### ĖġçãĖŞãŮžæãĤĤ

ĖŽŖãĭççžŞæĎĎäĭ;ŞãŖŖãžĖãĭĤăĖžæŸŞçŽĎĖÄŽĖĤĤġãŖĖãĖČãžñãÑĖĖĖĖãĬĖĖČŮãžĤŖãŖžĖşãäy■æĤĖãĎ' ĎçŖĖ  
ĖĤĖĖŽŚãĤŚãžñãĭŖã■ŖãžĕçãAãy■çŽĎäyŖãĤŮĤãžĕçãAçĤĤĤãĖŮĭĭñŽ

```
typedef struct Point {
    double x,y;
```

```

} Point;

extern double distance(Point *p1, Point *p2);

```

äÿÑéÍæŸřäÿÄäÿlä;£çŤíëČúâŽĽăÑĚëčĚPointçzŞæđĎä;ŞăŠŇ distance()  
 âĢ;æŤřçŽĎæĽŤřăŤäzčçăĂăđă;ŇijŽ

```

/* Destructor function for points */
static void del_Point(PyObject *obj) {
    free(PyCapsule_GetPointer(obj, "Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
    return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int must_free) {
    return PyCapsule_New(p, "Point", must_free ? del_Point : NULL);
}

/* Create a new Point object */
static PyObject *py_Point(PyObject *self, PyObject *args) {

    Point *p;
    double x,y;
    if (!PyArg_ParseTuple(args,"dd",&x,&y)) {
        return NULL;
    }
    p = (Point *) malloc(sizeof(Point));
    p->x = x;
    p->y = y;
    return PyPoint_FromPoint(p, 1);
}

static PyObject *py_distance(PyObject *self, PyObject *args) {
    Point *p1, *p2;
    PyObject *py_p1, *py_p2;
    double result;

    if (!PyArg_ParseTuple(args,"OO",&py_p1, &py_p2)) {
        return NULL;
    }
    if (!(p1 = PyPoint_AsPoint(py_p1))) {
        return NULL;
    }
    if (!(p2 = PyPoint_AsPoint(py_p2))) {
        return NULL;
    }
    result = distance(p1,p2);
}

```

```
    return Py_BuildValue("d", result);
}
```

ǎĲPythonäy■ǎRřžěǎČŘäyNéícèŁZæăüælä;ŁçŤlèŁZăŽZǎĞ;æŢriijŽ

```
>>> import sample
>>> p1 = sample.Point(2,3)
>>> p2 = sample.Point(4,5)
>>> p1
<capsule object "Point" at 0x1004ea330>
>>> p2
<capsule object "Point" at 0x1005d1db0>
>>> sample.distance(p1,p2)
2.8284271247461903
>>>
```

èóìèőž

ěČůãŽŁăŠŇCæŃĜēŚĹçśzajijjāĂĆăIJIăEĚēČlīijNǎoČazněŌũăRŮäyÄäylēĂžčTlīæŇĜēŚĹăŠŇäyÄäylaŘ  
PyCapsule\_New() äĞ;æTṙăꞤLăôžæYšÇŻDēcñăĹZăżzăĂĆ  
ăŘead' ŮiijNăyÄäylaŘfēĂĹ'çŽDæđŘæđĐăĞ;æTrēČ;ēcncžSăoŽăĹrēCűăŽĹăyŁiiJNcTlīēăăIJlēCűăŽĹărzesăć

PyCapsule\_GetPointer()

æIñēŁĆäy■īijNāyĀārzaūēāĒūāĜ;æȚrāĀtāĀȚ PyPoint\_FromPoint()  
 āŠŃ PyPoint\_AsPoint() ēćńċȚlāĒēāŁZāzżāŠŃāzŌēĆūāZŁārżēśāy■æŔŔāŔŪ-  
 Pointāōdā;ŃāĀĆ āIĴlāzzā;ȚāēLȚāśȚāĜ;æȚrāy■īijNāēŁSāznāijZā;ĲȚȚlēfZāzZāĜ;æȚrēĀNāy■æȚŕĲZt'æŌēā;  
 ēŁZġ■ēō;ēōāj;Ĳā;ŪāēŁSāznāŔŕfāzēā;ŁāōzāYŠĲZDāȚŔārzāŔēāĒārZPointāȚȚāyNĲZDāNĒēēĲZDāēZt'æȚzā;  
 ā;NāēĆīijNāēĆāēDĴā;āāēŚāōZā;ĲȚȚlāŔēād' ŪāyĀāyĲēĆūāZŁāzēīijNēĆCāzŁĀŔĲēĴāēēAēāZt'æȚzēēŁāyēd' āyĲā

```

    ħřzāzŎēĈŭāZĹāřzēsāyĀäyĹēZĭ;ĉĆzāĬĴāzŎāđĈāĬĴ;āZđæTŭāŠŇāĖĚā■ŸĉŏaçŘĚāĂĈ
PyPoint_FromPoint()                āĜ;æTřæŎēāRŮäyĀäyĹ                must_free
āŘĆæTřĭĭjŇ                ĉTĹæĹēāŇĜāŏZā;ŠēĈŭāZĹēĉnéTĀæřAæŮŭāzTřāsĆPoint                *
ĉzŠæđDā;ŠæŸřāŘēāzTēřēēĉnāZđæTŭāĂĈ āĬĴæšŘāzZĆāzĉĉāAäy■ĭĭjŇā;ŠāsđēŮŏēćŸēĀZāyŷā;ĹēZĭēĉnāđ'Ĭ
ĉĹNāzŘāŚŸāŘřāzēā;ĤĉTĹ extra āŘĆæTřæĹēāŎĝāĹŭĭĭjŇēĀŇäy■æŸřā■TæŮzéĹĉZĎāĖšāŏZāđĈāĬĴ;āZđæT
ēēAæšĹæĎŘĉZĎæŸřāŚŇĉŎřæĬĴēĈŭāZĹæĬĴāĖšĉZĎæđŘæđĎāZĹēĈ;ā;ĤĉTĹ
PyCapsule_SetDestructor() āĜ;æTřæĹēāZt'æTzāĂĈ

```

ārzāzŌæūL̥āRŁāLřčzŠædDä;ŞçŽĐCăzčcǎAèĀÑēlĀīijNā;ŁçTlēČúāZŁæYřāyĀāylærTē; ČāRŁçŘčēŽĐ  
 āčNāçCīijNāIJL̥æUūāĀZā;āāzūāy■ĀšşāłČēŽT' éIJSçzŠædDä;ŞçŽĐāEĒēČlāfæAřāŁŪēĀĒārĒāĒūē;nā■cā  
 éĀžēfGā;ŁçTlēČúāZŁīijNā;āāRřāzēāIJlāōČāyŁēlčāT;āyĀāylē;zéGRčzğčŽĐāÑēēcĒāZlīijNčĐūāRŌārĒāō

## 17.5 15.5 äZÖæL'ŕásTælaaIÜäy■áoŽázL'áŠNárijáGžCçŽĐAPI

### éÜóécŸ

ä;äæIJL'äyÄäyI CæL'ŕásTælaaIÜäy■NáIJlâEĚéČláoŽázL'ázEā;Lād'ŽæIJL'çTlçŽĐāG;æTrijNä;äæČšārEā  
APIā;ŽāĚüāzÜāIJŕæŰzā;£çTlāĀČ ä;äæČšāIJlâĚüāzÜæL'ŕásTælaaIÜäy■ā;£çTlē£ŽázZāG;æTrijNä;EæŸŕāy  
āzūāyTēĀŽē£GCçijŰērSāŽl/éS;æŌēāŽlāĚāAŽçIJNäyLāŌzçL'zāLnad'■āIČrijLæLŰēĀĚäy■āRŕēČ;āAŽāL

### èğčāEşæŰzæaĹ

æIJnèLČäyžèeAéÜóécŸæŸŕāeČā;Tād'DçRĚ15.4ārRèLČäy■æRŘāLŕçŽĐPointāržèšāāĀČāzTçzEāZđäy

```
/* Destructor function for points */
static void del_Point(PyObject *obj) {

    free(PyCapsule_GetPointer(obj, "Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
    return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int must_free) {
    return PyCapsule_New(p, "Point", must_free ? del_Point : NULL);
}
```

çŌŕāIJlçŽĐéÜóécŸæŸŕāeĀŌæūāŕE PyPoint\_AsPoint()  
āŠN Point\_FromPoint() āG;æTŕā;IJäyŽAPIāŕijāGžrijN  
è£ŽæāūāĚüāzÜæL'ŕásTælaaIÜēČ;ā;£çTlāzūeS;æŌēāōČāznrijNæŕTāeČāeČæđIJā;äæIJL'āĚüāzÜæL'ŕásTāzš  
èeAèğčāEşè£ŽäyIeÜóécŸrijNēeŰāĚLēeAäyž sample æL'ŕásTāEŽäyIæŰŕçŽĐād't æŰGāzūāŔ■āRn  
pysample.h rijNāeČäyNrijŽ

```
/* pysample.h */
#include "Python.h"
#include "sample.h"
#ifdef __cplusplus
extern "C" {
#endif

/* Public API Table */
typedef struct {
    Point *(*aspoint)(PyObject *);
    PyObject *(*frompoint)(Point *, int);
} _PointAPIMethods;

#ifdef PYSAMPLE_MODULE
/* Method table in external module */
```



```

static _PointAPIMethods *_point_api = 0;

/* Import the API table from sample */
static int import_sample(void) {
    _point_api = (_PointAPIMethods *) PyCapsule_Import("sample._point_
↪api", 0);
    return (_point_api != NULL) ? 1 : 0;
}

/* Macros to implement the programming interface */
#define PyPoint_AsPoint(obj) (_point_api->aspoint)(obj)
#define PyPoint_FromPoint(obj) (_point_api->frompoint)(obj)
#endif

#ifdef __cplusplus
}
#endif

```

ěfŽéGÑæIJÄéG■ēçAçŽDěČlálEæYřáGjæTřæŇGéŠĹeál \_PointAPIMethods .  
 āóČāijŽāIJlārijāGžælqāIŮæŮüēćnāLiāgŇāŇŮrijŇčDūāRŌārijāĚēælqāIŮæŮüēćnæšēæL'ǎLřāĀĆ  
 äfōæŤzāŎšāgŇčŽDæLŤāsŤælqāIŮæIēāqāāĚĚēāæāijāzūārĚāóČāČRäyŇéíćēfŽæāūārijāGžiiž

```

/* pysample.c */

#include "Python.h"
#define PYSAMPLE_MODULE
#include "pysample.h"

...
/* Destructor function for points */
static void del_Point(PyObject *obj) {
    printf("Deleting point\n");
    free(PyCapsule_GetPointer(obj, "Point"));
}

/* Utility functions */
static Point *PyPoint_AsPoint(PyObject *obj) {
    return (Point *) PyCapsule_GetPointer(obj, "Point");
}

static PyObject *PyPoint_FromPoint(Point *p, int free) {
    return PyCapsule_New(p, "Point", free ? del_Point : NULL);
}

static _PointAPIMethods _point_api = {
    PyPoint_AsPoint,
    PyPoint_FromPoint
};
...

```

```

/* Module initialization function */
PyMODINIT_FUNC
PyInit_sample(void) {
    PyObject *m;
    PyObject *py_point_api;

    m = PyModule_Create(&samplemodule);
    if (m == NULL)
        return NULL;

    /* Add the Point C API functions */
    py_point_api = PyCapsule_New((void *) &_amp;_point_api, "sample._point_
    ↪api", NULL);
    if (py_point_api) {
        PyModule_AddObject(m, "_point_api", py_point_api);
    }
    return m;
}

```

æIJĀăŘŮijŇăyŇéĬæŸřăyĂăyĭæŮřčŽĎæLŕăśŢăĭăĭŮăĭŇă■ŘiijŇčŢĭăĭăLăèĭăżŭăĭĭçŢĭăĭZăžZAPIă

```

/* ptexample.c */

/* Include the header associated with the other module */
#include "pysample.h"

/* An extension function that uses the exported API */
static PyObject *print_point(PyObject *self, PyObject *args) {
    PyObject *obj;
    Point *p;
    if (!PyArg_ParseTuple(args, "O", &obj)) {
        return NULL;
    }

    /* Note: This is defined in a different module */
    p = PyPoint_AsPoint(obj);
    if (!p) {
        return NULL;
    }
    printf("%f %f\n", p->x, p->y);
    return Py_BuildValue("");
}

static PyMethodDef PtExampleMethods[] = {
    {"print_point", print_point, METH_VARARGS, "output a point"},
    { NULL, NULL, 0, NULL}
};

static struct PyModuleDef ptexamplemodule = {
    PyModuleDef_HEAD_INIT,

```

```

"ptexample",          /* name of module */
"A module that imports an API", /* Doc string (may be NULL) */
-1,                  /* Size of per-interpreter state or -1 */
PtExampleMethods      /* Method table */
};

/* Module initialization function */
PyMODINIT_FUNC
PyInit_ptexample(void) {
    PyObject *m;

    m = PyModule_Create(&ptexamplemodule);
    if (m == NULL)
        return NULL;

    /* Import sample, loading its API functions */
    if (!import_sample()) {
        return NULL;
    }

    return m;
}

```

çijŮerSèfZäylæŮrælaaiŮæŮüijNä;ăçTŽèGşäy■éIJĂëçAăŌzèĂĈèŽŚæĂŌæăuârEăĜ;æŤrăžŞæLŮăžççă  
ăĭNăçĈijNä;ăăRrăžăăĈRăyNéİçèfZăăuăLZăžzăyĂăyĭçőĂă■ŤçŽĎ setup.py æŮĜăžüijŽ

```

# setup.py
from distutils.core import setup, Extension

setup(name='ptexample',
      ext_modules=[
          Extension('ptexample',
                  ['ptexample.c'],
                  include_dirs = [], # May need pysample.h
→directory
                  )
      ]
)

```

ăçĈădIJăyĂăLĜæ■čăyŷijNä;ăăijŽăRŚçŌřă;ăçŽĎæŮræL'ăśŤăĜ;æŤrèĈ;ăŠNăŏŽăžL'ăIJăăĚăžŮălaăă  
APIăĜ;æŤrăyĂèŷüèfŘèăNçŽĎăĭLăě;ăĂĈ

```

>>> import sample
>>> p1 = sample.Point(2,3)
>>> p1
<capsule object "Point *" at 0x1004ea330>
>>> import ptexample
>>> ptexample.print_point(p1)
2.000000 3.000000
>>>

```

## èõlèõž

æIJñèŁĆąšžāžŎäyÄäyłāŁ■æRŘāřsæYřijÑèČuāZŁāržèšæČ;èŎuāRŮāzzā;Ťā;āæČšèèAçŽĎāržèšæčŽĎ  
èŁŽæāūčŽĎèřijÑāōŽāzŁ'æłāāIŮāijŽāāñāĚĚäyÄäyłāG;æŤræŃĜéŠŁçŽĎçžšæđĎā;šijNāŁŽāžžāyÄäyłæŃČ  
ā;NāèČ sample.\_point\_api.

āĚūāžŮæłāāIŮèČ;ād' šāIJārījāĚæŮūèŎuāRŮāŁrèŁŽāyłāsđæĀğāžūæRŘāRŮāžŤāsČçŽĎæŃĜéŠŁāĀČ  
āžNāōđäyŁijŃPythonæRŘā;ŽāžĚ PyCapsule\_Import()  
āūèāĚūāG;æŤřijNāyžāžĚāōŃæŁræŁ'ĀæIJŁ'çŽĎæ■ēēłđ'āĀČ  
ā;āāRlèIJĀæRŘā;ŽāsđæĀğçŽĎāR■ā■Ůā■šāRřijŁæřŤæČsample.\_point\_apiiijL'ijŃçĎuāRŎāžŮāršāijŽāyĀ

āIJlārĚèčnārījāĜžāG;æŤrāRŸāyžāĚūāžŮæłāāIŮāy■æŽōéĀŽāG;æŤræŮūijŃæIJŁ'āyĀāžŽCçijŮčlNéŽū  
āIJl pysample.h æŮĜāžūāy■ijNāyÄäył \_point\_api  
æŃĜéŠŁèčnçŤlæłææŃĜāRšāIJārījāĜžæłāāIŮāy■èčnāŁlāğNāŃŮçŽĎæŮžæšŤeāłāĀČ  
āyÄäyłçŽyāĚšçŽĎāG;æŤř import\_sample() èčnçŤlæłææŃĜāRšèČuāZŁārījāĚēāžūāŁlāğNāŃŮèŁŽāyłæ  
èŁŽāyłāG;æŤrāŁĚēāžāIJlāžžā;ŤāG;æŤrèčnā;ŁçŤlāžNāŁ'■èčnèřČçŤlāĀČēĀŽāyŷæłèèšijŃāōČāijŽāIJlæłāāI  
æIJĀāRŎijŃČçŽĎéčĎād'ĎçĚĚāōRèčnāōŽāzŁ'ijŃèčnçŤlæłææĀŽèŁĜæŮžæšŤeāłāŎžāŁĚāRšèŁŽāžZAPIāG  
çŤlæŁūāRlèIJĀèèAā;ŁçŤlèŁŽāžžāŎšāğNāG;æŤrāR■çğřā■šāRřijNāy■èIJĀèèAèĀŽèŁĜāōRāŎžāžĚğčāĚūā

æIJĀāRŎijŃèŁŸæIJŁ'āyÄäyłéĜ■èèAçŽĎāŎšāžæđŮ'ā;āāŎžā;ŁçŤlèŁŽāyłæŁ'ĀæIJræłèèŠ;æŎèæłāāIŮā  
āèČæđIJā;āāy■æČšā;ŁçŤlæIJñæIJçŽĎæŁ'ĀæIJřijŃèČčā;āāršāĚĚēāžā;ŁçŤlāĚšāžnāžšçŽĎénŸçžğçŁ'žæĀğā  
ā;NāèČijŃārĚäyÄäyłæŽōéĀŽçŽĎAPIāG;æŤræŤ;āĚäyÄäyłāĚšāžnāžšāžūçāōāŁlæŁ'ĀæIJŁ'æŁ'āšŤæłāāIŮ  
èŁŽçğ■æŮžæšŤçāōāōđāRřēāŃijŃā;ĚæŸrāōČçŽyāržçžAçRŘijŃçŁ'žāŁnæŸrāIJlād'ğāđŃçšççžšāy■āĀČ  
æIJñèŁĆēijŤçđ'žāžĚāèČā;ŤēĀŽèŁĜPythonçŽĎæŽōéĀŽārījāĚēæIJžāŁūāšŃāžĚāžĚāĜāyłèČuāZŁèřČçŤlæ  
āržāžŎæłāāIŮçŽĎçijŮèřijŃā;āāRlèIJĀèèAāōŽāzŁ'ād't æŮĜāžūijŃèĀNāy■èIJĀèèAèĀČèŽšāG;æŤrāžšçŽ

æŽt'ād'ŽāĚšāžŎāŁl'çŤlČ APIæłæđĎéĀāæŁ'āšŤæłāāIŮçŽĎāŁæAřāRřāžēāRČèĀČ  
PythonçŽĎæŮĜæaç

## 17.6 15.6 āžŎCèr■èlĀäy■èřČçŤlPythonāžčçāĀ

### éŮōécŸ

ā;āæČšāIJlČäy■āōŁ'āĚlçŽĎæŁ'ğēāŃæšRřāyłPythonèřČçŤlāžūèŁŤāžđçžšæđIJçžŽČāĀČ  
ā;NāèČijŃā;āæČšāIJlČèr■èlĀäy■ā;ŁçŤlæšRřāyłPythonāG;æŤrā;IJāyžāyÄäyłāŽđèřČāĀČ

### èğčāĚšæŮžæāŁ

āIJlČèr■èlĀäy■èřČçŤlPythonēłđāyŷçōĀā■ŤijNāy■èŁĜèō;èōāāŁrāyĀāžžārRçł■éŮlāĀČ  
āyŃēlççŽĎCāžčçāĀāšŁèřŁ'ā;āæĀŎæāūāōŁ'āĚlçŽĎèřČçŤlřijŽ

```
#include <Python.h>

/* Execute func(x,y) in the Python interpreter. The
   arguments and return result of the function must
   be Python floats */

double call_func(PyObject *func, double x, double y) {
```

```

PyObject *args;
PyObject *kwargs;
PyObject *result = 0;
double retval;

/* Make sure we own the GIL */
PyGILState_STATE state = PyGILState_Ensure();

/* Verify that func is a proper callable */
if (!PyCallable_Check(func)) {
    fprintf(stderr, "call_func: expected a callable\n");
    goto fail;
}
/* Build arguments */
args = Py_BuildValue("(dd)", x, y);
kwargs = NULL;

/* Call the function */
result = PyObject_Call(func, args, kwargs);
Py_DECREF(args);
Py_XDECREF(kwargs);

/* Check for Python exceptions (if any) */
if (PyErr_Occurred()) {
    PyErr_Print();
    goto fail;
}

/* Verify the result is a float object */
if (!PyFloat_Check(result)) {
    fprintf(stderr, "call_func: callable didn't return a float\n");
    goto fail;
}

/* Create the return value */
retval = PyFloat_AsDouble(result);
Py_DECREF(result);

/* Restore previous GIL state and return */
PyGILState_Release(state);
return retval;

fail:
Py_XDECREF(result);
PyGILState_Release(state);
abort();    // Change to something more appropriate
}

```

èeAä;fçTlèfZäyläG;æTrijNä;æeIJÄèeAèOüaRÜäijäeÄSèfGæIèçZDæ§Räyläüš■YäIJlPythonèrČçTlçŽ  
 æIJLä;Läd'Žçg■æÚæşTäRfrazèèö'ä;æefZæäüäAŽrijN ærTäeCärEäyÄäyläRrèrČçTlärzèšäijäçzZäyÄäyläL

äyÑéÍæÝřäyÄäyŁçŃÄä■Tä¿Nä■ŘçŤlæİæŎl'éeřázŎäyÄäyŁąŤNäĚčŽĐPythonèğćéĠŁăZlăy■erČçŤlăy

```
#include <Python.h>

/* Definition of call_func() same as above */
...

/* Load a symbol from a module */
PyObject *import_name(const char *modname, const char *symbol) {
    PyObject *u_name, *module;
    u_name = PyUnicode_FromString(modname);
    module = PyImport_Import(u_name);
    Py_DECREF(u_name);
    return PyObject_GetAttrString(module, symbol);
}

/* Simple embedding example */
int main() {
    PyObject *pow_func;
    double x;

    Py_Initialize();
    /* Get a reference to the math.pow function */
    pow_func = import_name("math", "pow");

    /* Call it using our call_func() code */
    for (x = 0.0; x < 10.0; x += 0.1) {
        printf("%0.2f %0.2f\n", x, call_func(pow_func, x, 2.0));
    }
    /* Done */
    Py_DECREF(pow_func);
    Py_Finalize();
    return 0;
}
```

èeAædĐäzžä¿Nä■ŘäzčçäAijjNä¿æÍJĚeçAçijŮerŚCázũärĚăŃČeŞ¿æŎěăĹřPythonèğćéĠŁăZlăĚĂČ  
äyÑéÍçŽĐMakefileăRřäzěæŤŽä¿äæĚŎæăũăAŽiijLăy■efĠăĹlă¿äæIJăŽăZlăyŁéÍcéIJĚeçAäyÄäzŽéĚ■ç¿iijL

```
all::
    cc -g embed.c -I/usr/local/include/python3.3m \
        -L/usr/local/lib/python3.3/config-3.3m -lpython3.3m
```

çijŮerŚăzũeřRëaŇäijŽăžğçŤşçşăiijjăyÑéÍçŽĐè¿ŞăĠzriijŽ

```
0.00 0.00
0.10 0.01
0.20 0.04
0.30 0.09
0.40 0.16
...
```

äyÑéíçæÝřäyÄäyłçí■āŁőäy■āŔŇçŽĐäŁŇ■ŔiijŇāsŤçd'žāžEäyÄäyłæLŦ'āsŤāĜ;æŦŕiijŇ  
āōČæŎēāŔŮäyÄäyłāŔřērČçŤlāržēsāŠŇāĚüāzŮāŔČæŦŕiijŇāzūārĚāōČāznāijāēĀŠçzŽ  
call\_func() æĪēāĀŽætŦŇērŦiijŽ

```
/* Extension function for testing the C-Python callback */
PyObject *py_call_func(PyObject *self, PyObject *args) {
    PyObject *func;

    double x, y, result;
    if (!PyArg_ParseTuple(args, "Odd", &func, &x, &y)) {
        return NULL;
    }
    result = call_func(func, x, y);
    return Py_BuildValue("d", result);
}
```

ä;ŁçŤlēfŽäyłæLŦ'āsŤāĜ;æŦŕiijŇā;āēĀāČŔäyÑéíçēfŽæāüætŦŇērŦāōČŕiijŽ

```
>>> import sample
>>> def add(x, y) :
...     return x+y
...
>>> sample.call_func(add, 3, 4)
7.0
>>>
```

## ēōĪēōž

āēČædĪĲā;āāĪĲČēr■ēĪÄäy■ērČçŤĲPythoniijŇēēĀēōŕā;ŔæĪĲĀēĜ■ēēĀçŽĐæÝŕČēr■ēĪÄāijŽæÝřäyžā;ŠāĀ  
āzšāršæÝřērŦ'iiijŇČēr■ēĪĀēr'šet'čædĐēĀāŔČæŦŕāĀĀērČçŤĲPythonāĜ;æŦŕāĀĀæčĀæšēāijČäyŷāĀĀæčĀæ

ä;ĪĲäyžçñnäyĀæ■ēiijŇā;āāŁĒēāzāĒŁæĪĲ'äyÄäyłēāłçd'žā;āārĒēēĀērČçŤĲčŽĐPythonāŔřērČçŤlāržēsāā  
ēŁŽāŔŕāzēæÝřäyÄäyłāĜ;æŦŕāĀĀçšzāĀĀæŮzæsŤāĀĀāĒĒç;ōæŮzæsŤæŁŮāĒüāzŮāzžæĐŔāōđçŎŕāžĒ  
\_\_call\_\_() æŠ■ä;ĪĲçŽĐäyĪĲēēŁāĀČ äyžāžĒçāōāŁæÝřāŔřērČçŤĲčŽĐiijŇāŔŕāzēāČŔäyÑéíççŽĐāzççāĀē  
PyCallable\_Check() āĀŽæčĀæšēiijŽ

```
double call_func(PyObject *func, double x, double y) {
    ...
    /* Verify that func is a proper callable */
    if (!PyCallable_Check(func)) {
        fprintf(stderr, "call_func: expected a callable\n");
        goto fail;
    }
    ...
}
```

āĪĲČāzççāĀēĜŇād'ĐçŔĒēŤŽēŕŕā;āēĪĲēēĀæāijād'ŮçŽĐārŔāŁČāĀČäyĀēŁŇæĪēēōšŕiijŇā;āäy■ēČ;āzĒā  
ēŤŽēŕŕāzŤērēā;ŁçŤĲČāzççāĀæŮzāijŔæĪēēčŇād'ĐçŔĒāĀČāĪĲēŁŽēĜŇiijŇæŁŠāznæL'ŠçōŮārĒāržēŤŽērŕçŽĐ  
abort() çŽĐēŤŽērŕād'ĐçŔĒāŽĲāĀČ āōČāijŽçzŠæĪšæŎŁ'æŦŦ'äyłçíŇāzŔiijŇāĪĲĲĲšāōđçŎŕāčČäyŇéíçā;āā  
ä;āēēĀēōŕā;ŔçŽĐæÝřāĪĲēŁŽēĜŇČæÝřäyžēğŦŕiijŇāŽāæ■d'āzūæsāæĪĲ'ēu\$æŁŽāĜzāijČäyŷçŽyāržāzŤçŽĐæ  
ēŤŽēŕŕād'ĐçŔĒæÝřā;āāĪĲçijŮçĲŇæŮūāŁĒēāzēēĀēĀČēŽŠçŽĐāžŇæČĒāĀČ

erCçTlāyÄäylāG;æTṛçZyārfzælēðōšā;ŁçōĀā■TāĀTāĀTāRlēIJĀēēAä;ŁçTl  
PyObject\_Call() iijN äijäyÄäylāRrēŕCçTlārfzēšāçzZāōČāĀÄyÄäylāRCæTṛāĒCçzDāŠNäyÄäylāRréĀ  
ēēAædDāzžāRCæTṛāĒCçzDæLŪā■ŪāĒyijNä;āāRfāzēā;ŁçTl Py\_BuildValue()  
.æCāyNrijZ

```
double call_func(PyObject *func, double x, double y) {
    PyObject *args;
    PyObject *kwargs;

    ...
    /* Build arguments */
    args = Py_BuildValue("(dd)", x, y);
    kwargs = NULL;

    /* Call the function */
    result = PyObject_Call(func, args, kwargs);
    Py_DECREF(args);
    Py_XDECREF(kwargs);
    ...
}
```

æCædIJæšqæIJLāĒšēTōā■ŪāRCæTṛijNä;āāRfāzēāijæĀŠNULLāĀČā;Šā;āēēAēŕCçTlāG;æTṛæŪiijN  
éIJĀēēAçāōāŁlā;ŁçTlāzĒ Py\_DECREF() æLŪēĀĒ Py\_XDECREF() æyĒçREāRCæTṛāĀČ  
çññāzNäylāG;æTṛçZyārfzāōLāĒlçCzrijNāZāyZāōČāĒAēōyāijæĀŠNULLæNĠēŠLijLçZt'æŌēāŁ;çTṛēāōČrij  
ēŁZāzšæYṛāyZāzĀāzŁæLŠāznä;ŁçTlāōČāĒæyĒçREāRréĀLçZDāĒšēTōā■ŪāRCæTṛāĀČ

erCçTlāyGPythonāG;æTṛāzNāRŌijNä;āāĒĒēāzæčĀæšēæYṛāRææIJLāijCāyYāRŠçTšāĀČ  
PyErr\_Occurred() āG;æTṛāRfēcñçTlāĒāAZēŁZāzūāzNāĀČ  
ārfzārfzāzŌāijCāyYçZDād'DçREārsæIJLçČZēzçČēāzĒijNçTšāzŌæYṛçTlCēŕ■ēĀĀĒZçZDijNä;ææšqæIJLāČ  
āZāæ■d'ijNä;āāĒĒēāzēēAēō;ç;ōāyÄäylāijCāyYçLŪæĀAçāĀijNāL'Šā■ŕāijCāyYāŁæAṛæLŪāĒūāzŪçZyāzT  
āIJlēŁZēGNijNāLŠāznēĀL'æNl'āzĒçōĀā■TçZD abort()  
ælēād'DçREāĀČāRēād'ŪijNāijāçzšCçlNāzRāŠYāRfēC;āijZçZt'æŌēēōl'çlNāzRāēTæzČāĀČ

```
...
/* Check for Python exceptions (if any) */
if (PyErr_Occurred()) {
    PyErr_Print();
    goto fail;
}
...
fail:
    PyGILState_Release(state);
    abort();
}
```

āzŌēŕCçTlPythonāG;æTṛçZDēŁTāZdāĀijäy■æRRāRŪāŁæAṛéĀZāyYēēAēŁZēāNçszādNæčĀæšēāŠNā  
ēēAēŁZæāūāĀZçZDēŕlrijNä;āāĒĒēāzā;ŁçTlPythonārZēšāšCāy■çZDāG;æTṛāĀČ  
āIJlēŁZēGNāLŠāznä;ŁçTlāzĒ PyFloat\_Check() āŠN PyFloat\_AsDouble()  
ælēæčĀæšēāŠNāRRāRŪPythonætōçCzæTṛāĀČ

æIJĀāRŌāyÄäylēŪōēçYæYṛārfzāzŌPythonāĒlāsĀēTĀçZDçōāçREāĀČ  
āIJlCēŕ■ĒĀy■ēōŁēŪōPythonçZDæŪūāĀZijNä;æēIJĀēēAçāōāĒIGILēcñā■ççāççZDēŌūāRŪāŠNēGLæT;āz  
āy■çDūçZDēŕlrijNāRrēC;āijZārijēGt'ēğcēGLāZlēŁTāZdēTŻēŕŕæTṛæ■ōāLŪēĀĒçZt'æŌēāēTæzČāĀČ



```
double call_func(PyObject *func, double x, double y) {
    ...
    double retval;

    /* Make sure we own the GIL */
    PyGILState_STATE state = PyGILState_Ensure();
    ...
    /* Code that uses Python C API functions */
    ...
    /* Restore previous GIL state and return */
    PyGILState_Release(state);
    return retval;
}

fail:
    PyGILState_Release(state);
    abort();
}
```

```

    èeAæşlæĐRçŽDæYřærŘäyÄäyl      PyGILState_Ensure()
    èrČčTíáfEéazèùşçİĂăyĂăylaŇzeĚčŽD      PyGILState_Release()
    èrČčTíāĀtāĀtā■sä;ŁæIJL'ėTŻerrāRŚčTšāĀĆ   āIJlēfZēGNīijŇNēĹSāznā;ŁçTílāyĂăyl goto
    èr■āRēçIJŇNāyŁāŌzæYřäylāRræĀTçŽDëø;ēōaiijŇ ā;EæYřāōdēZĚäyŁæĹSāznā;ŁçTíāōCæIēēōšæŌğāĹūæIČē
    āIJl      fail:      æāĞč■;āRŌéIćçŽDäžčcāAāŠŇPythončŽD      fianl:
    āiUčŽDčTíēĀTæYřäyĂăæăüçŽDāĀĆ

```

17.7 15.7 äzÖCæL'ásTäy■ĖŁæT,ăĖÍásĂéTȦ

ä|äæČšèöl' CæL'l'ášTžzččäAäŠNPythonèğçéGŁăZlăy■čZďăĚüăzŮëfZčlNăyĂețuă■čçăoçZďăL'gèăŇiij  
éČčăzLă;ăârśéIJAèçAăŌzéGŁăT;ăžúéĚ■ăŮřěŌüăRŮăĚlăsĂèğçéGŁăZlėTAŋiijLGILiijL'ăĂĆ

ǎIÍĈŁ'ǎŝTăzčĉăAäy■īīŃGILăRărăžēěĂŽefĞăIJlăžĉĉăAäy■ăRŠăEěäyNélcèfZăăucŽĐăóRăielēcĜLă

èóíèőž

## 17.8 15.8 CăŞÑPythonăy■çŽĐçξçÍÑæũũçŦí

éŮóécŸ

èġčǎẸșæŮźæąŁ

áržāžŎāzžā;ŦērĈçŦĪPythonāržžēšāēĹŬPython C APIçŽĎCāžčçāAīijŊçaōāfĪā;āēēŬāĖĹāušçžRæ■čçaōā  
 èfZāRfāžēçŦĪ PyGILState\_Ensure() āŠŊ PyGILState\_Release()  
 ælēāZāĹrīijŊāçĀyŊāēL'Āçd'žīijŽ

```
PyGILState_Ensure()
PyGILState_Release() .
```

aIJaēuL'āRĽāLrCaŠNPythončŽDénYčžgčlNāžRāy■iijNā;Ľād'ŽāžNāčĚäyÄətuāAŽæYřā;ĽāyÿègAçŽĽ  
 āRrēČ;æYřāřzCāĀPythonāĀCčžfčlNāĀPythončžfčlNčŽDæuūāRĽā;fçTīāĀC  
 āRlèeAä;āçāōāfīègčēGLāŽlčēnā■čçāōčŽDāĽlāgNāNŮiijNāžūāyTæuL'āRĽāLrègčēGLāŽlčŽDCāžčçāAæL'g

**17.9 15.9 çTÍWSIGǎÑĚèčĚCăžčçăA**

ä;äæČšèöl'ä;ääEŽčŽDCäzččāAä;IJäyžäyÄäy!CæL'l'ásTælaä!Uæ!ëèøféUöiijNæČšéÄŽèĜä;£çTí  
SwigāNĖèçĖčTšæĹŖāZĪ ælěāōNæĹŖāĀĆ

SwigéÅžēſGēgčæđŔCād't'æŨGäzūāžūēGġāLāLāLZāžžæL't'āsŦāžččăAæĪæŞ■ĪJāĀĆ  
èĕAă;ſçŦlāōĆġġNă;ăăĒĒlèĕAæĪJL'äyĀyĲCād't'æŨGäzūāĀĆă;NăēĆġġNăĒSăžņčđ'žă;ŅčŽDād't'æŨGäzūāē

```
/* sample.h */

#include <math.h>

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
```

```

} Point;

extern double distance(Point *p1, Point *p2);

```

äyÄæŮëä;äæIJL'ázÈè£Zäyłäd't' æŮĜäzŭiijŇäyŇäyÄæ■ěåršæŸřcijŮâEžäyÄäyłSwigâÄIæŮěâŘčâÄIæŮæŇL'çĚğçžęăǫŽiijŇë£ZăžZæŮĜäzŭäzěâÄI.äÄIäŮŮçijÄäzŭäyŤçşzäijijäyŇéIćè£ZæăüiijŽ

```

// sample.i - Swig interface
%module sample
%{
#include "sample.h"
%}

/* Customizations */
%extend Point {
    /* Constructor for Point objects */
    Point(double x, double y) {
        Point *p = (Point *) malloc(sizeof(Point));
        p->x = x;
        p->y = y;
        return p;
    };
};

/* Map int *remainder as an output argument */
#include typemaps.i
%apply int *OUTPUT { int * remainder };

/* Map the argument pattern (double *a, int n) to arrays */
%typemap(in) (double *a, int n) (Py_buffer view) {
    view.obj = NULL;
    if (PyObject_GetBuffer($input, &view, PyBUF_ANY_CONTIGUOUS |
↪PyBUF_FORMAT) == -1) {
        SWIG_fail;
    }
    if (strcmp(view.format, "d") != 0) {
        PyErr_SetString(PyExc_TypeError, "Expected an array of doubles
↪");
        SWIG_fail;
    }
    $1 = (double *) view.buf;
    $2 = view.len / sizeof(double);
}

%typemap(freearg) (double *a, int n) {
    if (view$argsnum.obj) {
        PyBuffer_Release(&view$argsnum);
    }
}

```

```

/* C declarations to be included in the extension module */

extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);

```

äÿÄæŮë;ääEŽäë;äzEæŒëäRčæŮĜäzŭiijŇärsäRfäzëäIJläS;äzd'ëäŇäüëäEüäÿ■ërCçŤlSwigäzEiijŽ

```

bash % swig -python -py3 sample.i
bash %

```

swigçŽDè;ŠäĜžärsæŸräy'd'äylæŮĜäzŭiijŇsample\_wrap.cäŠŇsample.pyäĂĆ  
 äRŒëlcçŽDæŮĜäzŭärsæŸrçŤlæLüéIJäëçAärijaëççŽDäĂĆ èĂŇsam-  
 ple\_wrap.cæŮĜäzŭæŸréIJäëçAëcñcijŮërSälŖäR■äRñ \_sample  
 çŽDæŤræŇAælääIŮçŽDcäzççäAäĂĆ èŁŽäyŤäRfäzëéĂŽèŁĜëü\$æŽöéĂŽæL'l'äsŤælääIŮäÿĂæäüçŽDæLĂæ  
 ä;ŇäçĆiijŇä;ääLŽäzžäzEäÿĂäÿläçCäÿŇæL'Ăçd'žçŽD setup.py æŮĜäzŭiijŽ

```

# setup.py
from distutils.core import setup, Extension

setup(name='sample',
      py_modules=['sample.py'],
      ext_modules=[
          Extension('_sample',
                  ['sample_wrap.c'],
                  include_dirs = [],
                  define_macros = [],

                  undef_macros = [],
                  library_dirs = [],
                  libraries = ['sample']
                  )
      ]
)

```

èçAçijŮërSäŠŇæŤŇërŤiijŇäIJlsetup.pyäÿLæL'ğëäŇpython3iijŇäçCäÿŇiijŽ

```

bash % python3 setup.py build_ext --inplace
running build_ext
building '_sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
  ↳ prototypes
-I/usr/local/include/python3.3m -c sample_wrap.c

```

```

-o build/temp.macosx-10.6-x86_64-3.3/sample_wrap.o
sample_wrap.c: In function 'PySWIG_InitializeModule':
sample_wrap.c:3589: warning: statement with no effect
gcc -bundle -undefined dynamic_lookup build/temp.macosx-10.6-x86_64-
3.3/sample.o
build/temp.macosx-10.6-x86_64-3.3/sample_wrap.o -o _sample.so -
lsample
bash %

```

æċCædIJäyÄäLĜæ■čäyÿçŽDèrlīijNä;äaijŽâRŚçÖřä;ääřsâRřäzēā;LæŮzä;ŁçŽDä;ŁçŤlċŤšæLŘçŽDCæL

```

>>> import sample
>>> sample.gcd(42, 8)
2
>>> sample.divide(42, 8)
[5, 2]
>>> p1 = sample.Point(2, 3)
>>> p2 = sample.Point(4, 5)
>>> sample.distance(p1, p2)
2.8284271247461903
>>> p1.x
2.0
>>> p1.y
3.0
>>> import array
>>> a = array.array('d', [1, 2, 3])
>>> sample.avg(a)
2.0
>>>

```

## ëóíëőž

SwigæŸřPythonăŎĖâRšäy■ædĎäzžæL'ſâŤæſaāIŮçŽDâR■çğřāzūæŃĜăőŽăžĖCăd'ſæŮĜăzūīijN  
SwigēČ;ĕĜſāLſāNŮſā;Lăd'ŽăŃĖĕĕĖçŤšæLŘăZlċŽDăd'ĎçRĖāĀĆ

æL'ĂæIJL'SwigæŎĕâRĉéČ;ăžēçšzăijijăyNéÍĉèŁæăŭçŽDăyžăijĂăd'ſīijŽ

```

%module sample
%{
#include "sample.h"
%}

```

ēŁZăyſăzĖăzĖâRſæŸřăĉræŸŎăžĖæL'ſâŤæſaāIŮçŽDâR■çğřāzūæŃĜăőŽăžĖCăd'ſæŮĜăzūīijN  
ăyžăžĖĕČ;ĕŏſ'ċijŮĕřſéĂžēŁĜăŁĖĖăžēĕAăŃĖĖâRſēŁZăžŽăd'ſæŮĜăzūīijLă;■ăžŎ%{ăſŃ%}  
çŽDăžĉçăAīijL'īijNăřĖăăŎČăznăžNéŮſăd'■ăLŮçſŸĕſſăLřĕ;ſăĜăžăžĉçăAăy■īijNēŁZăžſæŸřă;ăĕĕAæŤċ;ŏæL

SwigæŎĕâRĉçŽDăžŤăyNéČſăLĖæŸřăyĂăyſCăĉræŸŎăLŮĕăſīijNă;ăĖIJĂĕĕAăIJăL'ſâŤăy■ăŃĖĖâRſăăŏ  
ēŁZéĂžăyſăžŎăd'ſæŮĜăzūăy■ĕĉnăd'■ăLŮăĂĈăIJăLſăžſçŽDă;Nă■Răy■īijNăLſăžſăžĖăžĖăĈRăyNéÍĉèŁ

```
%module sample
%{
#include "sample.h"
%}
...
extern int gcd(int, int);
extern int in_mandel(double x0, double y0, int n);
extern int divide(int a, int b, int *remainder);
extern double avg(double *a, int n);

typedef struct Point {
    double x,y;
} Point;

extern double distance(Point *p1, Point *p2);
```

æIJL'äyÄçÇzéIJÄëeAaijzèrÇçŽDæYrèfZázZäçræYÖaijZäSLeL'Swigä;äæÇsëeAäIJlPythonælaäIÜäy■  
éÄZäyyä;æeIJÄëeAçijÜë;SëfZäyIäçræYÖäLÜëaIæLÜçZyázTçŽDäföæTzäyNäoCäÄÇ  
ä;NäeCrijNäeCädlJä;ääy■æÇsæ§RäzZäçræYÖëcñäNĖäRnèfZæIërijNä;äeëAärEäoCäzÖäçræYÖäLÜëaIäy■  
ä;fçTlSwigæIJÄäd'■æIÇçŽDäIJræÜzæYræoCëÇ;çzŽCäzççäAæRŘä;Zäd'gëGRçŽDëGlaöŽázL'æS■ä;IJ  
èfZäyIäyžécYäd'läd'grijNëfZëGÑæUäæşTäşTäijÄrijNä;EæYræLSäzñäIJæIJnèLÇèfYäL'l'äsTçd'zäzEäyÄä  
çñnäyÄäyIëGlaöŽázL'æYr%extend æNGäzd'aĖAëöyæÜzæşTëcñéŽDäLäaLräušä■YäIJlçŽDçzŞædDä  
æLSä;Nä■Räy■rijNëfZäyIëcñçTlæIëæuäZääyÄäyIPointçzŞædDä;ŞçŽDædDëÄäZlæÜzæşTäÄÇ  
äoCäRfäzëëol'ä;ääCRäyNéIcèfZæuä;fçTlæfZäyIçzŞædDä;ŞrijZ

```
>>> p1 = sample.Point(2,3)
>>>
```

æeCädlJçTëèfGçŽDërfrijNPointärzèesäârşäĖEëazäzæZt'äLääd'■æIÇçŽDæÜzaijRæIëëcñäLZäzrijZ

```
>>> # Usage if %extend Point is omitted
>>> p1 = sample.Point()
>>> p1.x = 2.0
>>> p1.y = 3
```

çñnäyNäyIëGlaöŽázL'æul'äRĖLäLrärz typemaps.i äzŞçŽDäijTäĖëäSÑ  
%apply æNGäzd'rijN äoCäijZæNGçd'žSwigäRÇæTřç■äR■ int \*remainder  
ëeAëcñä;ŞAŽæYrë;ŞäGzäÄijäÄÇ èfZäyIäoðéZĖäyLæYräyÄäyIäIäijRäNzéĖ■ëgDälZäÄÇ  
äIJlæÖëäyNäIëçŽDæL'ÄæIJL'äçræYÖäy■rijNäzä;TæUüäÄZäRlëeAççräyL int  
\*remainder rijNäzÜärsäijZëcñä;IJäyžë;ŞäGzäÄÇ èfZäyIëGlaöŽázL'æÜzæşTäRfäzëëol'  
divide() äG;æTřèfTäZdäyd'äyIäÄijäÄÇ

```
>>> sample.divide(42,8)
[5, 2]
>>>
```

æIJÄäRÖäyÄäyIæul'äRĖLäLr%typemap æNGäzd'çŽDëGlaöŽázL'äRrëÇ;æYrèfZëGÑäşTçd'žçŽDæIJÄ  
äyÄäyItypemapärşæYräyÄäyIäIJlë;ŞäĖëäy■çL'zäoZäRÇæTřæIäaijRçŽDëgDälZäÄÇ  
äIJlæIJnèLÇäy■rijNäyÄäyItypemapëcñäoZäZL'äyžäNzéĖ■äRÇæTřæIäaijR (double \*a,

int n) . aIItypemapāEĒēČlāēYřāyĀäyĹCäzččāAçL'ĜæøġġġNāōČāŚLēřL'SwigæĀŌæūāřEäyĀäyĹPythonāř  
æIJñēŁCäzččāAā;ŁçTĪāžEPythonçŽDçġŠā■Yā■RēōōāŌzāŇzéĒ■āzzā;TçIJNāyĹāŌzçśzāġġġāRŇçš;āžæTřçž  
ġġġLārTāēČNumPyæTřçžDāĀArrayāġāĪŪāŁZāžžçŽDæTřçžDç■L'ġġġġġNæŽt'ād'ŽēřūāRCèĀČ15.3ārRēŁČ

āIItypemapāzččāAāEĒēČġġġN\$1āŚN\$2ēŁZæāūçŽDāRŸēĜRæZŁæ■āġġŽēŌūāRŪtypemapāġāġġRçŽDČ  
ġġġLārTāēČ\$1æYāārDäyž double \*a ġġġL'āĀČ\$ġinputæŇĜāRŚāyĀäyĹā;ġġyžē;ŠāĒēçŽD  
PyObject \* āRCæTřġġġ ēĀŇ \$argnum āřsāzčēāġāRČæTřçžŽDäyĹæTřāĀČ

çġġŪāEŽāŚŇçŘEēġçtypemapsæYřā;ŁçTĪSwigæġġġāšžæġġçŽDāL'■æRRāĀČ  
äy■āžEæYřēřt'āžččāAæŽt'çēđçġYġġġNēĀŇāyTā;āēġġēēAçŘEēġçPython C  
APIāŚŇSwigāŚŇāōČāžd'āžŠçŽDæŪžāġġRāĀČ SwigæŪĜæaçæġġL æŽt'ād'ŽēŁZæŪžēĹççŽDçžEēŁČġġġNāRřāž

äy■ēŁĜġġġNāēČæđġġā;āæġġL'ād'ġēĜRçŽDČäzččāAēġġġēēAēćnæŽt'ēġġsāyžæL'ġāsTāēġāġġŪāĀČ  
SwigæYřāyĀäyĹēĹđāyŷāġžād'ġçŽDāūēāĒūāĀČāĒēTōçČzāġġāžŌSwigæYřāyĀäyĹād'ĐçŘEČāčřæYŌçŽDçġġ  
ēĀŽēŁĜāġžād'ġçŽDæġāġġRāŇzéĒ■āŚNēĜġāōŽāžL'çžDāžŪġġġNāRřāžēēōġ'ā;āæŽt'æTžāčřæYŌæŇĜāōŽāŚŇç  
æŽt'ād'ŽāŁqæAřēřūāŌzæšēēYĒ Swigç;ŚçŇŽ ġġġ ēŁYæġġL'  
çŁ'žāōŽāžŌPythonçŽDçŽyāĒēšæŪĜæaç

## 17.10 15.10 çTĪCythonāŇĒēēČCäzččāA

### éŪōéćŸ

ā;āæČšā;ŁçTĪCythonæġēāŁZāžžāyĀäyĹPythonæL'ġāsTāēġāġġŪġġġçTĪæġēāŇĒēēČEæšŘāyĹāūšā■YāġġçŽD

### ēġçĀEşæŪžæāĹ

ā;ŁçTĪCythonæđDāžžāyĀäyĹæL'ġāsTāēġāġġŪçġġNāyĹāŌžā;ĹæL'ŇāEŽæL'ġāsTāēġġL'āžŽçśzāġġġġġ  
āžāäyžā;āēġġēēAāŁZāžžā;Ĺād'ŽāŇĒēēČĒĜ;æTřāĀČāy■ēŁĜġġġNēūšāL'■ēĹçāy■āRŇçŽDæYřġġġNā;āäy■ēġġġ

ā;ġġyžāĜEād'ĜġġġNāĜēō;æġġçāāžŇçž■ēČġāĹEçŽDçđ'žā;ŇāžččāAāūšçžŘēćŇçġġŪēřSāĹræšŘāyĹāR  
libsample çŽDČāĜ;æTřāžŠāy■āžEāĀČ ēçŪāĒĹāŁZāžžāyĀäyĹāR■āRŇ csample.pxd  
çŽDæŪĜāžŪġġġNāēČāyŇæL'Āçđ'žġġŽ

```
# csample.pxd
#
# Declarations of "external" C functions and structures

cdef extern from "sample.h":
    int gcd(int, int)
    bint in_mandel(double, double, int)
    int divide(int, int, int *)
    double avg(double *, int) nogil

    ctypedef struct Point:
        double x
        double y

    double distance(Point *, Point *)
```



ěĚŽäylæŮĜäzúãIJíCythonäy■çŽDä;IJçŤlärseũ§CçŽDäd't'æŮĜäzúäyÄæäüãĂĆ  
 åĹiågŇáčřæŸŮ cdef extern from "sample.h" æŇĜåōŽäžEæL'Äå■ęçŽĎCåd't'æŮĜäzúãĂĆ  
 æŌëäyŇæĹëçŽĎăčřæŸŌëĈ;æŸřæĹëçĜlăžŌëĈčäylăd't'æŮĜäzúãĂĆæŮĜäzúãŔ■æŸř  
 csample.pxd ĩijŇëĀŇäy■æŸř sample.pxd âĀŤâĀŤeĚŽćĈăĹLéĜ■ëęAăĂĆ  
 äyŇäyÄæ■ëřijŇăĹŽăžžäyÄäyĹăŔ■äyž sample.pyx çŽĎéŮőécŸăĂĆ  
 èřëæŮĜäzúãĳŽăōŽăžL'ăŇĚëçĚăŽĹĳŇçŤĹæĹëæąëæŌëPythonèĝčëĜĹăŽĹăĹř csample.  
 pxd äy■ăčřæŸŌçŽĎCăžčçăĂăĂĆ

```

# sample.pyx

# Import the low-level C declarations
cimport csample

# Import some functionality from Python and the C stdlib
from cpython.pycapsule cimport *

from libc.stdlib cimport malloc, free

# Wrappers
def gcd(unsigned int x, unsigned int y):
    return csample.gcd(x, y)

def in_mandel(x, y, unsigned int n):
    return csample.in_mandel(x, y, n)

def divide(x, y):
    cdef int rem
    quot = csample.divide(x, y, &rem)
    return quot, rem

def avg(double[:] a):
    cdef:
        int sz
        double result

    sz = a.size
    with nogil:
        result = csample.avg(<double *> &a[0], sz)
    return result

# Destructor for cleaning up Point objects
cdef del_Point(object obj):
    pt = <csample.Point *> PyCapsule_GetPointer(obj, "Point")
    free(<void *> pt)

# Create a Point object and return as a capsule
def Point(double x, double y):
    cdef csample.Point *p
    p = <csample.Point *> malloc(sizeof(csample.Point))
    if p == NULL:

```

```

        raise MemoryError("No memory to make a Point")
    p.x = x
    p.y = y
    return PyCapsule_New(<void *>p, "Point", <PyCapsule_Destructor>
↳del_Point)

def distance(p1, p2):
    pt1 = <csample.Point *> PyCapsule_GetPointer(p1, "Point")
    pt2 = <csample.Point *> PyCapsule_GetPointer(p2, "Point")
    return csample.distance(pt1, pt2)

```

èřæŮĜäzúæŽť ad'ŽčŽĎčzÈèŁĆéČlálĚajžŽaIJleóíeóžéČlálĚèřęczEāsŤajĀāĀĆ  
æIJĀāŔŌijŇäyžäžEæĎĎäžžæL'l'āsŤælqalŮijŇāČŘäyŇéíćèŁŽæăăăĹŽăžžăyĀăył setup.  
py æŮĜäzūijŽ

```

from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [
    Extension('sample',

                ['sample.pyx'],
                libraries=['sample'],
                library_dirs=['.'])]

setup(
    name = 'Sample extension module',
    cmdclass = {'build_ext': build_ext},
    ext_modules = ext_modules
)

```

èĕAæĎĎäžžæĹŚäžñætŇērŤčŽĎčŽóæăĜælqalŮijŇāČŘäyŇéíćèŁŽæăăăĹŽijŽ

```

bash % python3 setup.py build_ext --inplace
running build_ext
cythoning sample.pyx to sample.c
building 'sample' extension
gcc -fno-strict-aliasing -DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-
↳prototypes
-I/usr/local/include/python3.3m -c sample.c
-o build/temp.macosx-10.6-x86_64-3.3/sample.o
gcc -bundle -undefined dynamic_lookup build/temp.macosx-10.6-x86_64-
↳3.3/sample.o
-L. -lsample -o sample.so
bash %

```

æĕĆæĎIJäyĀāĹĜéazăĹl'čŽĎĕrlijŇä;ăăžŤēræIJĹăžEäyĀăyłæL'l'āsŤælqalŮ sample.  
so ijŇāŔŕāIJäyŇéíćă;Ňā■Řäy■ă;ŁçŤlīijŽ



```
File "<stdin>", line 1, in <module>
File "sample.pyx", line 7, in sample.gcd (sample.c:1284)
    def gcd(unsigned int x,unsigned int y):
OverflowError: can't convert negative value to unsigned int
>>>
```

æĈædIJä;äæĈşârZâNĖèĈĖĜ;æTŗăAŹăRĕăd' ŪĉZĎæĈĂæşĕiijNăRlĕIJĂĕĕAă;ĕĉTlăRĕăd' ŪĉZĎăNĖèĈĖ

```
def gcd(unsigned int x, unsigned int y):
    if x <= 0:
        raise ValueError("x must be > 0")
    if y <= 0:
        raise ValueError("y must be > 0")
    return csample.gcd(x,y)
```

ăIJlcsample.pxdæŪĜăZŭăy■ĉZĎ' 'in\_mandel()' 'ăĉræYŌæIJL'ăylă;ĹæIJL'ĕŭĉă;ĖæYŗæfTĕ;ĈĕZĭĉRĖĕĝ  
ăIJlĕfZăylæŪĜăZŭăy■iijNăĜ;æTŗĕĉnăĉræYŌăyĉĎŭăRŌăyĂăylbintĕĂNăy■æYŗăyĂăyhintăĂĈ  
ăŏĈăijZĕŏl'ăĜ;æTŗăLZăzăyĂăylæ■ĉĉăŏĉZĎBooleanăĀijĕĂNăy■æYŗĉŏĂă■TĉZĎæTŗ'æTŗăĂĈ  
ăZăæ■d'iijNĕĕTăZĎăĀijŌĕăĭĉd'žFalseĕĂNlĕăĭĉd'žTrueăĂĈ

ăIJlCythonăNĖèĈĖĂZlăy■iijNă;ăăRfăzĕĕĂL'æNl'ăĉræYŌCæTŗæ■ŏĉşădNĭijNăZşăRfăzĕă;ĕĉTlăL'ĂæIJ  
ârZăžŌ divide() ĉZĎăNĖèĈĖĂZlăşTĉd'žăZĖĕĕZăăŭăyĂăylă;Nă■RiijNăRŊNæŪŭĕĕYæIJL'ăĕĈă;TăŌZăd'D

```
def divide(x,y):
    cdef int rem
    quot = csample.divide(x,y,&rem)
    return quot, rem
```

ăIJlĕfZĕĜNĭijNrem âRŸĕĜRĕĉnæYĭĉd'žĉZĎăĉræYŌăyZăyĂăylCæTŗ'ăĎNăRŸĕĜRăĂĈ  
ă;ŞăŏĈĕĉnăijăăĖĕ divide() äĜ;æTŗĉZĎæŪŭăĂZiijN&rem  
ăLZăzăyĂăylĕŭşCăyĂăăŭĉZĎæNĜăRŞăŏĈĉZĎæNĜĕŞĹăĂĈ avg()  
ăĜ;æTŗĉZĎăZĉĉăĂăijTĉd'žăZĖCythonæZt'ĕnYĉĉĝĉZĎĉL'zæĂĝăĂĈ  
ĕĕŪăĖĹ def avg(double[:] a) äĉræYŌăZĖ avg()  
æŌĕăRŪăyĂăylăyĂĉzt'ĉZĎăRŊĉş;ăžĕăĖĖă■YĕĝĖăZĭăĂĈ æIJĂæĈĹăĕĜĉZĎĕĈlăĹæYŗĕĕTăZĎĉZĎĉZşăĎ

```
>>> import array
>>> a = array.array('d', [1,2,3])
>>> import numpy
>>> b = numpy.array([1., 2., 3.])
>>> import sample
>>> sample.avg(a)
2.0
>>> sample.avg(b)
2.0
>>>
```

ăIJlæ■d'ăNĖèĈĖĂZlăy■iijNă.sizeŌăŞŊ&a[Ō]ăĹĖăĹnăijTĉTlăTŗĉZĎăĖĈĉt'ăăylæTŗăŞŊăZTăśĈæN  
ĕr■æşT<double\*>&a[Ō]æTŻăjăæĂŌăăŭăŕĖæNĜĕŞĹĕ;ŋæ■ĉăyZăy■ăRŊĉZĎĉşăĎNăĂĈ  
ăL■ăĖRŖæYŗCăy■ĉZĎ avg() æŌĕăRŪăyĂăylæ■ĉĉăŏĉşăĎNĉZĎæNĜĕŞĹăĂĈ  
ăRĈĕĂĈăyNăyĂĕĹĈăĖşăžŌCythonăĖĖă■YĕĝĖăZĭĉZĎæZt'ĕnYĉĉĝĕŏşĕĕŕăĂĈ

```

    éZd'ázEad'DçREéAZâyçZDæTřčZDad'ÚiijNavg() çZDèfZâyłä;NäRèfYàsTçd'žžEäeCä;Tad'DçR
    èr■aRé with nogil: äçraeYÖäzEäyÄäyłä■éIJÄëAçILärsèC;æL'gëaNçZDäzççäAäIÜäÄC
    äIJlèfZâyłäIÜäy■iijNäy■èC;æIJL'äzzä;TçZDæZóéAZPythonärzèsäâATâATârleC;ä;fçTlècñäçraeYÖäyž
    cdef çZDärzèsäâSÑäG;æTřäÄC äRëad'ÚiijNad'ÚéCłäG;æTřäfEëazçÖrãðçZDäçraeYÖäóCäzñèC;äy■ä;Iè
    äZäa■d'iijNäIJlsample.pxdæÜGäzÜäy■iijNavg() ècñäçraeYÖäyž double avg(double
    *, int) nogil.

```

```

    ärzPointçZSædDä;ŞçZDad'DçREæYřäyÄäyłæNŠæLYäÄCæIJñèLCä;fçTlèCüäZLärzèsäârEPointärzèsä
    èeAèfZæäüäAZçZDèfIiijNäzTäsCCythonäzççäAçI■ä;öæIJL'çCzäd'■æICäÄC
    éeÜäELiijNäyNéIççZDärijaEëècñçTlæIëäijTäEëCäG;æTřäZŞäSÑPython
    APIäy■äóZäZL'çZDäG;æTřiijZ

```

```

from cpython.pycapsule cimport *
from libc.stdlib cimport malloc, free

```

```

    äG;æTřdel_Point() äŠNPoint() ä;fçTlèfZâyłäLşèC;æIëäLZäzzäyÄäyłèCüäZLärzèsäiijN
    äóCäijZäNëèçEäyÄäyłPoint * æNĞéŠLäÄCcdef del_Point() ärE del_Point()
    äçraeYÖäyžäyÄäyłäG;æTřiijN äRleC;éÄZèfGÇythonèðféÜöiijNèÄNäy■èC;äzÖPythonäy■èðféÜöäÄC
    äZäa■d'iijNèfZâyłäG;æTřärzäd'ÚéCłæYřäy■äRfègAçZDäATâATäóCècñçTlæIëä;ŞäAZäyÄäyłäZdèfCäG;æ
    äG;æTřèrCçTlærTäeC      PyCapsule_New()      äÄAPyCapsule_GetPointer()
    çZt'æÖëæIëèGHPython C APIäzÜäyTäzèäRñæäüçZDæÜzäijRècñä;fçTlæÄC

```

```

    distance äG;æTřäzÖ Point() äLZäzççZDèCüäZLärzèsäy■æRŘäRÜæNĞéŠLäÄC
    èfZéGÑèeAæşläDRçZDæYřä;äy■éIJÄëAæNëäfCäijCäyäd'DçREäÄC
    äeCædIJäyÄäyłèTZèfççZDärzèsäècñäijæèZæIëiijNPyCapsule_GetPointer()
    äijZæLZäGzäyÄäyłäijCäyÿiijN ä;EæYřCythonäüççZRçşèeAŞæÄÖäzLæşèæL;äLřäóCijNäzÜärEäóCäzÖ
    distance() äijäeÄŞäGzäÖzäÄC

```

```

    äd'DçREPointçZSædDä;ŞäyÄäyłçijçCzæYřäóCçZDäóðçÖræYřäy■äRfègAçZDäÄC
    ä;äy■èC;èðféÜöäzzä;TäsðæÄgæIëæşèçIJNäóCçZDäEëCłäÄC
    èfZéGÑæIJL'äRëad'ÜäyÄçg■ÜzæşTäÖzäNëèçEäóCijNärşæYřäóZäZL'äyÄäyłæL'äşTçşzädNijNäeCäyN

```

```

# sample.pyx

cimport csample
from libc.stdlib cimport malloc, free
...

cdef class Point:
    cdef csample.Point *_c_point
    def __cinit__(self, double x, double y):
        self._c_point = <csample.Point *> malloc(sizeof(csample.
        ↪Point))
        self._c_point.x = x
        self._c_point.y = y

    def __dealloc__(self):
        free(self._c_point)

    property x:
        def __get__(self):

```



## 17.11 15.11 CythonāĒŽénŸæĀğèĈĭçŽDæŦřçžDæŠ■äĭĬJ

### éŮóécŸ

äĭäëĒAāĒŽénŸæĀğèĈĭçŽDæŠ■äĭĬJæĪëèĠNumPyāžŦçšžçŽDæŦřçžDèőăçőŮăĠĭæŦřăĀĆ  
äĭăăŮšçžŦçššëĒAšăžĒCythonèĚæăŮçŽDăŮčăĒŮăĭjŽèđĲăđĈăŦŸăĬŮçđĀă■ŦĭĭjŦăĬĒŸŦăžŮăŸ■çăđăđŽèřăĀ

### èğĉăĒşæŮžæăĹ

äĭĬăŸžăŸĀăŸĲăĬŦă■ŦĭĭjŦăŸŦéĬçŽDăžčçăĒæĭjŦçđĲăžĒĒăŸĀăŸĬCythonăĠĭæŦřĭĭjŦçŦĲăĪăĲăđăĲăŦĲăŸĀă

```
# sample.pyx (Cython)

cimport cython

@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    '''
    Clip the values in a to be between min and max. Result in out
    '''
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same_
↪size")
    for i in range(a.shape[0]):
        if a[i] < min:
            out[i] = min
        elif a[i] > max:
            out[i] = max
        else:
            out[i] = a[i]
```

èĒAçĭjŮërŦăŦŦăđDăžžèĲŽăŸĲăĲĲăŦŦĭĭjŦăĬăĒĒĒăŸĀăŸĲăĲŦăŸŦéĬçĲŽæăŮçŽD  
setup.py æŮĠăžŮ ĭĭjĲăĬçŦĲ python3 setup.py build\_ext --inplace  
æĪëăđDăžžăđĈĭĭjŦĭjŽ

```
from distutils.core import setup
from distutils.extension import Extension
from Cython.Distutils import build_ext

ext_modules = [
    Extension('sample',
        ['sample.pyx'])
]

setup(
    name = 'Sample app',
```

```
cmdclass = {'build_ext': build_ext},
ext_modules = ext_modules
)
```

āĵāāijŽāŔŚçŎřçzŞæđIJaĜjæŦřçąóăóđărzæŦřçzĐèŁZèąŇçŽĐăŁóæ■čijŇăžűăyŦăŔřăzěéĂĆçŦlăžŎăđ'Žç

```
>>> # array module example
>>> import sample
>>> import array
>>> a = array.array('d', [1, -3, 4, 7, 2, 0])
>>> a

array('d', [1.0, -3.0, 4.0, 7.0, 2.0, 0.0])
>>> sample.clip(a, 1, 4, a)
>>> a
array('d', [1.0, 1.0, 4.0, 4.0, 2.0, 1.0])

>>> # numpy example
>>> import numpy
>>> b = numpy.random.uniform(-10, 10, size=1000000)
>>> b
array([-9.55546017,  7.45599334,  0.69248932, ...,  0.69583148,
        -3.86290931,  2.37266888])
>>> c = numpy.zeros_like(b)
>>> c
array([ 0.,  0.,  0., ...,  0.,  0.,  0.])
>>> sample.clip(b, -5, 5, c)
>>> c
array([-5.,          5.,          0.69248932, ...,  0.69583148,
        -3.86290931,  2.37266888])
>>> min(c)
-5.0
>>> max(c)
5.0
>>>
```

āĵăēŦŸăijŽāŔŚçŎřēŦŦřęŇçŦŦşæŁŦřçzŞæđIJeđăyŷçŽĐăŁăĂĆ  
ăyŇéĬăĹŚăžŇăŦĚăIJŇăĬŇăŦŦnumpyăy■çŽĐăűşă■ŸăIJĬçŽĐ clip()  
ăĜjæŦŦăŦăŦăyŦăyŦăĂĝēČjărzærŦijŽ

```
>>> timeit('numpy.clip(b, -5, 5, c)', 'from __main__ import b, c, numpy',
↳ number=1000)
8.093049556000551
>>> timeit('sample.clip(b, -5, 5, c)', 'from __main__ import b, c, sample
↳ ',
...       number=1000)
3.760528204000366
>>>
```

æ■čăēĆăjăçIJŇăĹŦřçŽĐijŇăőČēăĂăŁăăĬăđ'ŽăĂŦăĂŦēŁZæŸŦăyŦăyŦăĬăĹăIJĬēűčçŽĐçzŞæđIijŇăŽăă



## ěőłěőž

æIJñĚĹĈăĹĹ' ċŤlăžĚCythonċşzăđŇċŽĎăĚĚă■ŸĕğĚăŽĹ; ĩijŇăđĀăđ' ġċŽĎċŏĂăŇŮăžĚăŤŕċžĎċŽĎăŞă; I  
cpdef clip() âċŕăŸŎăžĚ clip() âŖŇăŮŮăŷžĈċžġăĹăĜ; æŤŕăžĕăŖĹPythonċžġăĹăĜ; æŤŕăĂĈ  
ăIJĹCythonăŷ■ĩijŇĕĤŽăŷĹăŸŕăĹĚĜ■ĕĚăĈŽĎĩijŇăŽăăŷžăŏĈĕăĹċđ' žă■đ' âĜ; æŤŕĕŕĈċŤĹĕĚăĤăŖăŤăĚŮăžŮCytho  
ĩijĹăŕŤăĚĈă; äăĈşăIJĹăŖĕăđ' ŮăŷŮăŷĹăŷ■ăŖŇċŽĎCythonăĜ; æŤŕăŷ■ĕŕĈċŤĹclip() ĩijĹăĂĈ

ċşzăđŇăŖĈăŤŕ double[:] a âŠŇ double[:] out  
âċŕăŸŎĕĤŽăžŽăŖĈăŤŕăŷžăŷĀċžŤ' ċŽĎăŖŇċş; âžĕăŤŕċžĎăĂĈ  
ă; IJăŷžĕ; ŞăĚĕĩijŇăŏĈăžŇăĭjŽĕŏĤĕŮŏăžžă; ŤăŏđċŎŕăžĚăĚĚă■ŸĕğĚăŽĹ; æŎĕăŖĈċŽĎăŤŕċžĎăŕžĕśăĩijŇĕĤŽăŷĹ  
3118æIJĹĕŕĕċžĚăŏŽăžĹăĂĈ âŇĚăŇŇăžĚNumPyăŷ■ċŽĎăŤŕċžĎăŠŇăĚĚċ; ŏċŽĎăŕŕăŷžăŞăĂĈ

ă; Şă; äċĭjŮăĚĈŤŤşăĹŖċžŞăđIJăŷžăŤŕċžĎċŽĎăžċċăĂăŮĩijŇă; äăžŤĕŕĕĕĂăĹăĹăŷĹĕĹċđ' žă; ŇĕĈĈăăŷĕ  
ăŏĈăĭjŽăŕĚăĹŽăžžĕ; ŞăĜăŤŕċžĎċŽĎĕŤ' ċăžžċžĚĕŕĈċŤĹĕĂĚĩijŇăŷ■ĕIJăĕĚăĈşĕĕĂşă; äăŞ■ă; IJċŽĎăŤŕċžĎċ  
ĩijĹăŏĈăžĚăžĚăĂĜĕŏ; æŤŕċžĎăŷşċžŖăĜĚăđ' Ĝăĕ; äžĚĩijŇăŖĹĕIJăĕĚăĂăŤăŷŮăăžŽăŕŖċŽĎăċĂăşĕăŕŤăĚĈăċă  
ăIJĹăĈŖNumPyăžŇċşċžĎăžŞăŷ■ĩijŇă; ĤċŤĹ numpy.zeros() æĹŮ numpy.  
zeros\_like() âĹŽăžžĕ; ŞăĜăŤŕċžĎċŽŷăŕžĕĂŇĕĹăŕŤĕ; ĈăŏžăŸşăĂĈăŖĕăđ' ŮĩijŇĕĚăĂăĹŽăžžăIJăĹăĹă  
ă; äăŖŕăžĕă; ĤċŤĹ numpy.empty() æĹŮ numpy.empty\_like() .  
ăĕĈăđIJă; äăĈşĕĕĚċŽŮăŤŕċžĎăĚĚăŏžă; IJăŷžċžŞăđIJċŽĎĕŕĹĕĂĹăŇĹ' ĕĤŽăŷŷđ' äŷĹăĭjŽăŕŤĕ; ĈăĤŇċĈăăĂĈ

ă; äă; äċŽĎăĜ; æŤŕăŏđċŎŕăŷ■ĩijŇă; äăŖĹĕIJăĕĚăĈşĕĕĂă■ŤċŽĎĕĂžĕĤĜăŷŇăăĜĕĹŖċŏŮăŠŇăŤŕċžĎăşĕă  
CythonăĭjŽĕŤ' şĕŤ' ċăŷžă; äċŤŤşăĹŖĕŇŸăŤĹċŽĎăžċċăĂăĂĈ

clip() âŏŽăžĹăăžŇăĹ■ċŽĎăŷđ' äŷĹĕĈĚĕĕŕăŽĹăŖŕăžĕăĭjŸăŇŮăŷŇăăĜĕĈ; äăĂĈ  
@cython.boundscheck(False) ċIJăăŎăžĚăĹăĂăIJĹċŽĎăŤŕċžĎĕŮĹċŤŇăċĂăşĕĕĩijŇ  
ă; Şă; äċşĕĕĂşăŷŇăăĜĕŏĤĕŮŏăŷ■ăĭjŽĕŮĹċŤŇċŽĎăŮŮăĂžăŖŕăžĕă; ĤċŤĹăŏĈăăĂĈ  
@cython.wraparound(False) æŮĹĚžđ' äžĚċŽŷăŕžăŤŕċžĎăŕžĕĹċĈċŽĎĕŤ' şăŤŕăŷŇăăĜċŽĎăđ' ĎċŖĚĩijŇ  
ăĭjŤăĚĕĕĤăŷđ' äŷĹĕĈĚĕĕŕăŽĹăŖŕăžĕăđĀăđ' ġċŽĎăŖŖă■ĜăăĜĕĈ; ĩijĹăŤŇĕŕŤĕĤŽăŷĹă; Ňă■ŖċŽĎăŮŮăĂžăđ'

ăžžă; ŤăŮŮăĂžăđ' ĎċŖĚăŤŕċžĎăŮŮĩijŇăċŤĹ' ŷăžŮăŤžăŮĎăžŤăşĈċŏŮăşŤăŖŇăăŮăŖŕăžĕăđĀăđ' ġċŽ  
ă; ŇăĕĈĩijŇĕĂĈĕŽŞăŕž clip() âĜ; æŤŕċŽĎăĈăŷŇăĤŏă■ċĩijŇă; ĤċŤĹăĹăžžĕăĹĕ; äĭjŖĩijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same_
↪size")
    for i in range(a.shape[0]):
        out[i] = (a[i] if a[i] < max else max) if a[i] > min else_
↪min
```

ăŏđĚĚăĤŇĕŕŤċžŞăđIJăŸŕĩijŇĕĤŽăŷĹĹăĹăIJŇċŽĎăžċċăĂăĕĹŖĕăŇĕĂşăžĕĕĚăĂăĤŇ50%ăžĕăŷĹĹăĭjĹ2.44ċş  
timeit() æŤŇĕŕŤċžŽĎ3.76ċşĖĩijĹăĂĈ

ăĹŕĕĤŽĚĜŇăŷžă■ċĩijŇă; äăŖŕĕĈ; æĈşċşĕĕĂşşĕĤċğ■ăžċċăĂăĂŎăžĹĕĈ; ĕŷşăĹăŇăĚĈĕŕ■ĕĹĂPKăŚċĭjş  
ă; ŇăĕĈĩijŇă; äăŖŕĕĈ; äĚăžĚăĈăŷŇċŽĎĈăĜ; æŤŕăžŮă; ĤċŤĹăĹăĕĹăĜăĕĹĈċŽĎăĹăIJăĹăĕĹăŇăĚăĹăŤă

```
void clip(double *a, int n, double min, double max, double *out) {
    double x;
```

```
for (; n >= 0; n--, a++, out++) {
    x = *a;

    *out = x > max ? max : (x < min ? min : x);
}
}
```

æĹŚāznæšqæIJĹ'āsTçd'žèfZäyĭçŽDæL'Ĺ'āsTāzççăĀiijNă;EæYrërTēĹNāzNăRŌiijNæĹŚāznăRŚçŌrăyĂă  
æIJĀāzTăyNçŽDăyĂèqNærTă;ăæCşesăçŽDèfRëaNçŽDăĹnă;Ĺăd'ŽăĂĆ

ă;ăăRfăzëărzăôďă;NăzççăĀæďDăzzăď'ŽăyĭæL'Ĺ'āsTăĂĆărzăžŌæ\$RăžZæTřçzDæ\$■ă;IJiijNæIJĀăë;èqA  
èqAèfZæăăAŽçŽDërĹiijNéIJĂèqAăĹŌæTžăzççăĀiijNă;ĹçTĹ with nogil: èĹ■ăRëiijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip(double[:] a, double min, double max, double[:] out):
    if min > max:
        raise ValueError("min must be <= max")
    if a.shape[0] != out.shape[0]:
        raise ValueError("input and output arrays must be the same_
↳size")
    with nogil:
        for i in range(a.shape[0]):
            out[i] = (a[i] if a[i] < max else max) if a[i] > min_
↳else min
```

ăqĈăďIJă;ăæCşăEŽăyĂăyĭæ\$■ă;IJăžNçzt'æTřçzDçŽDçĹĹæIJnĹiijNăyNéĹăYrăRfăzëărĈèĂĈăyNĹiijŽ

```
@cython.boundscheck(False)
@cython.wraparound(False)
cpdef clip2d(double[:, :] a, double min, double max, double[:, :]
↳out):
    if min > max:
        raise ValueError("min must be <= max")
    for n in range(a.ndim):
        if a.shape[n] != out.shape[n]:
            raise TypeError("a and out have different shapes")
    for i in range(a.shape[0]):
        for j in range(a.shape[1]):
            if a[i, j] < min:
                out[i, j] = min
            elif a[i, j] > max:
                out[i, j] = max
            else:
                out[i, j] = a[i, j]
```

ăyNæIJŽëržèĂĖăy■èqAăfYăžEæIJnĹĹĆæL'ĂæIJĹăzççăĀèĈ;ăy■ăiijŽçzŚăôŽăĹræ\$RăyĭçĹ'žăôŽæTřçzĹ  
èfZæăăăzççăĀăřsæŽt'æIJĹçĀtæt'zæĂgăĂĆăy■èfGĹiijNèqAæšĹæĐRçŽDæYrăqĈăďIJăď'ĐçRĖæTřçzDèqA  
èfZăžZăĖĖĂôžăăšçzRëŭĖăGzæIJnĹĹĆèNĈăŽt'iijNæŽt'ăď'ŽăĹæAřërŭăRĈèĂĈ PEP  
3118 iijNăřNæŬŭ CythonæŬGæqçăy■ăĖşăžŌăĀIJçşzăďNăĖĖă■YèqĖăŽ;ăĀĪ  
çřGăžşăĀiijă;ŬăyĂèřzăĂĆ

```
>>> from llvm.core import Module, Function, Type, Builder
>>> mod = Module.new('example')
```

```

>>> f = Function.new(mod, Type.function(Type.double(), \
                                     [Type.double(), Type.double()], False), 'foo')
>>> block = f.append_basic_block('entry')
>>> builder = Builder.new(block)
>>> x2 = builder.fmul(f.args[0], f.args[0])
>>> y2 = builder.fmul(f.args[1], f.args[1])
>>> r = builder.fadd(x2, y2)
>>> builder.ret(r)
<llvm.core.Instruction object at 0x10078e990>
>>> from llvm.ee import ExecutionEngine
>>> engine = ExecutionEngine.new(mod)
>>> ptr = engine.get_pointer_to_function(f)
>>> ptr
4325863440
>>> foo = ctypes.CFUNCTYPE(ctypes.c_double, ctypes.c_double, ctypes.
↳c_double)(ptr)

>>> # Call the resulting function
>>> foo(2, 3)
13.0
>>> foo(4, 5)
41.0
>>> foo(1, 2)
5.0
>>>

```

ázüäy■æÝřèrťǎIJlè£ŽäyłásĆéİççŁřäzEäzżä;TéŤŽèřřāřsāijŽārijeĠťPythoneğćcéĠŁāŽlæŇĆæŎLāĂĆ  
 èĕAèōřǎ;ŮčŽDæÝřǎ;ǎæÝřǎIJłçŽťæŎčèušæIJžāŽłçžġāŁńçŽDāĖĖā■ÝǎIJřǎĬāŠŇæIJňǎIJřæIJžāŽłçǎAæLŠā

## 17.13 15.13 äijäéĀŠNULLçzŠǎřçŽDā■ŮčņęäyšçzŻCāĠæŢřāžŠ

### éŮóécŸ

äjäèĕAāEŽäyĀäyłæLŦřāsŦæłāāĬŮiijŇéIJĀèĕAäijäéĀŠäyĀäyłNULLçzŠǎřçŽDā■ŮčņęäyšçzŻCāĠæŢřāž  
 äy■èĕĠiijŇǎ;ǎäy■æÝřǎ;ŁçǎđǎđŽæĀŎæǎüǎ;ĕçŦĬPythonçŽDUnicodeā■ŮčņęäyšǎŎžǎđđçŎřǎđČāĂĆ

### èğćăEşæŮzæǎĬ

èöyād'ŻCāĠæŢřāžŠāŇĖāŖñäyĀäzŽæŠ■ä;IJNULLçzŠǎřçŽDā■ŮčņęäyšiiijŇècñǎčřæÝŎçśzǎđŇäyž  
 char \*.èĂĆèŽŚăĕCăyŇçŽDĈāĠæŢřiiijŇæĹŚāzñçŦłæĬăĀŽæijŦçđ'žāŠŇæŦŇèřŦçŦłçŽDiiijŽ

```

void print_chars(char *s) {
    while (*s) {
        printf("%2x ", (unsigned char) *s);

        s++;
    }
}

```

```
printf("\n");
}
```

æd' aG; æTɾaijZæL' Ša' rēcnaïjæeŁZæIeā' ŰçņęäyšçZDærRäyłā' ŰçņęçZDā' AāĖ' eŁZāŁűealçd' žiijNēŁZ

```
print_chars("Hello"); // Outputs: 48 65 6c 6c 6f
```

āržāžŎaIJIPythonäy' ēŕČçTlēŁZæăũçZDcĀG; æTɾiiĵNā; āæIJL' āGăçğ' éĀL' æNl' āĀĆ  
éçŰāĖĹiĵNā; āāRfāžēéĀŽēŁGērČçTl PyArg\_ParseTuple()  
āžūāNĠāōŽāĀIyāĀIJē; nā' cčāAāæIēēZŔāŁűāōCāRlèČ; æŠ' ä; IJā' ŰèŁĆiĵNāeČäyNiiĴ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
    char *s;

    if (!PyArg_ParseTuple(args, "y", &s)) {
        return NULL;
    }
    print_chars(s);
    Py_RETURN_NONE;
}
```

çzŠædIJāG; æTɾçZDä; ŁççTlāŰZæŠTāeČäyNāĀĆäzTçzEēğCāršā; NāĖēäžENULLā' ŰèŁĆçZDā' Űçņęäyš

```
>>> print_chars(b'Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars(b'Hello\x00World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be bytes without null bytes, not bytes
>>> print_chars('Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' does not support the buffer interface
>>>
```

āeČædIJā; āæČšaijāeĀŠUnicodeā' ŰçņęäyšiiĴNāIJl PyArg\_ParseTuple()  
äy' ä; ŁççTlāĀIsāĀIJæaiĵāiĴRçāAiiĴNāeČäyNiiĴ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
    char *s;

    if (!PyArg_ParseTuple(args, "s", &s)) {
        return NULL;
    }
    print_chars(s);
    Py_RETURN_NONE;
}
```

ā; Šècā; ŁççTlçZDæŰūāĀŽiiĴNāōČaiĴZeĠlāŁlārEæL' ĀæIJL' ā' Űçņęäyšē; nā' cčāyžāžēNULLçzŠārçZDŰ  
8çijŰçāAāĀĆä; NāeČiĵZ

```

>>> print_chars('Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars('Spicy Jalape\u00f1o') # Note: UTF-8 encoding
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> print_chars('Hello\x00World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str without null characters, not str
>>> print_chars(b'Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: must be str, not bytes
>>>

```

æĈæđIĴăZăÿzæşŖăžZăŎşăŽăiĵŊă;ăĕAçŽt'æŎěă;ĲčŦĬ  
 PyObject \*            ĕĂŇăÿ■ĕĴ;ă;ĲčŦĬ            PyArg\_ParseTuple()            ĩĵŇ  
 äÿŇĕĲčŽĐă;Ŋă■ŖăŖŤă;ăăŤčđ'žăžEæĂŎăăüăžŎă■ŮĕĹĈăŤŇă■Ůĉņăÿšăŕžzèsăÿ■ăĉĂæşěăŤŇăŖŖăŖŮăÿ  
 char \*ăĳŦčŦĬĳĵŽ

```

/* Some Python Object (obtained somehow) */
PyObject *obj;

/* Conversion from bytes */
{
    char *s;
    s = PyBytes_AsString(o);
    if (!s) {
        return NULL;    /* TypeError already raised */
    }
    print_chars(s);
}

/* Conversion to UTF-8 bytes from a string */
{
    PyObject *bytes;
    char *s;
    if (!PyUnicode_Check(obj)) {
        PyErr_SetString(PyExc_TypeError, "Expected string");
        return NULL;
    }
    bytes = PyUnicode_AsUTF8String(obj);
    s = PyBytes_AsString(bytes);
    print_chars(s);
    Py_DECREF(bytes);
}

```

ăĹ■ĕĲăÿđ'ĉğ■ĕ;ŋă■ĕĕĴ;ăŖŖăžzĕĉăŏăĲăĲŕŇŮĲĲčžŤăŕĴ;ĉŽĐăŦŖă■ŏĳĵŇ  
 ä;EæŸŖăŏĈăžŋăžăÿüă■ăĉĂæşěă■Ůĉņăÿšăÿ■ĕŮ'æŸŖăŖĕăŦŇăĒĕăžĒŮĲă■ŮĕĹĈăĂĈ  
 ăŽăă■đ'ĳĵŇăĲæĈăđIĴĕĲZăÿĲă;ĲĕĴ■ĕĕAçŽĐĕŦĳĵŇĒĕĈă;ăĕIĴăĕĕAĕĴăăŮăŮăŎăăĂăŽăĉĂæşěăžEăĂĈ



```
>>> import ctypes
>>> lib = ctypes.cdll.LoadLibrary("./libsamle.so")
>>> print_chars = lib.print_chars
>>> print_chars.argtypes = (ctypes.c_char_p,)
>>> print_chars(b'Hello World')
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>> print_chars(b'Hello\x00World')
48 65 6c 6c 6f
>>> print_chars('Hello World')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ctypes.ArgumentError: argument 1: <class 'TypeError'>: wrong type
>>>
```

```
>>> print_chars('Hello World'.encode('utf-8'))
48 65 6c 6c 6f 20 57 6f 72 6c 64
>>>
```

17.14 15.14 äijäéĀŠUnicodeā■ŮčņäÿšçzŻĆăĜǰæŦřăžŞ

ä;äëAåEZäyÄäy!æL!ʼaſT!æ!a!iU!i!jNéIJ!ÄëA!ä!rE!äy!Ä!äy!P!y!tho!n!■Ü!ç!ñ!ä!y!ſ!ä!i!j!ä!é!Ä!Š!ç!ž!C!ç!ž!D!æ!ſ!Ř!ä!y!l!ä!ž!Š!

èƒẸǾŊæĹŚazñéIJĂèèAèĂCèŽŠă; ĹLad' ŽćŽĐeŮőécÿİijNă; EæYřæIJĂäyzèèAçŽĐeŮőécÿæYřćŎřă■Yç  
ăŽăæ■d' iijNă; ăcŽĐæŇSæĹYæYřărEPythonă■Ůčņäyşë; ñæ■căyžăyĂäylêÇ; êcnCcREğğççŽĐă; cáijRăĂĆ

```
void print_chars(char *s, int len) {
    int n = 0;

    while (n < len) {
        printf("%2x ", (unsigned char) s[n]);
        n++;
    }
}
```



```

    }
    printf("\n");
}

void print_wchars(wchar_t *s, int len) {
    int n = 0;
    while (n < len) {
        printf("%x ", s[n]);
        n++;
    }
    printf("\n");
}

```

árzāžŎéíĉāŘŠā■ŮēŁĆçŽĎǦ;æŦřprint\_chars() ĩjNă;ăéIJĂèēAăřEPythonă■Ůçñēăÿšè;ñă■ćăÿžă.  
8. äÿNéÍĉæŸřăÿĂăÿłēŁŽæăŭçŽĎæL'l'ásŦǧ;æŦřă;Nă■ŘiijŽ

```

static PyObject *py_print_chars(PyObject *self, PyObject *args) {
    char *s;
    Py_ssize_t len;

    if (!PyArg_ParseTuple(args, "s#", &s, &len)) {
        return NULL;
    }
    print_chars(s, len);
    Py_RETURN_NONE;
}

```

árzāžŎéĆĉăžŽéIJĂèēAăđ'ĐçŘEæIJžǧÍæIJňăIJř wchar\_t  
çşzādNçŽĎăžŞǧG;æŦřiijNă;ăăŘřăžăăČŘăÿNéÍĉèŁŽæăŭçijŮăEŽæL'l'ásŦăžĉăĂiijŽ

```

static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
    wchar_t *s;
    Py_ssize_t len;

    if (!PyArg_ParseTuple(args, "u#", &s, &len)) {
        return NULL;
    }
    print_wchars(s, len);
    Py_RETURN_NONE;
}

```

äÿNéÍĉæŸřăÿĂăÿłăžđ'ăžŠăijŽērÍæİēăijŦçđ'žèŁŽăÿłăǦ;æŦřăŸřăēĆă;Ŧăŭēă;IJçŽĎiijŽ

```

>>> s = 'Spicy Jalape\u00f1o'
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f
>>> print_wchars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 f1 6f
>>>

```

ăžŦçŻEğĆărşēŁŽăÿłēĉāŘŠā■ŮēŁĆçŽĎǧ;æŦř print\_chars()

æYřæĀŌæăăæŌěăRŪUTF-8çijŪçăAæŤřæ■ŏçŽĎiijŇ äžěăŘĽ print\_wchars()  
æYřæĀŌæăăæŌěăRŪUnicodeçijŪçăAăĀijçŽĎ

## ěóíěőž

ăIJĺçžğçž■æIJñèĽCăžŇăĽ■iijŇă;ăăžŤěřěēŪăĒĽă■ęăžăă;ăëŏĕŕŪŏçŽĎCăĜ;æŤřăžŤççŽĎçĽ'žă;AăĀĆ  
ăržăžŌă;Ľăđ'ŽCăĜ;æŤřăžŤiijŇěĂŽăyăiăiăēĂŖă■ŪēĽCěĂŇăy■æYřă■ŪçņęăyŝăijŽăřŤē;Čăē;ăžZăĀĆēēAēē

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {  
    char *s;  
    Py_ssize_t len;  
  
    /* accepts bytes, bytearray, or other byte-like object */  
    if (!PyArg_ParseTuple(args, "y#", &s, &len)) {  
        return NULL;  
    }  
    print_chars(s, len);  
    Py_RETURN_NONE;  
}
```

ăęĆăđIJă;ăăž■ĎŮēŤYæYřæČŝēēAăijăēĂŖă■ŪçņęăyŝiijŇ  
ă;ăēIJăēēAçŝēēAŖPython 3ăŘřă;ĕçŤĺăyĂăyĺăŘĽĂĆçŽĎă■Ūçņęăyŝēăĭçđ'žiiŇ  
ăŏČăžŭăy■çŽt'æŌēæYăărĎăĽăřă;ĕçŤĺăăĜăĜĒçŝăđŇ char \* æĽŪ  
wchar\_t \* iijĽăŽt'ăđ'ŽçžĒēĽCăŖCěĂĆPEP 393iijĽçŽĎCăĜ;æŤřăžŤăĀĆ  
ăŽăă■đ'iijŇēēAăIJĽCăy■ēăĭçđ'žēŤŽăyĺă■ŪçņęăyŝæŤřæ■ŏiijŇăyĂăžŽē;Ňă■çēŤYæYřăĒĒēăžēēAçŽĎăĀĆ  
ăIJĽPyArg\_ParseTuple() äy■ă;ĕçŤĺăĂĽŝăĀĽăŝŇăĂĽu#ăĂĽăăijăijŖăŇŪçăAăŖăžēăăŏĽăĒĽçŽĎăĽğēă

ăy■ēŤĜēŤŽçğ■ē;Ňă■çæIJĽăyĭçijžçČžărŝæYřăŏČăŖřēČ;ăijŽărĭjēĜt'ăŌŝăğŇă■ŪçņęăyŝăržēsăçŽĎăržărý  
ăyĂăŪēē;Ňă■çēŤĜăŖŌiijŇăijŽăēIJĽăyĂăyĽē;Ňă■çæŤřæ■ŏçŽĎăđ'■ăĽŭēŽĎăĽăăĽăŖăŌŝăğŇă■Ūçņęăyŝăržēs  
ă;ăăŖăžēēğČărŝăyŇēŤŽçğ■æŤĽăđIJiijŽ

```
>>> import sys  
>>> s = 'Spicy Jalape\u00f1o'  
>>> sys.getsizeof(s)  
87  
>>> print_chars(s)  
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f  
>>> sys.getsizeof(s)  
103  
>>> print_wchars(s)  
53 70 69 63 79 20 4a 61 6c 61 70 65 f1 6f  
>>> sys.getsizeof(s)  
163  
>>>
```

ăržăžŌărŖŖēĜŖçŽĎă■ŪçņęăyŝăržēsăiijŇăŖřēČ;æŝăăžĂăžĽă;ŝăŖ■iijŇ  
ă;ĒæYřăēĆăđIJă;ăēIJăēēAăIJăĽĽ'ăŝŤăy■ăđ'ĎçŖĒăđ'ğēĜŖçŽĎăŪĜăēIJñiijŇă;ăăŖřēČ;æČŝēĂăĒēŤŽăyĽ  
ăyŇēĽăēYřăyĂăyĺăŤŏēŏççĽĽăēIJăŖăžēēĂăăēŤŽçğ■ăĒă■Yă■ŝēĂŪiijŽ

```

static PyObject *py_print_chars(PyObject *self, PyObject *args) {
    PyObject *obj, *bytes;
    char *s;
    Py_ssize_t len;

    if (!PyArg_ParseTuple(args, "U", &obj)) {
        return NULL;
    }
    bytes = PyUnicode_AsUTF8String(obj);
    PyBytes_AsStringAndSize(bytes, &s, &len);
    print_chars(s, len);
    Py_DECREF(bytes);
    Py_RETURN_NONE;
}

```

èĀŅārẏ wchar\_t çŽĎād'ĐçŘĚæŮüæČşèĕAéAǵăĚ■ăĚĚă■Ÿæ■şèĀŮārsæZt'ăĽăéŽĭăĽđăžĚăĀĆ  
 āĬĬăĚĚēČĭijŅPythonă;ǵçŦĭæĬĬĀēñŸæŦĬçŽĎēāǵd'žæĬēă■ŸăĆĬă■ŮçņęäÿşăĀĆ  
 äĭŅăĚĈijŅăŦĭăŅĚăŦŅASCIIçŽĎă■Ůçņęäÿşēćŋă■ŸăĆĬăÿžă■ŮēĽĆæŦřçzĎĭijŅ  
 èĀŅăŅĚăŦŅēŅČăZt'ăžŮŮ+0000ăĽŦŮ+FFFFçŽĎă■ŮçņęçŽĎă■Ůçņęäÿşă;ǵçŦĭăŦŅă■ŮēĽĆēāǵd'žăĀĆ  
 çŦŦsăžŮārżăžŮæŦřæ■ŮççŽĎēāǵd'žă;ćăĭjŦăÿ■æŸŦă■ŦăÿĀççŽĎĭijŅă;ăäÿ■ēČ;ăŦĚăĚĚēČĬæŦřçzĎēĭŋæ■ćăÿž  
 wchar\_t \* çĎŮăŦŮăĬJşæĬJZăŮČēČ;æ■ćçăŮççŽĎăŮēă;ĬĬăĀĆ äĭăăžŦēŦēăĽZăžžăÿĀăÿĭ  
 wchar\_t æŦřçzĎăžŮăŦŦŦăĚŮăÿ■ăđ'■ăĽŮæŮĜæĬŋăĀĆ PyArg\_ParseTuple()  
 çŽĎăĬŮ#ăĬăĬăĭjăĭjŦçăĀăŦŦăžēăÿŮăĽĭă;ăēñŸæŦĬçŽĎăŮŅăĽŦăŮČĭijĬăŮČăŦĚăđ'■ăĽŮççççđĬĚŽĎăĽăăĽŮ

âĚĆăđĬĬă;ăæČşēĀǵăĚ■ēŦĬæŮŮēŮŦ'ăĚĚă■Ÿæ■şèĀŮĭijŅă;ăăŦŦăÿĀççŽĎēĀĽæŦŦ'ăŦŦæŸŦăđ'■ăĽŮUnicode  
 ăŦĚăŮČăĭjăēĀŞççççÇăĜ;æŦřĭijŅçĎŮăŦŮăZđæŦŮēŦZăÿĭæŦřçzĎççŽĎăĚĚă■ŸăĀĆăÿŦēĬæŸŦăÿĭăŦŦēČ;çž

```

static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
    PyObject *obj;
    wchar_t *s;
    Py_ssize_t len;

    if (!PyArg_ParseTuple(args, "U", &obj)) {
        return NULL;
    }
    if ((s = PyUnicode_AsWideCharString(obj, &len)) == NULL) {
        return NULL;
    }
    print_wchars(s, len);
    PyMem_Free(s);
    Py_RETURN_NONE;
}

```

āĬĬēŦZăÿĭăŮđçŮŦăÿ■ĭijŅPyUnicode\_AsWideCharString()  
 āĽZăžžăÿĀăÿĭăÿt'æŮŮççŽĎwchar\_tçĭjŞăĚşăžŮăđ'■ăĽŮæŦřæ■ŮēŦZăŮŦăĀĆ  
 ēŦZăÿĭçĭjŞăĚşēćŋăĭjăēĀŞççççÇçĎŮăŦŮŮēćŋēĜĽæŦĭ;æŮĽăĀĆ äĭĚăŸŦăĽŦŦŦăŦŦçççççŽĎæŮŮăĀZĭijŅē

âĚĆăđĬĬă;ăçşēēĀŞÇăĜ;æŦřăžŞēĬĬăēæççŽĎă■ŮēĽĆçĭjŮçăĀăžŮăÿ■æŸŦUTF-8ĭijŅ  
 äĭăăŦŦăžēăĭjžăĽŮPythonă;ǵçŦĭăĽŦ'ăşŦçăĀăĬēăĽĝēăŦŦæ■ćçăŮççççŽĎēĭŋæ■çĭijŅăŦŦŦăČŦăÿŦēĬēŦZăăŮĭijZ

```
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
    char *s = 0;
    int len;
    if (!PyArg_ParseTuple(args, "es#", "encoding-name", &s, &len)) {
        return NULL;
    }
    print_chars(s, len);
    PyMem_Free(s);
    Py_RETURN_NONE;
}
```

æIJĀāŔŌīījŅāēĈăđIJă;ăăĈşçŽŕ æŌēăđ'ĐçŘĚUnicodeă■ŮčņăÿšīījŅăÿŅéĭćçŽĐæŸřă;Ņă■ŘīījŅăījŤç

```
static PyObject *py_print_wchars(PyObject *self, PyObject *args) {
    PyObject *obj;
    int n, len;
    int kind;
    void *data;

    if (!PyArg_ParseTuple(args, "U", &obj)) {
        return NULL;
    }
    if (PyUnicode_READY(obj) < 0) {
        return NULL;
    }

    len = PyUnicode_GET_LENGTH(obj);
    kind = PyUnicode_KIND(obj);
    data = PyUnicode_DATA(obj);

    for (n = 0; n < len; n++) {
        Py_UCS4 ch = PyUnicode_READ(kind, data, n);
        printf("%x ", ch);
    }
    printf("\n");
    Py_RETURN_NONE;
}
```

ăIJĭēŹăÿĭăžčçăĀăÿ■īījŅPyUnicode\_KIND()      ăŠŅ      PyUnicode\_DATA()  
 èŹăÿđ'ăÿĭăŏŔăŠŅUnicodeçŽĐăŔŕăŔŸăŏ;ăžăă■ŸăĈĭæIJĭăĔşīījŅēŹăÿĭăIJĭPEP  
 393ăÿ■æIJĭæŔŔēŹŕăĈ kind âŔŸéĠŔçīījŮčăĀăžŤăşĈă■ŸăĈĭīījĭ8ă;■ăĀĀ16ă;■ăĬŮ32ă;■īījĭ'ăžăăŔĭæŅŏ  
 ăIJĭăŏđéŽĔæĈĔăĔŕăÿ■īījŅă;ăăžŭăÿ■ēIJĭăēĀçşēéĀşăžă;ŤêŭşēŹăžŽăĀījæIJĭăĔşçŽĐăÿIJēŹīījŅ  
 âŔĭēIJăēĀăIJĭæŔŔăŔŮă■ŮčņăçŽĐæŮŭăĂžăŕĔăŏĈăžŅăījăçžŽ      PyUnicode\_READ()  
 ăŏŔăĈĈ

ēŹŸæIJĭ'æIJĭăŔŌăĠăăŔēīījŽă;ŞăžŌPythonăījăēĂŞUnicodeă■ŮčņăÿşçžŽÇçŽĐæŮŭăĂŽīījŅă;ăăžŤēră  
 ăēĈăđIJăIJĭUTF-8ăŠŅăŏ;ă■Ůčņăÿđ'çğ■ēĂĭæŅĭ'īījŅērŭēĂĭæŅĭ'UTF-8.      âŕžUTF-  
 8çŽĐæŤŕæŅĀæŽŕăĭĭăăŹŏéĀ■ăÿĂăžŽīījŅăžşăÿ■ăŏžæŸŞçĭŕéŤŽīījŅēğçéĠăŽĭăžşēĈ;æŤŕæŅĀçŽĐæŽŕăæ  
 æIJĀāŔŌīījŅçăŏăŹă;ăăžŤçžĔēŸĔērăžăĔăĔşăžŌăđ'ĐçŘĚUnicodeçŽĐçŽÿăĔşæŮĠăæç

## 17.15 15.15 CāŨčņęäyšē;ñæ■cäyžPythonāŨčņęäyš

### ēŨóécŸ

æĀŌæāüăŕĒCäy■çŽDāŨčņęäyšē;ñæ■cäyžPythonāŨčŁĆăĹŨäyĀäyĭāŨčņęäyšăŕžēsăijš

### èğcāEşæŨzæqĹ

CāŨčņęäyšă;ŁçŦĭăyĀăŕž char \* āŠŇ int æĭēēāĹcd' žiijŇ  
ă;ăēĪĴăēĒAăEşăŏŽăŨčņęäyšăĹŕăžŦæŸŕçŦĭăyĀäyĭāŌšăġŇāŨčŁĆăŨčņęäyšēŁŸæŸŕăyĀäyĭUnicodeăŨč  
ăŨčŁĆăŕžēsăăŕŕăžēăĈŕăyŇéĭcēŁŽăăüă;ŁçŦĭ Py\_BuildValue() æĭēăđDăžžăijŽ

```
char *s; /* Pointer to C string data */
int len; /* Length of data */

/* Make a bytes object */
PyObject *obj = Py_BuildValue("y#", s, len);
```

ăēĈăđĪă;ăēĒAăĹŽăžžăyĀäyĭUnicodeăŨčņęäyšijŇăžüăyŦă;ăçšēēAş s  
æŇĠăŕŖŖŖăžEUTF-8çijŨčăAçŽDæŦŕæ■ōijŇăŕŕăžēă;ŁçŦĭăyŇéĭcēŁŽDæŨzăijŖijŽ

```
PyObject *obj = Py_BuildValue("s#", s, len);
```

ăēĈăđĪ s ä;ŁçŦĭăĒŭăžŨçijŨčăAæŨzăijŖijŇéĈăžĹăŕŕăžēăĈŕăyŇéĭcă;ŁçŦĭ  
PyUnicode\_Decode() æĭēăđDăžžăyĀäyĭāŨčņęäyšijŽ

```
PyObject *obj = PyUnicode_Decode(s, len, "encoding", "errors");

/* Examples */
obj = PyUnicode_Decode(s, len, "latin-1", "strict");
obj = PyUnicode_Decode(s, len, "ascii", "ignore");
```

ăēĈăđĪă;ăēAŕăē;ăĪĹăyĀäyĭŁçŦĭ wchar\_t \*, len ŕŕžēāĹcd'žçŽDăŏ;ăŨčņęäyšijŇ  
ăĪĹăĠăçģģ■ēĀĹæŇŦæĀģăĀĈēēŨăĒĹă;ăăŕŕăžēă;ŁçŦĭ Py\_BuildValue() ijŽ

```
wchar_t *w; /* Wide character string */
int len; /* Length */

PyObject *obj = Py_BuildValue("u#", w, len);
```

ăŖēăđŦijŇă;ăēŁŸăŕŕăžēă;ŁçŦĭ PyUnicode\_FromWideChar() :

```
PyObject *obj = PyUnicode_FromWideChar(w, len);
```

ăŕžăžŌăŏ;ăŨčņęäyšijŇăžüăēsăăĪĹăŕžăŨčņęæŦŕæ■ŏēŁŽăŇēğçăđŖăĀŦăĀŦăŏĈēcŇăAĠăŏŽăŸŕăŌ;

## ěóíěőž

ărĖCăy■çŽĎă■Ůčņęäyšë;ñæ■ćäyžPythonă■ŮčņęäyšëAṭā;łăŠŃĬ/OăŔŃăăũçŽĎăŎșăĹZăĂĆ  
ăžšăřsăŸřèřt'īijŃăĭēēĜĬCăy■çŽĎăŤŕă■ōăĤĖéązæăžæ■ōăyĂăžŽēğççăAăŽĭécăŸĭăijŔçŽĎēğççăAăyžăyĂă  
éĂŽăyŷçijŮčăAăăijăijŔăŃĖăŃŃASCIIăĂĂLatin-1ăŠŃUTF-8.  
ăĕĆăđĬJă;ăăžŭăy■çăăőăŎŽçijŮčăAăŮžăijŔăĹŮēĂĖăŤŕă■ōăŸŕăžŃēĤZăĹŮçŽĎīijŃă;ăăĬĂăăĕ;ărĖă■Ůčņęäy  
ă;ŠăđĎéĂăăyĂăyĭăŕžsăçŽĎăŮŭăĂŽīijŃPythonéĂŽăyŷăijŽăđ'■ăĹŭă;ăăŔŔă;ŽçŽĎă■ŮčņęäyšăŤŕă■ōăĂĆ  
ăĕĆăđĬJăĬĬăĤĖēĕAçŽĎĕŕĭijŃă;ăĕĬĂĕĕAăĬĬăŔŎĭĕăŎžéĜĹăŤ;Că■ŮčņęäyšăĂĆ  
ărŃăŮŭīijŃăyžăžĖēŏĭ'çĭŃăžŔăŽt'ăĹăăAăĕăčŏīijŃă;ăăžŤĕŕăărŃăŮŭă;ĤçŤĭăyĂăyĭăŃĜĖŠĹăŠŃăyĂăyĭăđ'ğ  
ĕĂŃăy■ăŸŕă;ĭĕŭNULLçžšăŕ;ăŤŕă■ōăĭăĹZăžă■ŮčņęäyšăĂĆ

## 17.16 15.16 äy■çăăőăŎŽçijŮčăAăăijăijŔçŽĎCă■Ůčņęäyš

### éŮŏécŸ

ă;ăĕĕAăĬĬăCăŠŃPythonçŽt'ăŎŏăĭăăŽĎē;ñæ■ćă■ŮčņęäyšīijŃă;ĖăŸŕCăy■çŽĎçijŮčăAăăijăijŔăžŭăy■ç  
ă;ŃăĕĆīijŃăŔŕĕĈ;Căy■çŽĎăŤŕă■ōăĬJšăĬJZăŸŕUTF-8īijŃă;ĖăŸŕăžŭăšăĖĬĬăijžăĹăăŏĈăĤĖéązæŸŕăĂĆ  
ă;ăăĈççijŮăĖŽăžççăAăĭăăžĕăyĂçğ■ăijŸĕŽĖçŽĎăŮžăijŔăđ'ĎçŔĖĕĤZăžŽăy■ăŔĹăăijăŤŕă■ŏīijŃēĤZăăŭă

## ěğçăĖşşăŮžăăĹ

ăyŃĕĭăŸŕăyĂăžŽCçŽĎăŤŕă■ōăŠŃăyĂăyĭăĜ;ăŤŕăĭăăijŤçđ'žĕĤZăyĭéŮŏécŸīijŽ

```
/* Some dubious string data (malformed UTF-8) */
const char *sdata = "Spicy Jalape\xc3\xbf\xae";
int slen = 16;

/* Output character data */
void print_chars(char *s, int len) {
    int n = 0;
    while (n < len) {
        printf("%2x ", (unsigned char) s[n]);
        n++;
    }
    printf("\n");
}
```

ăĬĭĕĤZăyĭăžççăAăy■īijŃă■Ůčņęäyš sdata äŃĖăŔŃăžĖUTF-  
ğăŠŃăy■ăŔĹăăijăŤŕă■ōăĂĆ äy■ĕĤĜīijŃăĕĆăđĬçŤĭăĹăăĬĬăCăy■ĕŕĈçŤĭ  
print\_chars(sdata, slen) īijŃăŏĈçijžĕĈ;ă■ćăyŷăŭă;ĬJăĂĆ  
çŎŕăĬĬăĂĜĕŏ;ă;ăăĈşăŕĖ sdata çŽĎăĖĖăŏžē;ñæ■ćäyžăyĂăyĭPythonă■ŮčņęäyšăĂĆ  
ĕĤZăyĂă■ăĂĜĕŏ;ă;ăăĬĬăŔŎĭĕĕĖŸăĈşĕĂŽĕĤĜăyĂăyĭăĹĭ'ăšŤăŕĖĕĈçăyĭă■Ůčņęäyšăijăăyĭ  
print\_chars() äĜ;ăŤŕăĂĆ äyŃĕĭăŸŕăyĂçğ■çŤĭăĭăĖĭăĹđ'ăŎșăğŃăŤŕă■ŏçŽĎăŮžăşŤīijŃăŕşçŏŮă

```
/* Return the C string back to Python */
static PyObject *py_retstr(PyObject *self, PyObject *args) {
    if (!PyArg_ParseTuple(args, "")) {
```

```

        return NULL;
    }
    return PyUnicode_Decode(sdata, slen, "utf-8", "surrogateescape");
}

/* Wrapper for the print_chars() function */
static PyObject *py_print_chars(PyObject *self, PyObject *args) {
    PyObject *obj, *bytes;
    char *s = 0;
    Py_ssize_t len;

    if (!PyArg_ParseTuple(args, "U", &obj)) {
        return NULL;
    }

    if ((bytes = PyUnicode_AsEncodedString(obj, "utf-8",
↪ "surrogateescape"))
        == NULL) {
        return NULL;
    }
    PyBytes_AsStringAndSize(bytes, &s, &len);
    print_chars(s, len);
    Py_DECREF(bytes);
    Py_RETURN_NONE;
}

```

æĈædIJä;ääIJPythonäy■ärIerTēfZāzZāG;æTrijNäyNélcæYrèfRèaÑæTLædIJijZ

```

>>> s = retstr()
>>> s
'Spicy JalapeÃso\udcae'
>>> print_chars(s)
53 70 69 63 79 20 4a 61 6c 61 70 65 c3 b1 6f ae
>>>

```

āzTçzEëgĈârşçzŞædIJä;ääijZāRŚçŌriijNäy■āRLæaijā■ŪçņäyşēcñcijŪçāAālRāyÄäyPythonā■Ūçņäy  
 āzūāyTā;ŞāōCēcñāZdaijaçzZCçZDæŪūāĀZrijNēcñē;ñæ■cāyZāŞNāzNāL■āŌşāgŊCā■ŪçņäyşäyĀæūçZDā

èóIèőž

æIJnèLCāsTçd'zāzEāIJæL'l'āsTæIqāIŪäy■ād'DçREā■ŪçņäyşæŪūaijZēĒ■āLrçZDāyÄäyIæcYæLŊāRl  
 āzşārşæYrèft'ijNāIJæL'l'āsTāy■çZDcā■ŪçņäyşāRrēČ;äy■āijZāyēæaijéAṭā;IPythonæL'ĀæIJşæIJZçZDUn  
 āZāæ■d'tijNā;LāRrēČ;äyĀāzZāy■āRLæaijCæTṛæ■ōaijæĀŞāLrPythonäy■āŌzāĀĈ  
 äyÄäyIā;Lāē;çZDā;Nā■RārşæYræūL'āRLāLrāzTāşCçşçzçşērČçTlærTāeCæŪGāzūāR■ēfZæūçZDā■Ūçņäy  
 ā;NāeČriijNāeCædIJäyÄäyIçşçzçşērČçTlēfTāZdçzZēgçéGLāZlāyÄäyIæ■şāIRçZDā■ŪçņäyşriijNäy■ēČ;ēcñā

äyĀēLñæIēēōsriijNāRfāzēēĀZēfGālŪāōZāyĀāzZēTZèrrç■ŪçTærfTāeCāyēæaijāĀĀāf;çTēāĀĀæZfāzç  
 äy■ēfGriijNēfZāzZç■ŪçTēçZDāyÄäyIçijžçCzæYrāōCāznæryāzĒæĀgçātāIRāzEāŌşāgŊā■ŪçņäyşçZDāĒĒ  
 ā;NāeČriijNāeCædIJä;Nā■Rāy■çZDāy■āRLæaijæTṛæ■ōā;çTlēfZāzZç■ŪçTēāzNäyĀēgççāAriijNā;ääijZā;Ū

```
>>> raw = b'Spicy Jalape\xc3\xbl\xae'
>>> raw.decode('utf-8', 'ignore')
'Spicy JalapeÃso'
>>> raw.decode('utf-8', 'replace')
'Spicy JalapeÃso?'
>>>
```

surrogateescape éTŽérřád'ĐčŘĚč■ŮčTěäijŽärĚæL'ÄæIJL'äy■äRřěğččäAä■ÜèLCè;ňäNŮäyžäyÄä  
ä;NäčĆijŽ

```
>>> raw.decode('utf-8', 'surrogateescape')
'Spicy JalapeÃso\uudcae'
>>>
```

■TčNňčŽDä;Ůä;■äzččŘĚä■ŮčñæærTäčē \udcae äIJUni-  
codeäy■æYřéIdæšTčŽDäÄČ äŽäæ■d'rijNèčŽäylä■ŮčñæäyšärsæYřäyÄäyléIdæšTèäłčd'žäÄČ  
äödéŽĚäyLüijNäčCædIJä;äärĚäöČäijääyläyÄäylæL'gèäNè;ŠäGžčŽDäG;æTrijNä;ääijŽä;ÜäLräyÄäyléTŽérř

```
>>> s = raw.decode('utf-8', 'surrogateescape')
>>> print(s)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
UnicodeEncodeError: 'utf-8' codec can't encode character '\udcae'
in position 14: surrogates not allowed
>>>
```

čDűëÄNrijNäĚæöyžäččŘĚè;ňæ■ččŽDäĚséTōčČzäIJläžŮäžŮČäijäčžŽPythonäRĹLäZđäijäčžŽCčŽDäy■  
ä;ŠèčŽäylä■ŮčñæäyšäĚ■æňä;ččTí surrogateescape çijŮčäAæŮürijNäzččŘĚä■ŮčñæäijŽè;ňæ■čäŽđäŮ

```
>>> s
'Spicy JalapeÃso\uudcae'
>>> s.encode('utf-8', 'surrogateescape')
b'Spicy Jalape\xc3\xbl\xae'
>>>
```

ä;IJäyžäyÄèLňäĚĚäLŽrijNæIJÄäč;éAčäĚ■äzččŘĚçijŮčäAäÄTäÄTäēCædIJä;äæ■ččäöčŽDä;ččTíäžĚçij  
äy■ēčGrijNæIJL'æŮüäÄŽčäöäöđäijŽäGžčŮřä;ääzüäy■ēČ;æŮğäLüæTřæ■öçijŮčäAäzüäyTä;ääRĹäy■ēČ;äč;  
éČčäzLärsäRřäžä;ččTíæIJñèLCčŽDæLÄæIJräžĚäÄČ

æIJäÄRŮäyÄčČžèAæšIæĐŘčŽDæYřijNPythonäy■èöyäd'ŽéIčäRŠčšzčžščŽDäG;æTrijNçL'zäLnäYřä  
éČ;äijŽä;ččTíäzččŘĚçijŮčäAäÄČä;NäčĆijNäčCædIJä;ää;ččTíäČR os.listdir()  
èčŽæäüčŽDäG;æTrijN äijääĚäyÄäyläNĚäRnäžĚäy■äRřěğččäAæŮGäzüäR■čŽDčŽöä;TčŽDërrijNäöČäijŽ  
äRČèÄČ5.15čŽDčŽyäĚšçnäèLCäÄČ

PEP 383 äy■æIJL'æŽt'äd'ŽäĚšäžŮæIJnäIJžæRŘäLřčŽDäžäRĹLäŠNsurroga-  
teescapeéTŽérřád'ĐčŘĚčŽyäĚščŽDäčæAřäÄČ



## 17.17 15.17 äijäéĀŠæŮĜäzúâĤčžŻCæL'ŕásŤ

### éŮóécŸ

ä;äéIJĀèĕAâĤŖŚCăžŞăĜ;æŤrăijäéĀŠæŮĜäzúâĤĤijŇă;EæŸréIJĀèĕAçăőăĬæŮĜäzúâĤĤæăžæĤčžžŞă

### èĝčăEşæŮzæąĹ

âĖŻăŸĀăŸĹæŬčăŖŮăŸĀăŸĹæŮĜäzúâĤĤăŸžăŖĆæŤŕçŻĐæL'ŕásŤăŤăĜ;æŤŕijŇăĕCăŸŇèĤZăăŭijŻ

```
static PyObject *py_get_filename(PyObject *self, PyObject *args) {
    PyObject *bytes;
    char *filename;
    Py_ssize_t len;
    if (!PyArg_ParseTuple(args, "O&", PyUnicode_FSConverter, &bytes)) {
        return NULL;
    }
    PyBytes_AsStringAndSize(bytes, &filename, &len);
    /* Use filename */
    ...

    /* Cleanup and return */
    Py_DECREF(bytes);
    Py_RETURN_NONE;
}
```

âĕĆăđIJă;ăăŭşçžŖæIJĹăžĖăŸĀăŸĹ PyObject \* ĩijŇăŸŇăĕIJŽăŕĖăĤŭĕ;ŇăĤăĹŖăŸĀăŸĹæŮĜäzúâĤĤĤĤ

```
PyObject *obj;      /* Object with the filename */
PyObject *bytes;
char *filename;
Py_ssize_t len;

bytes = PyUnicode_EncodeFSDefault(obj);
PyBytes_AsStringAndSize(bytes, &filename, &len);
/* Use filename */
...

/* Cleanup */
Py_DECREF(bytes);
```

If you need to **return** a filename back to Python, use the following **code**:

```
/* Turn a filename into a Python object */

char *filename;      /* Already set */
int filename_len;    /* Already set */
```

```
PyObject *obj = PyUnicode_DecodeFSDefaultAndSize(filename, filename_
↪len);
```

## èõìèõž

äzēāRŕçğžæd'■æŰzâijRæIēād'DçRĒæŰĠzūāR■æŸfäyÄäyĭā;ĹæčŸæL'NçŽĐēŰōécŸrijNæIJĀāRŌāžd'āēČædIJā;āāIJæL'āſŦāžčçāAäy■ā;ſçŦĪæIJnēLČçŽĐæLĀæIJfijNæŰĠzūāR■çŽĐād'DçRĒæŰzâijRāŠNāŠāNĒæNñçijŰçāA/çŦNēIcā■ŰēLČrijNād'DçRĒāiRā■ŰçñēijNāžčçRĒē;ñæ■cāŠNāEūāzŰād'■æiCæČĒāEĭāAO

## 17.18 15.18 äijäéĀŠaũsæL'ŠâijĀçŽĐæŰĠzūçžŽCæL'āſŦ

### ēŰōécŸ

ä;āāIJĪPythonäy■æIJL'äyÄäyĭæL'ŠâijĀçŽĐæŰĠzūāržèsajijNä;EæŸféIJĀèeAārEāōČäijäçžŽēeAä;ſçŦĪ

## èğçāEşæŰzæqĹ

èeAārEäyÄäyĭæŰĠzūē;ñæ■cäyžäyÄäyĭæŦ'ādNçŽĐæŰĠzūāRŔèſçñēijNä;ſçŦĪ  
PyFile\_FromFd() ĩijNæČäyNĭijŽ

```
PyObject *fobj; /* File object (already obtained somehow) */
int fd = PyObject_AsFileDescriptor(fobj);
if (fd < 0) {
    return NULL;
}
```

çžŠædIJæŰĠzūāRŔèſçñēæŸféĀŽèſĠērČçŦĪ fobj äy■çŽĐ fileno()
æŰzæşŦēŌūā;ŰçŽĐāĀČ āZāæ■d'rijNāzzā;ŦāžèèſŽçğ■æŰzâijRæŽt'ēIJşçžŽäyÄäyĭæRŔèſçŦāŽĪçŽĐāržèsæČ
äyÄæŰēā;āæIJL'āžEèſŽäyĭæRŔèſçŦāŽĪrijNāōČārseČ;ècnäijäéĀŠçžŽād'Žäyĭā;ŌçžğçŽĐāRŕād'DçRĒæŰĠzū

āēČædIJā;äeIJĀèeAē;ñæ■cäyÄäyĭæŦ'ādNæŰĠzūāRŔèſçñēäyžäyÄäyĭPythonāržèsajijNēĀČçŦĪäyNē  
PyFile\_FromFd() :

```
int fd; /* Existing file descriptor (already open) */
PyObject *fobj = PyFile_FromFd(fd, "filename", "r", -1, NULL, NULL, NULL,
↪1);
```

PyFile\_FromFd() çŽĐāRČæŦŕāržāžŦāEĒç;ōçŽĐ open() āĠ;æŦŕāĀČ NUL-  
Lēāſçd'žçijŰçāAāĀēŦŽèſŦāŠNæ■cēāNāRČæŦŕā;ſçŦĪéžŸēōd'āĀijāĀČ

## èõìèõž

āēČædIJārEPythonäy■çŽĐæŰĠzūāržèsajijäçžŽCrijNæIJL'äyÄäzŽæşĭæĐRāžNéāžāĀČ  
ēēŰāĒĪrijNPythonēĀŽèſĠ io æĭāāĪŰæL'ğēāNēĠāũşçŽĐĪ/OçijŞāEşāĀČ

āIJlāijāēĀšāzā;TçšzādNçŽDæŮGāzūæŔŔèĤŕçñçzŽCāzNāL'■īijNā;āēČ;èēAēēŮāĒLāIJlçŽyāžTæŮGāzūārf  
āy■çŽDūçŽDèŕīijNā;āāijZæL'ŠāzšæŮGāzūçšzçžšāyLéIççŽDæTŕæ■ōāĀĆ

āĒŮāēñāīijNā;āēIJĀèēAçL'zāLŋæšlāēDRæŮGāzūçŽDā;ŠāsđēĀĒzēāŔLāĒšēŮ■æŮGāzūçŽDèAŇet'čāĀC  
āēČædIJāyĀāylæŮGāzūæŔŔèĤŕçñèèēñāijāçzŽCīijNā;EæŸŕāIJlPythonāy■ēĤŸāIJlèēñā;ĤçTlçĪāīijNā;āēIJĀèē  
çšzāijijçŽDīijNāēČædIJāyĀāylæŮGāzūæŔŔèĤŕçñèèēñē;ñæ■cāyžāyĀāylPythonæŮGāzūārfzēsāīijNā;āēIJĀèē  
PyFile\_FromFd() çŽDæIJĀāŔŌāyĀāylāŔCæTŕècēēōç;ōæLŔlīijNçTlāēlæŇGāGžPythonāžTèŕēāĒšēŮ

āēČædIJā;āēIJĀèēAžŌCæāGāGEI/OāžŠāy■ā;ĤçTlāēCāĀĀfdopen()  
āG;æTŕælēāLZāžzāy■āŔNçšzādNçŽDæŮGāzūārfzēsāæŕTāēC FILE \* ārfzēsāīijN  
ā;āēIJĀèēAçL'zāLŋāŕŔāēČāžEāĀČēĤZæāūāAžāijZāIJl/OāāEæāLāy■āžgçTšāyd'āylāōNāĒlāy■āŔNçŽDl/Oç  
īijLāyĀāylæŸŕælēēGŀPythonçŽD io ælāālŮīijNāŔēāyĀāylælēēGŀCçŽD studio  
īijLāĀĆ āČŔCāy■çŽD fclose() āijZāĒšēŮ■PythonēēAā;ĤçTlçŽDæŮGāzūāĀĆ  
āēČædIJēōl'ā;āēĀL'çŽDèŕīijNā;āāžTèŕēāijZēĀL'æNl'āŌzædDāžzāyĀāylæLl'āsTāžççāAælēād'ĐçŔEāžTāsČ  
ēĀNāy■æŸŕā;ĤçTlāēlēēGŀ<stdio.h>çŽDēñŸāsCæL;ēsāāLšēČ;āĀĆ

## 17.19 15.19 āžŌCèr■ēlĀāy■èŕzāŔŮçšzæŮGāzūārfzēsā

### éŮōēčŸ

ā;āēēAāEŽCæLl'āsTālēēŕzāŔŮælēēGŀāžzā;TPythonçšzæŮGāzūārfzēsāy■çŽDæTŕæ■ōīijLæŕTāēČæZŌC

### ègçĀEšæŮzæāĻ

èēAēŕzāŔŮāyĀāylçšzæŮGāzūārfzēsāçŽDæTŕæ■ōīijNā;āēIJĀèēAēG■ād'■èŕČçTl  
read() æŮzæšTīijNçDūāŔŌæ■ççāōçŽDègççāAēŌūāçŮçŽDæTŕæ■ōāĀĆ

āyNēlCæŸŕāyĀāylCæLl'āsTāG;æTŕāçNā■ŔīijNāzĒāzĒāŔlæŸŕēŕzāŔŮāyĀāylçšzæŮGāzūārfzēsāy■çŽD

```
#define CHUNK_SIZE 8192

/* Consume a "file-like" object and write bytes to stdout */
static PyObject *py_consume_file(PyObject *self, PyObject *args) {
    PyObject *obj;
    PyObject *read_meth;
    PyObject *result = NULL;
    PyObject *read_args;

    if (!PyArg_ParseTuple(args, "O", &obj)) {
        return NULL;
    }

    /* Get the read method of the passed object */
    if ((read_meth = PyObject_GetAttrString(obj, "read")) == NULL) {
        return NULL;
    }

    /* Build the argument list to read() */
    read_args = Py_BuildValue("(i)", CHUNK_SIZE);
```

```

while (1) {
    PyObject *data;
    PyObject *enc_data;
    char *buf;
    Py_ssize_t len;

    /* Call read() */
    if ((data = PyObject_Call(read_meth, read_args, NULL)) == NULL)
→{
        goto final;
    }

    /* Check for EOF */
    if (PySequence_Length(data) == 0) {
        Py_DECREF(data);
        break;
    }

    /* Encode Unicode as Bytes for C */
    if ((enc_data=PyUnicode_AsEncodedString(data, "utf-8", "strict
→")) ==NULL) {
        Py_DECREF(data);
        goto final;
    }

    /* Extract underlying buffer data */
    PyBytes_AsStringAndSize(enc_data, &buf, &len);

    /* Write to stdout (replace with something more useful) */
    write(1, buf, len);

    /* Cleanup */
    Py_DECREF(enc_data);
    Py_DECREF(data);
}
result = Py_BuildValue("");

final:
    /* Cleanup */
    Py_DECREF(read_meth);
    Py_DECREF(read_args);
    return result;
}

```

èçAætNèrTefZäyläzççäAijNäĒŁæđDéĀäyĀäylçszæŨĠäzũáržèsærfTæĆäyĀäyĬStringIOăőđäĬNĭijŇç

```

>>> import io
>>> f = io.StringIO('Hello\nWorld\n')
>>> import sample
>>> sample.consume_file(f)

```

```
Hello
World
>>>
```

## èõléõž

åŠŇæŽóéĀŽçşçzşæŮĠäzûäy■āŖŇçŽĎæŸřijŇäyĀäylçşzæŮĠäzûärzèşqāzûäy■éIJĀèèAä;ŁçŦlā;Ŏçžğ  
āŽāæ■d'rijŇä;āäy■ēĈ;ā;ŁçŦlāæŽóéĀŽçŽĎCāzŞāĠ;æŦŕæİēèøŁéŮōāōĈāĀĈ  
ā;āéIJĀèèAä;ŁçŦlPythonçŽĎC APIæİēāĈŖæŽóéĀŽæŮĠäzûçşzāijijçŽĎéĈcæāūæŞ■ā;IJçşzæŮĠäzûärzèşqāĀ

āIJlæĹSāznçŽĎèğçāEşæŮzæāLäy■rijŇread() æŮzæşŦāzŎècnāijāéĀŞçŽĎärzèşqāy■æŖŖāŖŮāĠzæİē  
äyĀäylāŖĈæŦŕāLŮēālēcnæĎĎāzžçĎūāŖŎäy■æŮ■çŽĎècnāijāçžž PyObject\_Call()  
æİēēŖĈçŦlèçŽäylæŮzæşŦāĀĈ èèAæçĀæşèæŮĠäzûæIJnār;rijŮEOFrijL'rijŇä;ŁçŦlāžE  
PySequence\_Length() æİèæşèçIJŇæŸŖāŖçèçŦāŽĎärzèşqéŦŖāžæyž0.

āržāžŎæL'ĀæIJL'çŽĎI/OæŞ■ā;IJrijŇä;āéIJĀèèAāEşæşlāžŦāşĈçŽĎçijŮçāAæāijāijŖrijŇèçŸæIJL'ā■ŮèŁ  
æIJnèŁĈæijŦçd'žāžEāçĈā;ŦāžææŮĠæIJnāīāijŖēržāŖŮäyĀäylæŮĠäzûāzûārEçzŞæĎIJæŮĠæIJnèğççāAäyž  
āçĈæĎIJā;āæĈşāžèāžŇèçŽāĹūēāīāijŖēržāŖŮæŮĠäzûrijŇāŖİéIJĀèèAāēōæŦžāyĀçĈççĈā■şāŖrijŇä;ŇāçĈ

```
...
/* Call read() */
if ((data = PyObject_Call(read_meth, read_args, NULL)) == NULL) {
    goto final;
}

/* Check for EOF */
if (PySequence_Length(data) == 0) {
    Py_DECREF(data);
    break;
}
if (!PyBytes_Check(data)) {
    Py_DECREF(data);
    PyErr_SetString(PyExc_IOError, "File must be in binary mode");
    goto final;
}

/* Extract underlying buffer data */
PyBytes_AsStringAndSize(data, &buf, &len);
...
```

æIJnèŁĈæIJĀéŽ;çŽĎāIJŖæŮzāIJlāžŎāçĈā;ŦèçŽèāŇæ■ççāōçŽĎāEĒā■ŸçōaçŖEāĀĈ  
ā;Şād'ĎçŖE PyObject \* `` āŖŸéĠŖçŽĎæŮūāĀŽīijŇéIJĀèèAæşlæĎŖçōaçŖEāijŦçŦlèōæŦ  
ārž ``Py\_DECREF() çŽĎērĈçŦlārşæŸŖæİēāAŽèçŽäylçŽĎāĀĈ

æIJnèŁĈāzççāAāžèäyĀçğ■éĀŽçŦlæŮzāijŖçijŮāEŽrijŇāŽāæ■d'āzŮāzşèĈ;éĀĈçŦlāžŎāEūāzŮçŽĎæŮĈ  
ā;ŇāçĈrijŇèèAāEŽæŦŕæ■ōrijŇāŖİéIJĀèèAèŎūāŖŮçşzæŮĠäzûärzèşqçŽĎ write()  
æŮzæşŦrijŇārEæŦŕæ■ōè;ñæ■cāyžāŖĹéĀĈçŽĎPythonāržèşq rijLā■ŮèŁĈæĹŮUni-  
coderijL'rijŇçĎūāŖŎērĈçŦlèŖææŮzæşŦārEç;ŞāEēāEŽāĒēāĹŖæŮĠäzûāĀĈ

æIJĀāŖŎrijŇār;çōaçşzæŮĠäzûärzèşqéĀŽäyÿèçŸæŖŖā;ŽāEūāzŮæŮzæşŦrijLærŦāçCreadline(),

read\_info()iijL'iiijŃ æĹŠäzñæIJĀāē;āRĥā;ĤçTĭāšžæIJñçŽD read() āŠŃ write()  
æŰzæşTāĀĆ āIJĭāEŻCæL'ĭāsTçŽDæŰūāĀŽiijŃēČ;çōĀā■Tārsār;éGRçōĀā■TāĀĆ

## 17.20 15.20 ad'DçRĖCèr■ēĬÄäy■çŽDāRrè£■äzčāržèśą

### éŰóécŸ

ä;äæČşāEŻCæL'ĭāsTäzčçāAāad'DçRĖæĭēēGlāzzä;TāRrè£■äzčāržèśąāēČāĹŰēāĭāĀAāĖČçzDāĀAæŰĜä

### èğčāEşæŰzæąĹ

äyŃēĭcæŸřäyĀäyĭCæL'ĭāsTāĜ;æTřä;Ńā■ŘiijŃæijTçd'žāžEæĀŌæūūad'DçRĖāRrè£■äzčāržèśąäy■çŽDā

```
static PyObject *py_consume_iterable(PyObject *self, PyObject_  
↪*args) {  
    PyObject *obj;  
    PyObject *iter;  
    PyObject *item;  
  
    if (!PyArg_ParseTuple(args, "O", &obj)) {  
        return NULL;  
    }  
    if ((iter = PyObject_GetIter(obj)) == NULL) {  
        return NULL;  
    }  
    while ((item = PyIter_Next(iter)) != NULL) {  
        /* Use item */  
        ...  
        Py_DECREF(item);  
    }  
  
    Py_DECREF(iter);  
    return Py_BuildValue("");  
}
```

### èőĭèőž

æIJñēĹČäy■çŽDäzčçāAāŠŃPythonäy■āržāžTāzčçāAçşzäiijāĀĆ  
PyObject\_GetIter() çŽDèrČçTĭāŠŃèrČçTĭ iter()  
äyĀæūāRrèŌūāĹŰäyĀäyĭē£■äzčāŽĭāĀĆ PyIter\_Next() āĜ;æTřèrČçTĭ next  
æŰzæşTē£TāŽdäyŃäyĀäyĭāĖČçt'āæĹŰNULL(āēČædIJæşæIJĹ'āĖČçt'āāžE)āĀĆ  
ēēAæşĭæĎRæ■ççāōçŽDāEĖā■ŸçōaçRĖāĀTāĀT Py\_DECREF()  
ēĬJāēēAāRŃæŰūāIJĭäžğçTšçŽDāĖČçt'āāŠŃē£■äzčāŽĭāržèśąæIJñēžnäyĹāRŃæŰūēčñèrČçTĭiijŃ  
äzēēAĤāĖ■āĜžçŌřāEĖā■ŸæşĎēIJšāĀĆ

## 17.21 15.21 èrlæÚ■áLEæóéTŽèrr

### éUóécY

ègçéGLáZÍlāZāyžæšŘäyÍáLEæóéTŽèrrāĀæĀzçžféTŽèrrāĀèóféUóèúLçTŇæLŪāĒūāzŪèGt'ás;éT  
ä;āæČšèŌūā;ŪPythonāĀEæāLāfæAřijŇāzŌèĀŇæL;āGžāIJlāRŚçTšéTŽèrrçŽDæŪūāĀZā;āçŽDçlŇāzRèf

### ègçāEşæŪzæqL

faulthandler æÍāāIŪèČ;èçñçTlæİēāyōä;æègçāEşæfZāyİéUóécYāĀČ  
āIJlā;āçŽDçlŇāzRāy■āijTāĒēāyŇāLŪāzççāAřijŽ

```
import faulthandler
faulthandler.enable()
```

āRēād'ŪèfYāRfāzēāČRāyŇéİçèfZæūūā;fçTl -Xfaulthandler  
æİēèfRēāŇPythonijŽ

```
bash % python3 -Xfaulthandler program.py
```

æIJāāRŌijŇā;āāRfāzēèō;ō PYTHONFAULTHANDLER çŌfāçČāRŸéGRāĀČ āijĀāRf-  
faulthandlerāRŌijŇāIJlCæL'āsTāy■çŽDèGt'ás;éTŽèrrāijZāřijèGt'āyĀāyIŸPythonéTŽèrrāāEæāLèçñæL'Sā■ř

```
Fatal Python error: Segmentation fault

Current thread 0x00007fff71106cc0:
  File "example.py", line 6 in foo
  File "example.py", line 10 in bar
  File "example.py", line 14 in spam
  File "example.py", line 19 in <module>
Segmentation fault
```

ār;çōæfZāyİāzūāy■èČ;āSĻèrL'ā;āCāzççāAāy■āSĻéGŇāGžéTŽāžEřijŇā;EæYřèGšārSèČ;āSĻèrL'ā;āPytl

### èóİèőž

faulthandlerāijZāIJlPythonāzççāAæL'gēāŇāGžéTŽçŽDæŪūāĀZāRŚā;āāsTçd'žèùšèyİāfæAřāĀČ  
èGšārSřijŇāōČāijZāSĻèfL'ā;āāGžéTŽæŪūèçñèrČçTlçŽDæIJāéāūçžgæL'āsTāG;æTřæYřāSĻāyİāĀČ  
āIJlpdbāSŇāĒūāzŪPythonèrČèrTāZlçŽDāyōāL'āyŇijŇā;āārSèČ;èf;æāzæžfæžRæL;āLřéTŽèrræL'ĀāIJlçŽD

faulthandlerāy■āijZāSĻèrL'ā;āāzžā;TÇèr■ēlĀāy■çŽDèTŽèrrāfæAřāĀČ  
āZāæ■d'ijŇā;æĪĀèēAā;fçTlāijāçzšçŽDÇèrČèrTāZlřijŇærTāeČgdbāĀČ  
āy■èfGřijŇāIJlfaulthandlerèf;èyİāfæAřāRfāzēèŌ'ā;āāŌzāLd'æŪ■āzŌāSĻéGŇçIāæL'ŇāĀČ  
èfYèēAæšlæDŘçŽDæYřāIJlCāy■æšRāžZçszādŇçŽDèTŽèrrāRřèČ;āy■ād'İāōzæYŠæAçād'■āĀČ  
ā;ŇāeČřijŇāeČædIJāyĀāyIæL'āsTāyçāijCāžEçlŇāzRāāEæāLāfæAřijŇāōČāijZèŌ'faulthandlerāy■āRřçT  
éČcāzLā;āāzšā;Ūāy■āLřāzžā;Tç;SāGžřijLéZd'āžEçlŇāzRāēTæžČād'ŪřijL'āĀČ

# 18 éŽĎāṭA

## 18.1 ālJlćžĚtĎæžŘ

<http://docs.python.org>

āĉĆæđIJā;āēIJĀēĉAæŭsāĚēžĚēġĉæŌćl' ŭēr■ēlĀāŠNæĪāāĪŮĉŽĎĉzĚēĹĆiijNéĆčāzĹäy■āĚĚērt' iijNPyth  
3 ĉŽĎæŮĜæāĉēĀNāy■æYřāžēāL■ĉŽĎēĀAçL'ĹæIJñ

<http://www.python.org/dev/peps>

āĉĆæđIJā;āāŘŚĉŘĚēġĉāyžpythonēr■ēlĀæŭzāĹāæŮřĉL'žæĀġĉŽĎāĹāIJzāžēāŘĹāōđĉŌřĉŽĎĉzĚēĹĆiijN  
Enhancement ProposalsāĀT-PythonāijĀāŘŚĉijŮĉāAēġĎēNĉiijL'ĉzĪāřzæYřēĪdāyŷāōĪērt'ġĉŽĎēġĎæžŘāĀĆārd'

<http://pyvideo.org>

ēĚŽēĜNæIJL'æĪēēĜĹæIJĀēĚŚĉŽĎPyConād'ġāijŽāĀAçŤĹæĹŮĉzĎēġAēĪĉāijŽĉ■L'ĉŽĎād'ġēĜŘēġĚēĉŚāē  
3āy■æŭzāĹāĉŽĎĉŽĎæŮřĉL'žæĀġāĀĆ

<http://code.activestate.com/recipes/langs/python>

ēŤĹæIJšāžēāēĪēiijNActiveStateĉŽĎPythonĉL'ĹāĪŮāŭšĉzŘæĹŘāyžāyĀāyĹæL'ġāĹŤæŤřāžēā■ĈēōāĉŽĎēŚĹ

<http://stackoverflow.com/questions/tagged/python>

Stack Overflow ĉŽōāL'■æIJL'ēŭĚēĚĜ175,000āyĪēŮōēĉYēĉnæāĜēōřāyžPythonĉŽyāĚšīijĹēĀNāĚŭāy■ād'  
3ĉŽĎiijL'āĀĆāřġĉōāēŮōēĉYāŠNāŽđĉ■ŤĉŽĎērt'ĪēĜŘāy■āŘNīijNā;ĚæYřāž■ĉĎŭēĈ;āŘŚĉŌřāĹāđ'Žāē;āijYĉġ

## 18.2 Pythonā■ēāžāāžēĉs■

āyNēĪĉēĚŽāžŽāžēĉs■æŘŘā;ŽāžĚāřzPythonĉijŮĉĪNĉŽĎāĚēēŮĪāžNĉz■iijNāyŤēĜ■ĈzæŤ;āIJāžĚPytho  
3āyĹāĀĆ

Beginning Python: From Novice to Professional, 2nd Edition, by Magnus Lie HetâĀŘ  
land, Apress (2008). Programming in Python 3, 2nd Edition, by Mark Summerfield, Addison-  
Wesley (2010).

- *Learning Python*iijNĉññāŽŽĉL'Ĺ iijNā;IJēĀĚ Mark LutzīijN ŌāĀŽReilly & Associates  
āĜžĉL'Ĺ (2009)āĀĆ
- *The Quick Python Book*iijNā;IJēĀĚ Vernon CederīijN Manning āĜžĉL'Ĺ(2010)āĀĆ
- *Python Programming for the Absolute Beginner*iijNĉññāyĹĉL'ĹiijNā;IJēĀĚ Michael  
DawsonīijNCourse Technology PTR āĜžĉL'Ĺ(2010).
- *Beginning Python: From Novice to Professional*iijNĉññāžNĉL'ĹiijN ā;IJēĀĚ Magnus  
Lie HetâĀŘ landīijN Apress āĜžĉL'Ĺ(2008).
- *Programming in Python 3*iijNĉññāžNĉL'ĹiijNā;IJēĀĚ Mark SummerfieldīijNAddison-  
Wesley āĜžĉL'Ĺ (2010).



## 18.3 éñŸçžğäzëçs■

äÿÑéİççŽĐēfŽăžŽăzëçs■æRRă;ŽăžEæŽt'âd'ŽénŸçžğçŽĐēŇČăŽt'rijŇăžšăŇĚăŔŋPython  
3æŮzélççŽĐăĚĚăőžăĂĆ

- *Programming Python*ijŇçňňăŽŽçL'Ĺ, by Mark Lutz, OăĂŽReilly & Associates  
ăĜžçL'Ĺ(2010).
- *Python Essential Reference*ijŇçňňăŽŽçL'ĹijŇă;IJèĂĚ David Beazley, Addison-Wesley  
ăĜžçL'Ĺ(2009).
- *Core Python Applications Programming*ijŇçňňăÿL'çL'ĹijŇă;IJèĂĚ Wesley Chun,  
Prentice Hall äĜžçL'Ĺ(2012).
- *The Python Standard Library by Example* ijŇ ä;IJèĂĚ Doug HellmannijŇAddison-  
Wesley äĜžçL'Ĺ(2011).
- *Python 3 Object Oriented Programming*ijŇă;IJèĂĚ Dusty Phillips, Packt Publishing  
ăĜžçL'Ĺ(2010).
- *Porting to Python 3*ijŇ ä;IJèĂĚ Lennart RegebroijŇCreateSpace äĜžçL'Ĺ(2011), <http://python3porting.com>.

## 19 äĚşăžŎërSèĂĚ

äĚşăžŎërSèĂĚ

- äĝŞăŔ■ijŽ çĒĹèČ;
- ä;őăĚăijŽ yidao620
- EmailijŽ yidao620@gmail.com
- ä■ŽăőćijŽ <http://yidao620c.github.io/>
- GitHubijŽ <https://github.com/yidao620c>

## 20 Roadmap

2014/08/10 - 2014/08/31:

	githubéázççŽőæŔ■ăžžii jŇreadthedocsæŮĜæăççŤSæĹŔăĂĆ
	æŤt' äÿĹéázççŽőçŽĐăăĚăđúăőŇăĹŔ

2014/09/01 - 2014/10/31:

	ăĹ' ■4çŋăççfzèŕSăőŇăĹŔ
--	------------------------

2014/11/01 - 2015/01/31:

	åL'■8çnáçfzèrSåõÑæLŘ
2015/02/01 - 2015/03/31:	
	åL'■9çnáçfzèrSåõÑæLŘ
2015/04/01 - 2015/05/31:	
	10çnáçfzèrSåõÑæLŘ
2015/06/01 - 2015/06/30:	
	11çnáçfzèrSåõÑæLŘ
2015/07/01 - 2015/07/31:	
	12çnáçfzèrSåõÑæLŘ
2015/08/01 - 2015/08/31:	
	13çnáçfzèrSåõÑæLŘ
2015/09/01 - 2015/11/30:	
	14çnáçfzèrSåõÑæLŘ
2015/12/01 - 2015/12/20:	
	15çnáçfzèrSåõÑæLŘ
2015/12/21 - 2015/12/31:	
	årzáĚléČlçfzèrSèfZèaÑæäaårzáÿĂæñą
2016/01/01 - 2016/01/10:	
	<div> <div>årzáđ'ŮåĚñăi jĂăRŚăÿČăõÑæTt'çL'Í1.</div> <div>↪0ïi jŇăŇĚæŇñè; ñæ■căŘŎçŽĎPDFæŮĞăžů</div> </div>