免費電子書

學習

Python Language

Free unaffiliated eBook created from
Stack Overflow contributors.

#python

1
1: Python 2
2
2
Python 3.x 2
Python 2.x 3
Examples 3
3
Python 3
IDLEPython 4
Hello World Python 4
Python shell 5
5
6
6
6
8
IDLE - Python GUI 9
9
9
9
9
10
10
10
11
12
12
12
13

13
14
15
15
18
19
- strrepr 20
20
STR 20
pip 21
/ 21
21
21
Python 2.7.x3.x. 22
2: * args kwargs 24**
24
h11 24
h12 24
h13 24
Examples 24
* args 24
** kwargs 25
* args 25
** kwargs 26
* args 26
27
kwarg 27
** kwargs 27
3: __name__ 28
28
28

Examples 28
__name__ =='__ main__' 28
1 28
2 28
function_class_or_module .__ name__ 28
..... 30
4: 2to3 31
..... 31
..... 31
..... 31
Examples 31
..... 31
Unix 32
..... 32
Unix 32
..... 32
5: ArcPy 34
..... 34
Examples 34
“” 34
createDissolvedGDBgdb 34
6: Asyncio 35
Examples 35
..... 35
..... 36
UVLoop 36
..... 37
..... 37
..... 37
Websocket 38
asyncio 38

7: base64 39
..... 39
..... 39
..... 39
..... 40
Examples 40
Base64 40
Base32 41
Base16 42
ASCII85 42
Base85 43
8: ChemPy - python 45
..... 45
Examples 45
..... 45
..... 45
..... 45
..... 45
..... 46
..... 46
9: configparser 48
..... 48
..... 48
..... 48
Examples 48
..... 48
..... 48
10: ctypes 50
..... 50
Examples 50
..... 50
..... 50

50
51
ctypes 51
ctypes 51
ctypes 52
52
11: Deque 54
54
54
54
Examples 54
54
deque 54
deque 54
55
12: dis 57
Examples 57
dis 57
Python 57
57
13: Django 59
59
Examples 59
DjangoHello World 59
14: Functools 61
Examples 61
61
total_ordering 61
61
lru_cache 62
cmp_to_key 62

15: hashlib 63
..... 63
Examples 63
MD5 63
OpenSSL 63
16: Heapq 65
Examples 65
..... 65
..... 65
17: HTML 66
Examples 66
BeautifulSoup 66
BeautifulSoupCSS 66
PyQuery 67
18: ijson 68
..... 68
Examples 68
..... 68
19: IterablesIterators 69
Examples 69
Iterator vs Iterable vs Generator 69
..... 71
..... 71
iterable 71
..... 71
..... 72
20: Itertools 73
..... 73
Examples 73
..... 73
..... 73
itertools.product 74

itertools.count 74
itertools.takewhile 75
itertools.dropwhile 75
..... 76
Itertools 76
..... 77
itertools.repeat 77
..... 77
..... 78
itertools.permutations 78
21: JSON 79
..... 79
..... **79**
..... 79
..... 79
..... 79
..... 79
..... 79
..... 79
..... 79
..... 80
Examples 80
Python dictJSON 80
JSONPython dict 80
..... 81
..... 81
`load` vs`load``dump` vs`stacks` 81
`json.tool`JSON 82
JSON 82
..... **82**
..... **82**
..... **82**
JSON 83

22: kivy - NUIPython 84
..... 84
Examples 84
..... 84
23: Neo4jCypherPy2Neo 87
Examples 87
..... 87
Neo4j 87
Neo4j 87
1 87
2 88
Cypher 88
24: os.path 89
..... 89
..... 89
Examples 89
..... 89
..... 89
..... 89
..... 90
..... 90
..... 90
25: os 91
..... 91
..... 91
..... 91
Examples 91
..... 91
..... 91
..... 91
..... 91
POSIX 91

92
makedirs - 92
26: Pandas Transform 93
Examples 93
93
93
pandas transform 93
94
transform 94
27: Pickle 95
95
95
95
Pickleable 95
pickle 95
Examples 95
Pickle 95
95
96
picklebyte 96
96
28: pipPyPI 98
98
98
98
Examples 98
98
98
99
`pip` 99
99

Linux 99
Windows 99
requirements.txt 100
virtualenvrequirements.txt 100
Pythonpip 100
101
101
101
29: PostgreSQL 103
Examples 103
103
pip 103
103
30: py.test 104
Examples 104
py.test 104
104
104
104
104
104
py.test 105
105
31: pyaudio 107
107
107
Examples 107
I / O 107
I / O 108
32: pyautogui 110
110

Examples 110
110
110
ScreenShot 110
33: pygame 111
111
111
111
Examples 111
pygame 111
Pygame 111
111
112
112
34: Pyglet 113
113
Examples 113
PygletHello World 113
Pyglet 113
Pyglet 113
PygletOpenGL 113
PygletOpenGL 114
35: PyInstaller - Python 115
115
115
Examples 115
115
Pyinstaller 115
116
116
116

116

36: Python HTTP Server **117**

Examples 117

HTTP 117

..... 117

SimpleHTTPServerAPI 118

BaseHTTPRequestHandlerGETPOSTPUT 119

37: Python Lex-Yacc **121**

..... 121

..... 121

Examples 121

PLY 121

“HelloWorld” PLY - 121

1Lex 124

..... **125**

h21 128

h22 129

h23 131

h24 132

h25 132

h26 134

h27 135

h28 136

h29 136

h210 137

2Yacc 138

..... **139**

h211 142

38: Python **144**

..... 144

Examples 144

.....

144
144
145
39: PythonMutable vs ImmutableHashable **146**
Examples 146
146
Immutables **146**
147
Mutables **147**
148
148
149
40: Python **150**
150
Examples 150
Lambda 150
150
150
150
41: PythonShell **151**
151
Examples 151
ssh 151
42: python **152**
Examples 152
152
152
43: Pythonpyserial **153**
153
153
153

Examples 153
..... 153
..... 153
..... 153
44: Python 155
Examples 155
..... 155
..... 155
..... 156
45: PythonExcel 157
Examples 157
Excel 157
OpenPyXL 157
xlsxwriterexcel 158
xlrdexcel 160
xlsxwriterExcel 161
46: Python 162
..... 162
..... 162
Examples 162
Python 162
..... 163
47: Python 164
..... 164
Examples 164
..... 164
..... 164
..... 164
..... 164
..... 164
..... 164
48: Python 166

166
Examples 166
SSE 166
Asyncio SSE 166
49: PythonCurses 167
167
Examples 167
167
wrapper 167
50: Python 169
Examples 169
169
169
Deque 170
170
..... 171
51: Python 172
172
Examples 172
Python - 172
Http 172
TCP 173
UDP 174
Simple HttpServer 174
52: Python - virtualenv 176
176
Examples 176
176
176
Virtualenv 176
virtualenv 176
53: Python 178

178
Examples 178
178
179
179
179
180
180
54: setup.py 182
182
182
Examples 182
setup.py 182
python 183
setup.py 183
184
55: Sqlite3 185
Examples 185
Sqlite3 - 185
185
56: SYS 187
187
187
187
Examples 187
187
187
187
187
57: tempfile NamedTemporaryFile 189
189
Examples 189

189
58: Tkinter 190
190
190
Examples 190
tkinter 190
191
191
191
191
59: Unicode 193
193
193
Examples 193
193
Unicode 193
unicode 194
/ 194
194
194
194
I / O 195
60: WebWSGI 196
196
Examples 196
196
61: Web 197
197
197
197
197

Examples 198
URL 198
URL 199
62: “with”Statement 200
200
200
200
Examples 200
with 200
200
201
201
202
203
63: Python 2Python 3 204
204
204
Examples 204
204
Unicode 206
207
Reduce 209
xrange 209
211
Iterables 211
214
.next 215
216
216
217
execPython 3 217
Python 2hasattr 218

219

219

Python 3“”220

<>``=repr220

/221

Python 3cmp222

222

222

filtermapzip225

/225

226

I / O226

round227

227

round227

228

228

228

229

64: intfloatstrtuplefrozensets230

Examples230

230

230

Frozenset230

65: 231

231

Examples231

231

ParentChildren231

C232

PyPar232

66: 233
233
Examples 233
233
233
233
67: 235
235
Examples 235
235
68: “pip”PyPI 236
236
236
Examples 236
236
ImportError 237
237
69: AMQPStormRabbitMQ 239
239
Examples 239
RabbitMQ 239
RabbitMQ 239
RabbitMQ 240
70: Pandas 242
242
Examples 242
Pandas 242
71: PythonWindows 244
244
Examples 244
Python 244

Flask Web 245
72: PythonRaspberry PI 247
Examples 247
- 247
73: PythonWeb 250
..... 250
..... 250
WebPython 250
..... 250
requests 250
requests-cache 250
scrapy 250
selenium 250
HTML 250
BeautifulSoup 250
lxml 250
Examples 250
lxml 250
Web 251
Scrapy 251
Scrapy 251
BeautifulSoup4 252
Selenium WebDriver 252
urllib.requestWeb 252
..... 253
74: Python 254
Examples 254
Matplotlib 254
Seaborn 255
Mayavi 258
Plotly 259
75: ZIP 261

261
261
Examples 261
Zip 261
Zipfile 261
zip 262
262
76: 263
Examples 263
263
77: 264
264
264
Examples 264
264
264
265
265
266
266
268
268
269
269
270
270
270
270
78: 272
272
272
272

Examples 272
..... 272
..... 272
..... 273
..... 273
..... 274
..... 274
..... 275
..... 275
..... 275
..... 275
..... 276
..... 276
..... 276
79: 277
..... 277
..... 277
Examples 277
..... 277
..... 278
..... 278
..... 278
Python 23six 279
..... 279
..... 279
..... 279
..... 279
..... 280
..... 280
80: 282
..... 282

Examples 282
PyTesseract 282
PyOCR 282
81: CLI 284
..... 284
..... 284
Examples 284
..... 284
argparse 285
argparse 286
82: 289
..... 289
Examples 289
math.sqrtcmath.sqrt 289
**pow 289
math.pow 290
math.expcmath.exp 291
1math.expm1 291
cmath 292
3pow 294
nth-root 295
..... 295
83: 296
Examples 296
py2app 296
cx_Freeze 296
84: 298
..... 298
..... 298
Examples 298
“” 298
..... 298

298
299
85: 300
300
300
300
Examples 300
300
301
301
301
301
302
303
304
304
305
306
307
307
86: 308
308
308
308
Examples 308
308
309
309
- 310
310
311

87: **312**
Examples 312
..... 312
..... 312
..... 312
..... 312
..... 313
..... 313
..... 313
..... 314
..... 314
88: **316**
Examples 316
IPython%% timeittimeit 316
timeit 316
timeit 316
line_profiler 316
cProfileProfiler 317
89: Python **318**
..... 318
Examples 318
..... 318
PyPI 318
.pypirc **318**
testpypi **319**
..... **320**
PyPI **320**
..... **321**
..... 321
..... **321**
..... 322

90: **323**
..... 323
..... 323
..... 323
..... 323
..... 324
Examples 324
..... 324
..... 325
..... 325
..... 326
..... **326**
..... 326
..... 326
..... **326**
..... **327**
..... **327**
..... 328
..... 328
..... 328
..... 329
..... 329
..... 329
Lambda/ 329
..... 331
..... 332
..... 333
..... 333
..... 334
..... 334
..... 336
Lambda 336

336
91: 338
Examples 338
338
338
Python 338
PYTHONSTARTUP 339
339
340
92: 341
341
Examples 341
341
342
343
343
343
del 344
del v 344
del v.name 345
del v[item] 345
del v[a:b] 345
345
345
346
346
global vs nonlocal Python 3 347
93: 348
348
348
348

Examples 348
348
349
352
352
352
353
353
353
354
354
355
355
355
94: 357
357
Examples 357
357
unittest.mock.create_autospec 357
test.TestCase 357
358
Unittests 359
pytest 360
95: GIL 362
362
GIL 362
GIL 362
GIL 362
GIL 362
362
Examples 363
Multiprocessing.Pool 363

David BeazleyGIL **363**
Cython nogil 364
David BeazleyGIL **364**
nogilCYTHON **364**
96: **365**
..... 365
Examples 365
PyDotPlus 365
..... 365
PyGraphviz 365
97: windowsvirtualenvwrapper **367**
Examples 367
windowsvirtualenvwrapper 367
98: **368**
..... 368
Examples 368
..... 368
99: **369**
..... 369
..... 369
..... 369
Examples 369
mapitertools.imapfuture_builtins.map 369
..... 370
..... 370
Map“None”python 2.x 371
..... 373
100: **375**
..... 375
..... 375
Examples 375
.....

375
.....376
del.....377
.....377
.....378
.....378
.....378
.....379
101: **380**
Examples.....380
inputraw_input.....380
.....380
.....380
.....381
stdin.....382
.....382
102: **384**
.....384
.....384
.....384
Examples.....384
ListStack.....384
.....385
103: **387**
Examples.....387
.....387
.....388
104: **389**
Examples.....389
.....389
.....391
105: **393**

Examples 393
..... 393
... . 393
106: **395**
..... 395
Examples 395
..... 395
..... 395
..... 396
..... 397
..... 397
while 398
107: **400**
..... 400
..... 400
Examples 400
UDP 400
UDP 400
TCP 401
TCP 401
Linux 405
108: **406**
..... 406
..... 406
Examples 406
..... 406
Popen 406
..... **406**
..... **406**
..... **406**
..... **406**

407

407

407

109: 408

408

408

408

Examples 408

408

dict 408

KeyError 408

409

410

410

410

dict 410

dict() 410

410

410

411

** 411

412

Python 3.5+ 412

Python 3.3+ 412

Python 2.x3.x 412

413

413

413

414

414

110: 416

. 417

Examples 417

. 417

str.casefold() 417

str.upper() 417

str.lower() 417

str.capitalize() 418

str.title() 418

str.swapcase() 418

str 418

. 419

str.split(sep=None, maxsplit=-1) 419

str.rsplit(sep=None, maxsplit=-1) 419

. 419

str.replace(old, new[, count]) 420

str.formatf-strings 420

. 421

str.count(sub[, start[, end]]) 421

. 422

str.startswith(prefix[, start[, end]]) 422

str.endswith(prefix[, start[, end]]) 423

. 423

str.isalpha 423

str.isupper str.islower str.istitle 424

str.isdecimal str.isdigit str.isnumeric 424

str.isalnum 425

str.isspace 425

str.translate 426

/ 426

str.strip([chars]) 426

str.rstrip([chars])str.lstrip([chars]) 427

. 427

428
428
string.ascii_letters 428
string.ascii_lowercase 428
string.ascii_uppercase 428
string.digits 429
string.hexdigits 429
string.octaldigits 429
string.punctuation 429
string.whitespace 429
string.printable 429
429
430
strunicode 430
431
111: 432
432
432
432
Examples 432
432
433
f-string 434
datetime 434
GetitemGetattr 434
435
436
436
437
437
438
Python 2.x 438
Python 3.2+ 438

438
112: 439
439
439
439
Examples 439
439
439
440
440
pycrypto 440
pycryptoRSA 441
pycryptoRSA 442
113: / 444
444
444
Examples 446
446
448
114: 450
Examples 450
CHello World 450
C Extensions 451
Cc ++Boost 451
C ++ 452
115: PythonSQL Server 457
Examples 457
457
116: 458
458
458

Examples 458
..... 458
..... 459
..... 459
__all__ 460
..... 460
..... 461
PEP8 461
..... 461
__import __ 461
..... 461
Python 2 462
Python 3 462
117: **463**
..... 463
Examples 463
@property 463
@property 463
gettersetterdeleter 464
..... 464
118: **467**
..... 467
Examples 467
..... 467
..... 467
119: **469**
Examples 469
..... 469
..... 469
..... 470
..... 470
```` ..... 470
.....
````

471
120: **472**
..... 472
Examples 472
..... 472
..... 474
..... 475
..... 477
int literals 478
..... 478
sys.argv [0] 479
h14 ..... **479**
..... 479
GIL 480
..... 481
..... 481
Pythonic JSON 482
121: CPython **483**
..... 483
..... 483
Examples 484
CPython 484
CPython 484
122: **486**
..... 486
Examples 486
if / else 486
..... 486
..... 487
..... 487
123: CSV **489**
..... 489

489

489

Examples 489

489

CSV 490

124: 491

491

491

491

Examples 491

491

492

492

492

break 492

continue 493

493

returnbreak 494

“else” 494

496

497

Loop 497

498

498

498

499

“” 499

499

125: 500

500

Examples 500

500

126: 502

Examples 502

..... 502

..... 502

..... 502

“__main__” 503

127: 504

Examples 504

..... 504

..... 506

128: abc 507

Examples 507

ABCMeta 507

/ABCMeta@abstractmethod 508

129: 510

Examples 510

python 510

130: 512

..... 512

..... 512

Examples 512

AND 512

OR 512

..... 513

..... 513

..... 513

NOT 514

..... 515

131: 517

Examples 517

..... 517

517
maxmin 517
517
518
518
518
519
NN 522
132: 524
Examples 524
524
524
133: / 526
526
Examples 526
526
526
134: 527
Examples 527
527
528
135: 530
530
Examples 530
str.indexstr.rindexstr.findstr.rfind 530
530
530
530
531
531
531
list.indextuple.index 531

dict....531
bisect.bisect_left....532
....533
__ content____iter__....534
136: 536
Examples....536
....536
Methodcaller....536
Itemgetter....536
137: XML....538
....538
Examples....538
ElementTree....538
XML....538
XML....539
iterparseXML....539
XPathXML....540
138: 543
....543
....543
....543
....543
Examples....543
....543
....543
....544
....544
139: 546
....546
....546
Examples....546
collections.Counter....546

collections.defaultdict 547
collections.OrderedDict 547
collections.namedtuple 549
collections.deque 549
collections.ChainMap 550
140: **552**
Examples 552
ceiltrunc 552
..... 554
..... 555
..... 555
..... 556
..... 556
..... 556
/ 556
..... 556
..... 556
..... 556
..... 557
NaN“” 557
..... 559
cmath 559
141: **562**
..... 562
..... 562
..... 562
Examples 562
JSON 562
Pickle 563
142: **564**
..... 564
Examples 564

MySQLdbMySQL....564
SQLite....566
SQLite....567
....567
h212....568
Connection....569
Cursor....569
SQLitePython....572
psycopg2PostgreSQL....573
....573
....573
....574
Oracle....575
....576
sqlalchemy....577
143: 578
Examples....578
iterablecollections.Counter....578
-scollections.Counter.most_common....578
list.counttuple.count....578
str.count....579
numpy....579
144: 580
....580
....580
Examples....580
....580
....581
append....581
insert....581
extendpython....581
fromlist....582

remove....582
pop....582
index....582
reversepython....582
buffer_info....583
count....583
tostring....583
tolistpython....583
fromstringchar....583
145: I / O.....584
....584
....584
....584
....584
....584
Examples....584
....584
....586
....586
....587
....588
....588
....589
....589
....589
mmap....590
....590
....590
146: 592
Examples....592
....592
147: 593
....

593
Examples 593
..... 593
..... 593
..... 594
..... 594
..... 595
..... 596
..... 598
..... 598
ISO 8601 598
..... 599
..... 599
..... 599
ISO 8601 600
..... 600
..... 600
..... 600
148: 601
Examples 601
..... 601
datetime 601
datetime 601
149: 602
Examples 602
..... 602
JPEG 602
150: 603
..... 603
Examples 603
Python 2.43.3 603
..... 603

151: **604**
..... 604
..... 604
Examples 604
ifelif 604
“” 604
..... 604
..... 605
..... 605
..... **605**
..... **605**
..... **606**
..... **606**
..... 607
cmp 608
..... 608
None 609
152: **610**
..... 610
Examples 610
..... 610
..... 610
153: **612**
..... 612
Examples 612
os.access 612
154: **613**
..... 613
..... 613
Examples 613
..... 613
.....

615

615

615

616

616

616

616

616

617

617

617

618

618

618

618

`re.finditer` 618

619

155: **620**

620

620

Examples 620

620

620

621

621

621

621

`is` vs` ==` 621

623

Python 623

156: **625**
..... 625
..... 625
Examples 625
..... 625
..... 626
157: **628**
..... 628
..... 628
Examples 628
..... 628
URL 628
HTTP 629
..... 630
..... 630
..... 631
..... **631**
..... **631**
..... **631**
158: `exec``eval` **632**
..... 632
..... 632
..... 632
Examples 632
exec 632
eval 632
..... 632
eval 633
ast.literal_evalPython 633
execevalast.literal_eval 633
159: Matplotlib **635**

635
Examples 635
Matplotlib 635
635
MATLAB 636
637
XYtwinx 639
twinyYX 640
160: 643
Examples 643
643
644
161: 645
645
645
Examples 645
645
next 645
645
646
646
647
648
- 649
649
649
650
650
650
651
162: urllib 652
Examples 652

HTTP GET 652
Python 2 652
Python 3 652
HTTP POST 652
Python 2 652
Python 3 653
..... 653
163: WebSockets 654
Examples 654
aiohttp 654
aiohttp 654
Websocket 655
164: 657
..... 657
..... 657
..... 657
Examples 657
..... 657
..... 658
..... 658
..... 659
..... 660
..... 660
..... 661
..... 661
..... 662
..... 663
165: 664
..... 664
..... 664
..... 664
Examples 664

664
665
665
__ getitem ____ setitem____delitem__ 665
667
667
667
166: 669
669
669
Examples 669
669
669
671
MethodsBuiltins 671
672
672
672
672
672
672
673
673
673
167: 674
Examples 674
674
168: 675
Examples 675
675
675

676

169: 677

677

Examples 677

IndentationErrors 677

IndentationError / SyntaxError 677

677

IndentationError / SyntaxErrorunindent 677

677

IndentationError 678

678

IndentationError 678

678

678

TypeErrors 678

TypeError[/] 678

678

TypeError[]'???"???' 679

679

TypeError'???' object/ 679

679

TypeError'???' 679

680

NameErrorname'???' 680

680

680

import 680

PythonLEGB 680

680

AssertError 680

KeyboardInterrupt 681
ZeroDivisionError 681
..... 681
170: 683
..... 683
..... 683
Examples 683
..... 683
virtualenv 683
..... 683
..... 683
..... 684
..... 684
..... 684
..... 684
..... 684
python 685
virtualenvwrapper 686
..... 686
..... 686
..... 686
..... 687
Unix / Linuxpython 687
virtualenv 687
Anaconda 688
..... 688
..... 688
..... 688
..... 688
..... 688
171: virtualenvwrapper 690
.....

690
Examples 690
virtualenvwrapper 690
172: **693**
..... 693
..... 693
..... 693
Examples 693
..... 693
..... 694
..... **694**
..... **695**
..... 695
..... **695**
..... **695**
..... 696
..... **696**
..... **696**
..... **696**
..... 697
..... 697
173: **698**
Examples 698
..... 698
..... 698
..... 698
..... 698
..... 698
174: **700**
..... 700
Examples 700
.....

700
.....700
175: 701
.....701
Examples.....701
Python ZipFile.extractallZIP.....701
Python TarFile.extractalltarball.....701
176: 702
.....702
Examples.....702
argparseHello.....702
docopt.....703
argparse.....703
argv.....704
argparse.....705
argparse.add_argument_group.....705
docoptdocopt_dispatch.....706
177: 708
Examples.....708
Python.....708
.....709
178: Python.....712
Examples.....712
.....712
.....712
.....712
.....712
.....713
.....713
179: 714
.....714
Examples.....714

714
Singleton 715
717
180: 719
719
719
Examples 719
719
719
719
720
720
docstrings 720
721
PEP 257 721
721
Google Python 722
181: PythonJavaScript 723
723
Examples 723
`in` 723
182: 724
724
Examples 724
724
183: 725
Examples 725
Python_pdb_ 725
IPythonipdb 725
726
184: CSV 727
Examples 727

TSV 727
..... 727
..... 727
..... 727
185: 728
Examples 728
/ Dunder 728
..... 729
..... 730
..... 731
..... 731
186: 734
..... 734
..... 734
Examples 734
..... 734
..... 734
..... 735
..... 735
NamedTuple 736
..... 736
187: 737
..... 737
..... 737
..... 737
..... 737
Examples 737
1 737
2 737
3 738
4 738
188: 740

740
740
Examples 741
741
741
189: 742
742
Examples 742
742
743
743
744
744
190: 746
746
746
Examples 746
python 746
191: 748
748
748
748
Examples 748
748
748
749
filterfalseifilterfalse 749
192: 751
751
Examples 751
1n 751
751

753
754
- 754
754
193: 757
757
757
757
757
Examples 757
757
194: 758
Examples 758
Conda 758
195: 760
760
Examples 760
760
196: 766
Examples 766
python 766
197: GZip 767
767
Examples 767
GNU zip 767
198: 768
768
768
768
Examples 768
768

reduce 769
769
/ 769
truthy / falsy 770
199: 771
771
Examples 771
771
200: 772
772
Examples 772
772
772
772
772
randintrandrangerandomuniform 772
randint 772
randrange 773
773
773
774
774
775
775
201: 776
Examples 776
776
202: Python 777
Examples 777
IronPython 777
777

777
Jython 777
777
778
Transcrypt 778
778
HTML 778
JavaScriptDOM 778
JavaScript 778
PythonJavaScript 779
779
203: 780
Examples 780
Pyglet 780
WAV 780
780
780
pythonffmpeg 780
Windows 780
204: 782
782
Examples 782
782
782
783
784
785
786
787
789

791

792

792

793

795

796

205: __str____repr__ **798**

798

798

798

Examples 798

798

799

1 **799**

2 **800**

..... **801**

801

eval-round-trip__repr __ 801

206: **803**

Examples 803

Ninja TwistTurtle Graphics 803

804

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: python-language

It is an unofficial and free Python Language ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Python Language.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: Python

Python。

- Python。 。
- Python。 PHPWeb。 Python Web。
- Python。 。

```
if something:
    x = 1
else:
    x = 'this is a string'
print(x)
```

- 。 ;

```
if something:
    x = 1
else:
    x = 'this is a string'
print(x)
```

- **:)** Python。 。 Python。 。

The Zen of Python。

Python。 Python 2.x。 Python 3.x。 。

PythonPython。

Python.orgCPython。 IronPython .NETPython Jython JavaPyPy Python。

Python 3.x

[3.7]	201758
3.6	20161223
3.5	2015913
3.4	2014317
3.3	2012929

3.2	2011-02-20
3.1	2009-06-26
3.0	2008-12-03

Python 2.x

2.7	2010-07-03
2.6	2008-10-02
2.5	2006-09-19
2.4	2004-11-30
2.3	2003-07-29
2.2	2001-12-21
2.1	2001415
2.0	20001016

Examples

PythonGuido van Rossum1991.Python。 。

Python

- Python 3.x。
- Python 2.x2020。 。 Python 2Python 3。

Python。 Python 3Python 2 。 Python。 。

Python

PythonWindowspython

```
$ python --version
```

Python 3.x 3.0

Python 3

```
$ python --version
```

Python 2.x 2.7

Python 2

```
$ python --version
```

Python 3`$ python --version`Python 2Python 2。 MacOSLinux。 `$ python3`Python 3。

IDLEPython

IDLEPythonPython。

IDLEHelloWorld

- IDLE。
  - WindowsWindows`All Programs`。
  - Windows 8+`IDLE`。
  - UnixMac`$ idle python_file.py`shell。
- shell。

shell

```
$ python --version
```

```
$ python --version
```

`Enter` 。

```
$ python --version
```

Hello World Python

`hello.py`

Python 3.x 3.0

```
$ python --version
```

Python 2.x 2.6

Python 2Python 3 `printimport`

```
$ python --version
```

Python 2`__future__`Python 3。

Python 2.x 2.7

Python 2。 Python 3。

```
$ python --version
```

`hello.py`。

`python hello.py Enter`。

```
$ python --version
```

`Hello, World`。

`hello.py`。 Linux“user”`python /home/user/hello.py`。

Python shell

`python`Python shell。 PythonREPL“”。

```
$ python --version
```

Python 3`python3`。

```
$ python --version
```

`python -i <file.py>`。

```
$ python --version
```

Python shell

```
$ python --version
```

```
$ python --version
```

`CTRL + D`shell。

Interpreter shell`CTRL + C`.

Python shell。

Python shell。

shell

- python。
- Python。
- 。

  。

- https://www.python.org/shell/ - PythonPython shell。
- https://ideone.com/ - 。
- https://repl.it/languages/python3 - IDE。 Python。
- https://www.tutorialspoint.com/execute_python_online.php - UNIX shell。
- http://rextester.com/l/python3_online_compiler - IDE

Pythonshell

```
$ python --version
```

shell。

- PyPAPythonPIP 。 `pip install <the package name>` 。 `pip install numpy` 。 WindowspipPATH。 `python -m pip install <the package name>`

Shell - Pythonshell。 ShellPython。 shellIDLE - GUI IPython - 。

- .pyshell IDE PyCharm Jupyter/。 ;。

PythonPython。

PEP8Python。 。

Python。

```
<variable name> = <value>
```

Python=。 。 。

```
<variable name> = <value>
```

。 。

```
<variable name> = <value>
```

python。

```
<variable name> = <value>
```

---

1. 。

```
<variable name> = <value>
```

2. 。

```
<variable name> = <value>
```

3. 。

```
<variable name> = <value>
```

PythonPython

```
<variable name> = <value>
```

python。

= = 。 what = does 。

```
<variable name> = <value>
```

pi = 3.14 pi 3.14 。 。

。 =

```
<variable name> = <value>
```

。 _

```
<variable name> = <value>
```

_。 “”

```
<variable name> = <value>
```

。

```
<variable name> = <value>
```

a bc 1int。 a bcint。

```
<variable name> = <value>
```

list dictint string tuple

```
<variable name> = <value>
```

。 。

```
<variable name> = <value>
```

---

python。 。

```
<variable name> = <value>
```

Python - =。

```
<variable name> = <value>
```

=。 int2a astring“”。

input Python 2.xraw_input Python 2.xinput

Python 2.x 2.3

```
name = raw_input("What is your name? ")
# Out: What is your name? _
```

Python2input() - PythonPython3eval(input())。 。

Python 3.x 3.0

```
name = raw_input("What is your name? ")
# Out: What is your name? _
```

Python 3。

。 。

```
name = raw_input("What is your name? ")
# Out: What is your name? _
```

“Bob”Enternam"Bob"

```
name = raw_input("What is your name? ")
# Out: What is your name? _
```

`inputstr`。 `strstr`

```
name = raw_input("What is your name? ")
# Out: What is your name? _
```

`try` / `except` 。 `raw_input`int uncastable`ValueError` 。

IDLE - Python GUI

IDLEPython。 IDLEpython。 WindowsPython。

IDLE

-
- Python shell
-
- Python
- Python.pyIDLE。

IDLE`F5run Python Shell`。 IDLE。

。

- Windows`python` 。 `"'python' is not recognized"`Python`PATH`。 “”“”“”“”。 “”“”...”。 `PATH`PythonScript `C:\Python27;C:\Python27\Scripts` 。 。

  Python`python.exe`。 `python27.exe``python27`Python。

  Python Launcher for Windows。 `py -[xy]``python[xy]`Python。 Python 2`py -2` Python 3`py -3` 。

- **Debian / Ubuntu / MacOS**

  `pythonPATH`。

  Debian / Ubuntu / MacOS`python` for Python 2.x`python3` for Python 3.x.

  `which python`Python。

- **Arch Linux**

  Arch LinuxPythonPython 3Python 3.x`python`python3 Python 2.x`python2` 。

- Python 3`python`python3 。 Python 2`python2` 。

`bool` `True`False 。 `and` `or` `not`。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
```

```
not x     # if x is True then False, otherwise True
```

Python 2.xPython 3.x`int` ◦ `bool`int `True`False

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

`True`False10


```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```


- `int`


  ```
  x or y    # if x is False then y otherwise x
  x and y   # if x is False then x otherwise y
  not x     # if x is True then False, otherwise True
  ```


  Python◦

  Python`longint`◦ ◦

- `float` ;CPython `float`C double◦


  ```
  x or y    # if x is False then y otherwise x
  x and y   # if x is False then x otherwise y
  not x     # if x is True then False, otherwise True
  ```


- `complex`


  ```
  x or y    # if x is False then y otherwise x
  x and y   # if x is False then x otherwise y
  not x     # if x is True then False, otherwise True
  ```


`<` `<=` `>>=`TypeError◦

Python 3.x 3.0

- `str` **unicode** ◦ `'hello'`
- `bytes` ◦ `b'hello'`

Python 2.x 2.7

- `str` ◦ `'hello'`
- `bytes` `str`
- `unicode` **unicode** ◦ `u'hello'`

Python`set``dict` 。

- `str` `bytes` `unicode`

- `reversed` `str``reversed`

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

- `tuple` `n` `n >= 0` 。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

;;

- `list` `n` `n >= 0`

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

;。

- `set` 。 。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

- `dict` ; 。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

`__hash__()` `__eq__()` `__hash__()` 。 Hashable。

- `True` `bool`
- `False` `bool`false
- `None` 。
- `Ellipsis` `...` Python3 +Python2.7 +。 `numpy`“”。
- `NotImplemented` PythonPython。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

Python 3.x 3.0

None◦ < <= >= > TypeError ◦

Python 2.x 2.7

None None < -32True ◦

---

pythontypetype ◦

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

isinstance◦ ◦

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

type()isinstance() Pythonisinstancetype

NoneType

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

◦

'123'strint◦

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

float'123.456'◦

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

◦

- `b'foo bar'` Python 3`bytes` Python 2`str`
- `u'foo bar'` Python 3`str` Python 2`unicode`
- `'foo bar'` `str`
- `r'foo bar'`

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

---

mutable 。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

。

```
x or y    # if x is False then y otherwise x
x and y   # if x is False then x otherwise y
not x     # if x is True then False, otherwise True
```

`x` `x`。 `x`。

。

- `int` `long` `float` `complex`
- `str`
- `bytes`
- `tuple`
- `frozenset`

- `bytearray`
- `list`
- `set`
- `dict`

Python。 。

```
>>> pow(2,3)    #8
```

python`dir()`.。 。

```
>>> pow(2,3)    #8
```

`help` 。

```
>>> pow(2,3)    #8
```

◦ math◦

```
>>> pow(2,3)    #8
```

◦

```
>>> pow(2,3)    #8
```

__doc__

```
>>> pow(2,3)    #8
```

◦ helloWorld.py

```
>>> pow(2,3)    #8
```

```
>>> pow(2,3)    #8
```

- dir

```
>>> pow(2,3)    #8
```

str◦ print

```
>>> pow(2,3)    #8
```

Python◦ Python◦ ◦

Python : ◦ Python if◦ ◦

```
def my_function():    # This is a function definition. Note the colon (:)
    a = 2             # This line belongs to the function because it's indented
    return a          # This line also belongs to the same function
print(my_function())  # This line is OUTSIDE the function block
```

```
def my_function():    # This is a function definition. Note the colon (:)
    a = 2             # This line belongs to the function because it's indented
    return a          # This line also belongs to the same function
print(my_function())  # This line is OUTSIDE the function block
```

```
def my_function():    # This is a function definition. Note the colon (:)
    a = 2             # This line belongs to the function because it's indented
    return a          # This line also belongs to the same function
print(my_function())  # This line is OUTSIDE the function block
```

```
def my_function():    # This is a function definition. Note the colon (:)
    a = 2             # This line belongs to the function because it's indented
    return a          # This line also belongs to the same function
```

```
print(my_function())  # This line is OUTSIDE the function block
```

`IndentationError。 pass`

```
def my_function():    # This is a function definition. Note the colon (:)
    a = 2             # This line belongs to the function because it's indented
    return a          # This line also belongs to the same function
print(my_function())  # This line is OUTSIDE the function block
```

4。

PEP 8 Python。

Python 3.x 3.0

Python 3。 `Inconsistent use of tabs and spaces in indentation。`

Python 2.x 2.7

Python 2;。 **8** 。 **4**。

PEP 8

`-t`Python 2。 `-tt`。

“”。 '\ t'`Tab`。

- 8 - 。 。
- `Tab`4。

Pythonautopep8pep8 -conformant。 Python reindent.py

Python。 `intstr`。

`list`Python。 JavaScript。 PythonPython。

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

Python

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
```

```
string_list = ['abc', 'defghi']
```

◦ Python01

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

-1◦

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

/

L.append(object)L.append(object) None ◦

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

◦ L.insert(index, object)

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

L.remove(value)L.remove(value) None

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

x◦ ◦

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

L.pop([index])L.pop([index])

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

tuple。 。 IP。

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

。 Python。

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

tuple

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

Pythondictionary。 。 。

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

JSON。 PythonjsonJSON。

set。 。 setlist。

setdictionary

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

listset

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

`inset`

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

`set` 。

defaultdict

`defaultdict`。 `defaultdict`。

`defaultdict`KeyError。 。

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

python

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

`defaultdict` 。 collections。

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

Give。 dict

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

dictpythonBoston

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

`dictionary`

```
int_list = [1, 2, 3]
string_list = ['abc', 'defghi']
```

Python。 Python

```
>>> help()
```

```
>>> help()
```

```
>>> help()
```

```
>>> help()
```

```
>>> help()
```

```
>>> help()
```

```
>>> help()
```

。

quit

。

.py。

```
# hello.py
def say_hello():
    print("Hello!")
```

。

。 Python lib。

```
# hello.py
def say_hello():
    print("Hello!")
```

。

```
# hello.py
def say_hello():
    print("Hello!")
```

。

```
# hello.py
def say_hello():
    print("Hello!")
```

。

```
# hello.py
def say_hello():
    print("Hello!")
```

。

```
# hello.py
def say_hello():
    print("Hello!")
```

```
# hello.py
def say_hello():
    print("Hello!")
```

python`__init__.py`。

- strrepr

。

`repr(x)` `x.__repr__()` x。 eval。

`str(x)` `x.__str__()` 。 。

`eval()`。 。 。

STR

。 this`repr(object)` `str(object)` `eval()`。 “”。 `''` 。

1

```
s = """w'o"w"""
repr(s) # Output: '\'w\\\'o"w\''
str(s)  # Output: 'w\'o"w'
eval(str(s)) == s  # Gives a SyntaxError
eval(repr(s)) == s # Output: True
```

2

```
s = """w'o"w"""
repr(s) # Output: '\'w\\\'o"w\''
str(s)  # Output: 'w\'o"w'
eval(str(s)) == s  # Gives a SyntaxError
eval(repr(s)) == s # Output: True
```

```
s = """w'o"w"""
repr(s) # Output: '\'w\\\'o"w\''
str(s)  # Output: 'w\'o"w'
eval(str(s)) == s  # Gives a SyntaxError
eval(repr(s)) == s # Output: True
```

```
s = """w'o"w"""
repr(s) # Output: '\'w\\\'o"w\''
str(s)  # Output: 'w\'o"w'
```

```
eval(str(s)) == s  # Gives a SyntaxError
eval(repr(s)) == s # Output: True
```

pip

pythonPyPI `pip`。 python.orgPython 2> = 2.7.9Python 3> = 3.4`pip`。 Linux* nix `pip`。

Python 2Python 3 `pip`Python 2 `pip3`Python 3.`pip`Python 2`pip3`Python 3。

/

```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```

/Python

```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```

`xxx`。

```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```

。

```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```

```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```

`pip`。

---

pip

- LinuxmacOS X

  ```
  $ pip search <query>
  # Searches for packages whose name or summary contains <query>
  ```

  Linuxpip`sudo`

- Windows

```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```


```
$ pip search <query>
# Searches for packages whose name or summary contains <query>
```


pip 。

Python 2.7.x3.x.

Python 2.7Python 3.x。

https://www.python.org/downloads/Python 2.7。 MSI。 。

Python

```
C:\Python27\
```

PATH。

PythonC\ Python27PATH

```
C:\Python27\
```

Pythoncmd

```
C:\Python27\
```

Python 2.x3.x Side-By-Side

WindowsPython 2.x3.x

1. MSIPython 2.x.
    - Python。
    - Python`PATH` `python`Python 2.x.
2. Python 3.x.
    - Python。
    - Python`PATH``python`Python 3.x.Python 2.x `PATH`PATH。
    - `py launcher` 。

Python 3PythonPython 2.xPython 3.x

```
C:\Python27\
```

Python`pip`

```
C:\Python27\
```

LINUX

CentOSFedoraRedhat EnterpriseRHELUbuntuPython 2.7。

LinuxPython 2.7

```
C:\Python27\
```

PATHpython。 python`/root/python-2.7.Xexport PATH = $PATH:/root/python-2.7.X`

Python

```
C:\Python27\
```

Ubuntu

Python 3.6Ubuntu 16.1017.043.6。 Ubuntu 16.04

```
C:\Python27\
```

macOSPython 2.7.10Python。

OS XPython。 OS XPython。

Homebrew

```
C:\Python27\
```

Python 2.7

```
C:\Python27\
```

Python 3.x`brew install python3`。

Python https://riptutorial.com/zh-TW/python/topic/193/python

2: * args** kwargs

1. `argskwargs`。

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

2. `argskwargs`

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

3. `*args`。 `*args`。 Python 3。

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

4. `**kwargs`。

```
def func(*args, **kwargs):
    print(args)
    print(kwargs)
```

Examples

* args

star *

```
def print_args(farg, *args):
   print("formal arg: %s" % farg)
   for arg in args:
       print("another positional arg: %s" % arg)
```


```
def print_args(farg, *args):
   print("formal arg: %s" % farg)
   for arg in args:
       print("another positional arg: %s" % arg)
```


fargargs。

** kwargs

`**`

```
def print_kwargs(**kwargs):
    print(kwargs)
```

Python


```
def print_kwargs(**kwargs):
    print(kwargs)
```


** kwargs。


```
def print_kwargs(**kwargs):
    print(kwargs)
```


`kwargs`;`kwargs`


```
def print_kwargs(**kwargs):
    print(kwargs)
```


`print_kwargs(a="two", b=1)`


```
def print_kwargs(**kwargs):
    print(kwargs)
```


* args

`*args`。 `__init__`

```
class A(object):
    def __init__(self, b, c):
        self.y = b
        self.z = c

class B(A):
    def __init__(self, a, *args, **kwargs):
        super(B, self).__init__(*args, **kwargs)
```

```
        self.x = a
```

a。

```
class A(object):
    def __init__(self, b, c):
        self.y = b
        self.z = c

class B(A):
    def __init__(self, a, *args, **kwargs):
        super(B, self).__init__(*args, **kwargs)
        self.x = a
```

B__init__1, 2, 3。 a 1 a == 1。

*args 1, 2 *args。 args == [2, 3]。

argsA__init__。 Pythonargs“”。 B__init__A__init__2, 3 A(2, 3)。

xa 1。

** kwargs

;

```
def test_func(arg1, arg2, arg3): # Usual function with three arguments
   print("arg1: %s" % arg1)
   print("arg2: %s" % arg2)
   print("arg3: %s" % arg3)

# Note that dictionaries are unordered, so we can switch arg2 and arg3. Only the names matter.
kwargs = {"arg3": 3, "arg2": "two"}

# Bind the first argument (ie. arg1) to 1, and use the kwargs dictionary to bind the others
test_var_args_call(1, **kwargs)
```

* args

*

```
def print_args(arg1, arg2):
    print(str(arg1) + str(arg2))

a = [1,2]
b = tuple([3,4])

print_args(*a)
# 12
print_args(*b)
# 34
```

。

python`zip*`

```
def print_args(arg1, arg2):
    print(str(arg1) + str(arg2))

a = [1,2]
b = tuple([3,4])

print_args(*a)
# 12
print_args(*b)
# 34
```

Python 3。star *。 *。 。

```
def print_args(arg1, *args, keyword_required, keyword_only=True):
    print("first positional arg: {}".format(arg1))
    for arg in args:
        print("another positional arg: {}".format(arg))
    print("keyword_required value: {}".format(keyword_required))
    print("keyword_only value: {}".format(keyword_only))

print(1, 2, 3, 4) # TypeError: print_args() missing 1 required keyword-only argument:
'keyword_required'
print(1, 2, 3, keyword_required=4)
# first positional arg: 1
# another positional arg: 2
# another positional arg: 3
# keyword_required value: 4
# keyword_only value: True
```

kwarg

```
def foobar(foo=None, bar=None):
    return "{}{}".format(foo, bar)

values = {"foo": "foo", "bar": "bar"}

foobar(**values) # "foobar"
```

** kwargs

** kwargs

```
def fun(**kwargs):
    print kwargs.get('value', 0)

fun()
# print 0
fun(value=1)
# print 1
```

* args** kwargs https://riptutorial.com/zh-TW/python/topic/2475/--args---kwargs

3: __name__

__name____name__。

Python__name__。 '__main__'。 。

obj.__name__ 。

Examples

__name__ =='__ main__'

__name__ 。 import 。 if __name__ == '__main__'。

module_1.py

```
import module2.py
```

module2.py

1

module2.py

```
import module2.py
```

module1.pyhello
module2.pyhello

2

module2.py

```
import module2.py
```

module1.py
module2.pyhello

function_class_or_module .__ name__

__name__。

```
import os

class C:
    pass
```

```
def f(x):
    x += 2
    return x


print(f)
# <function f at 0x029976B0>
print(f.__name__)
# f

print(C)
# <class '__main__.C'>
print(C.__name__)
# C

print(os)
# <module 'os' from '/spam/eggs/'>
print(os.__name__)
# os
```

__name__。

```
import os

class C:
    pass

def f(x):
    x += 2
    return x


print(f)
# <function f at 0x029976B0>
print(f.__name__)
# f

print(C)
# <class '__main__.C'>
print(C.__name__)
# C

print(os)
# <module 'os' from '/spam/eggs/'>
print(os.__name__)
# os
```

```
import os

class C:
    pass

def f(x):
    x += 2
    return x
```

```
print(f)
# <function f at 0x029976B0>
print(f.__name__)
# f

print(C)
# <class '__main__.C'>
print(C.__name__)
# C

print(os)
# <module 'os' from '/spam/eggs/'>
print(os.__name__)
# os
```

logging__name__

```
logger = logging.getLogger(__name__)
```

。

__name__ https://riptutorial.com/zh-TW/python/topic/1223/--name--

4: 2to3

- $ 2to3 [-options] path / to / file.py.

filename / directory_name	2to3。 Python。
-f FIX - fix = FIX	;。 `--list-fixes`
-j PROCESSES - processes = PROCESSES	23
-x NOFIX - nofix = NOFIX	
-l - list-fixes	
-p - print-function	`print()`
-v - verbose	
--no-diff	
-w	
-n - nobackups	
-o OUTPUT_DIR - output-dir = OUTPUT_DIR	。 `-n`。
-W -	。 `-o`。 `-w` 。
--add= ADD_SUFFIX	。 `-n` 。 - `--add-suffix='3'.py3`。

2to3pythonPython 2.xPython 3.x。 Python 2.xPython 3.x。

2to3lib2to3 。 lib2to32to3。

Examples

Python2.x。 `example.py`

Python 2.x 2.0

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
greet(name)
```

◦ `raw_input()`Python 3.x`input() print`◦ 2to3Python 3.x◦

Unix

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
greet(name)
```

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
greet(name)
```

◦

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
greet(name)
```

-w◦ -n`example.py.bak`◦

Unix

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
greet(name)
```

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
greet(name)
```

`example.py`Python 2.xPython 3.x◦

`example.py`Python3.x

Python 3.x 3.0

```
def greet(name):
    print "Hello, {0}!".format(name)
print "What's your name?"
name = raw_input()
```

```
greet(name)
```

2to3 https://riptutorial.com/zh-TW/python/topic/5320/2to3

5: ArcPy

ArcPyda。

arcpy.da.SearchCursorarcpy.SearchCursor。

arcpy.daArcGIS 10.1 for Desktop。

Examples

“”

C\ TempTest.gdbTestFCTestField

```
with arcpy.da.SearchCursor(r"C:\Temp\Test.gdb\TestFC",["TestField"]) as cursor:
    for row in cursor:
        print row[0]
```

createDissolvedGDBgdb

```
def createDissolvedGDB(workspace, gdbName):
    gdb_name = workspace + "/" + gdbName + ".gdb"

    if(arcpy.Exists(gdb_name):
        arcpy.Delete_management(gdb_name)
        arcpy.CreateFileGDB_management(workspace, gdbName, "")
    else:
        arcpy.CreateFileGDB_management(workspace, gdbName, "")

    return gdb_name
```

ArcPy https://riptutorial.com/zh-TW/python/topic/4693/arcpy

6: Asyncio

Examples

Python 3.5+ `asyncio`Python 3.5。

Python 3.x 3.5

Python 3.5`asyncawait`。 `await func()`。

```
import asyncio

async def main():
    print(await func())

async def func():
    # Do time intensive stuff...
    return "Hello, world!"

if __name__ == "__main__":
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
```

Python 3.x 3.3 3.5

Python 3.5 `@asyncio.coroutine`。 。 `yield from func()`yield from func()。

```
import asyncio

async def main():
    print(await func())

async def func():
    # Do time intensive stuff...
    return "Hello, world!"

if __name__ == "__main__":
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
```

Python 3.x 3.5

```
import asyncio

async def main():
    print(await func())

async def func():
    # Do time intensive stuff...
    return "Hello, world!"

if __name__ == "__main__":
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
```

Python 3.5+ async / await

asyncioconcurrent.futuresExecutor。 run_in_executor()Executor Callable Callable。

Executor

```
import asyncio
from concurrent.futures import ThreadPoolExecutor

def func(a, b):
    # Do time intensive stuff...
    return a + b

async def main(loop):
    executor = ThreadPoolExecutor()
    result = await loop.run_in_executor(executor, func, "Hello,", " world!")
    print(result)

if __name__ == "__main__":
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main(loop))
```

“” ExecutorExecutor 。 Executorset_default_executor()。

```
import asyncio
from concurrent.futures import ThreadPoolExecutor

def func(a, b):
    # Do time intensive stuff...
    return a + b

async def main(loop):
    executor = ThreadPoolExecutor()
    result = await loop.run_in_executor(executor, func, "Hello,", " world!")
    print(result)

if __name__ == "__main__":
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main(loop))
```

concurrent.futuresExecutor ThreadPoolExecutorProcessPoolExecutor 。 ThreadPoolExecutor5. ThreadPoolExecutorCPUI / O。 ProcessPoolExecutor ProcessPoolExecutor。 ProcessPoolExecutor。 。 ExecutorThreadPoolExecutor 。

UVLoop

uvloop libuv asyncio.AbstractEventLoop nodejs。 99asyncioasyncio.EventLoop 。 uvloop Windows pip install uvloop。

```
import asyncio
import uvloop

if __name__ == "__main__":
```

```
    asyncio.set_event_loop(uvloop.new_event_loop())
    # Do your stuff here ...
```

EventLoopPolicyuvloop ◦

```
import asyncio
import uvloop

if __name__ == "__main__":
    asyncio.set_event_loop(uvloop.new_event_loop())
    # Do your stuff here ...
```

Event ◦

◦

```
import asyncio

# event trigger function
def trigger(event):
    print('EVENT SET')
    event.set()  # wake up coroutines waiting

# event consumers
async def consumer_a(event):
    consumer_name = 'Consumer A'
    print('{} waiting'.format(consumer_name))
    await event.wait()
    print('{} triggered'.format(consumer_name))

async def consumer_b(event):
    consumer_name = 'Consumer B'
    print('{} waiting'.format(consumer_name))
    await event.wait()
    print('{} triggered'.format(consumer_name))

# event
event = asyncio.Event()

# wrap coroutines in one future
main_future = asyncio.wait([consumer_a(event),
                            consumer_b(event)])

# event loop
event_loop = asyncio.get_event_loop()
event_loop.call_later(0.1, functools.partial(trigger, event))  # trigger event in 0.1 sec

# complete main_future
done, pending = event_loop.run_until_complete(main_future)
```

B

B
A

Websocket

asyncioecho websocket。 /。 websocketmain。 。

```
import asyncio
import aiohttp

session = aiohttp.ClientSession()                          # handles the context manager
class EchoWebsocket:

    async def connect(self):
        self.websocket = await session.ws_connect("wss://echo.websocket.org")

    async def send(self, message):
        self.websocket.send_str(message)

    async def receive(self):
        result = (await self.websocket.receive())
        return result.data

async def main():
    echo = EchoWebsocket()
    await echo.connect()
    await echo.send("Hello World!")
    print(await echo.receive())                            # "Hello World!"


if __name__ == '__main__':
    # The main loop
    loop = asyncio.get_event_loop()
    loop.run_until_complete(main())
```

asyncio

asnycio-GIL。

asyncio asyncio。 asyncio.sleep 。 await。 time.sleep。 time.sleep。

requests。 asyncio。 aiohttp asyncio。 。

- CPUasyncio 。 threadsmultiprocessing 。
- IOasyncio。

Asyncio https://riptutorial.com/zh-TW/python/topic/1319/asyncio

7: base64

Base 64radix 64ASCII.base64Python。 。 base64。

- base64.b64encodesaltchars = None
- base64.b64decodesaltchars = Nonevalidate = False
- base64.standard_b64encodeS
- base64.standard_b64decodeS
- base64.urlsafe_b64encodeS
- base64.urlsafe_b64decodeS
- base64.b32encodeS
- base64.b32decodeS
- base64.b16encodeS
- base64.b16decodeS
- base64.a85encodebfoldspaces = Falsewrapcol = 0pad = Falseadobe = False
- base64.a85decodebfoldpaces = Falseadobe = Falseignorechars = b'\ t \ n \ r \'\ n'
- base64.b85encodebpad = False
- base64.b85decodeb

`base64.b64encode(s, altchars=None)`	
altchars	2+Base64“+”“=”。 。
`base64.b64decode(s, altchars=None, validate=False)`	
altchars	2+Base64“+”“=”。 。
	valideTrueBase64
`base64.standard_b64encode(s)`	
`base64.standard_b64decode(s)`	
`base64.urlsafe_b64encode(s)`	
`base64.urlsafe_b64decode(s)`	
`b32encode(s)`	
`b32decode(s)`	
`base64.b16encode(s)`	

`base64.b16decode(s)`	
`base64.a85encode(b, foldspaces=False, wrapcol=0, pad=False, adobe=False)`	
b	
foldspaces	foldspacesTrue“y”4。
wrapcol	0
	padTrue4
	adobeTrueAdobe'<'''>。
`base64.a85decode(b, foldspaces=False, adobe=False, ignorechars=b'\t\n\r\v')`	
b	
foldspaces	foldspacesTrue“y”4。
	adobeTrueAdobe'<'''>。
ignorechars	
`base64.b85encode(b, pad=False)`	
b	
	padTrue4
`base64.b85decode(b)`	
b	

Python 3.4base64`bytesbytes bytearray`。 。

Examples

Base64

base64

```
import base64
```

base64 。 Python。 `UTF-8`Python。

```
import base64
```

```
b'Hello World!'
```

```
b。
```

Base64`base64.b64encode()`

```
import base64
```

```
b'SGVsbG8gV29ybGQh'
```

bytes。 Python`decode()`UTF-8

```
import base64
```

```
SGVsbG8gV29ybGQh
```

```
base64.b64decode()
```

```
import base64
```

```
import base64
```

Base32

base64Base32。 Base64

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# Base32 Encode the bytes
e = base64.b32encode(b)
# Decoding the Base32 bytes to string
s1 = e.decode("UTF-8")
# Printing Base32 encoded string
print("Base32 Encoded:", s1)
# Encoding the Base32 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the Base32 bytes
d = base64.b32decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# Base32 Encode the bytes
```

```
e = base64.b32encode(b)
# Decoding the Base32 bytes to string
s1 = e.decode("UTF-8")
# Printing Base32 encoded string
print("Base32 Encoded:", s1)
# Encoding the Base32 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the Base32 bytes
d = base64.b32decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```

Base16

base64Base16。16 。Base64Base32

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# Base16 Encode the bytes
e = base64.b16encode(b)
# Decoding the Base16 bytes to string
s1 = e.decode("UTF-8")
# Printing Base16 encoded string
print("Base16 Encoded:", s1)
# Encoding the Base16 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the Base16 bytes
d = base64.b16decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# Base16 Encode the bytes
e = base64.b16encode(b)
# Decoding the Base16 bytes to string
s1 = e.decode("UTF-8")
# Printing Base16 encoded string
print("Base16 Encoded:", s1)
# Encoding the Base16 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the Base16 bytes
d = base64.b16decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```

ASCII85

Adobe**ASCII85** Base85。 Adobe PDF。 Python 3.4。 `base64.a85encode()base64.a85encode()`

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# ASCII85 Encode the bytes
e = base64.a85encode(b)
# Decoding the ASCII85 bytes to string
s1 = e.decode("UTF-8")
# Printing ASCII85 encoded string
print("ASCII85 Encoded:", s1)
# Encoding the ASCII85 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the ASCII85 bytes
d = base64.a85decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```


```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# ASCII85 Encode the bytes
e = base64.a85encode(b)
# Decoding the ASCII85 bytes to string
s1 = e.decode("UTF-8")
# Printing ASCII85 encoded string
print("ASCII85 Encoded:", s1)
# Encoding the ASCII85 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the ASCII85 bytes
d = base64.a85decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```


Base85

Base64Base32Base16Base85`base64.b85encode()base64.b85decode()`

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# Base85 Encode the bytes
e = base64.b85encode(b)
# Decoding the Base85 bytes to string
s1 = e.decode("UTF-8")
# Printing Base85 encoded string
print("Base85 Encoded:", s1)
# Encoding the Base85 encoded string into bytes
b1 = s1.encode("UTF-8")
```

```
# Decoding the Base85 bytes
d = base64.b85decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```

```
import base64
# Creating a string
s = "Hello World!"
# Encoding the string into bytes
b = s.encode("UTF-8")
# Base85 Encode the bytes
e = base64.b85encode(b)
# Decoding the Base85 bytes to string
s1 = e.decode("UTF-8")
# Printing Base85 encoded string
print("Base85 Encoded:", s1)
# Encoding the Base85 encoded string into bytes
b1 = s1.encode("UTF-8")
# Decoding the Base85 bytes
d = base64.b85decode(b1)
# Decoding the bytes to string
s2 = d.decode("UTF-8")
print(s2)
```

base64 https://riptutorial.com/zh-TW/python/topic/8678/base64

8: ChemPy - python

ChemPypython。 Python。

Examples

```
from chempy import Substance
ferricyanide = Substance.from_formula('Fe(CN)6-3')
ferricyanide.composition == {0: -3, 26: 1, 6: 6, 7: 6}
True
print(ferricyanide.unicode_name)
Fe(CN)₆³⁻
 print(ferricyanide.latex_name + ", " + ferricyanide.html_name)
Fe(CN)_{6}^{3-}, Fe(CN)<sub>6</sub><sup>3-</sup>
 print('%.3f' % ferricyanide.mass)
211.955
```

0。

```
 from chempy import balance_stoichiometry  # Main reaction in NASA's booster rockets:
 reac, prod = balance_stoichiometry({'NH4ClO4', 'Al'}, {'Al2O3', 'HCl', 'H2O', 'N2'})
 from pprint import pprint
 pprint(reac)
{'Al': 10, 'NH4ClO4': 6}
 pprint(prod)
{'Al2O3': 5, 'H2O': 9, 'HCl': 6, 'N2': 3}
 from chempy import mass_fractions
 for fractions in map(mass_fractions, [reac, prod]):
...     pprint({k: '{0:.3g} wt%'.format(v*100) for k, v in fractions.items()})
...
{'Al': '27.7 wt%', 'NH4ClO4': '72.3 wt%'}
{'Al2O3': '52.3 wt%', 'H2O': '16.6 wt%', 'HCl': '22.4 wt%', 'N2': '8.62 wt%'}
```

```
from chempy import Equilibrium
 from sympy import symbols
 K1, K2, Kw = symbols('K1 K2 Kw')
 e1 = Equilibrium({'MnO4-': 1, 'H+': 8, 'e-': 5}, {'Mn+2': 1, 'H2O': 4}, K1)
 e2 = Equilibrium({'O2': 1, 'H2O': 2, 'e-': 4}, {'OH-': 4}, K2)
 coeff = Equilibrium.eliminate([e1, e2], 'e-')
 coeff
[4, -5]
 redox = e1*coeff[0] + e2*coeff[1]
 print(redox)
20 OH- + 32 H+ + 4 MnO4- = 26 H2O + 4 Mn+2 + 5 O2; K1**4/K2**5
 autoprot = Equilibrium({'H2O': 1}, {'H+': 1, 'OH-': 1}, Kw)
 n = redox.cancel(autoprot)
 n
20
 redox2 = redox + n*autoprot
 print(redox2)
12 H+ + 4 MnO4- = 4 Mn+2 + 5 O2 + 6 H2O; K1**4*Kw**20/K2**5
```

```
 from chempy import Equilibrium
```

```
 from chempy.chemistry import Species
 water_autop = Equilibrium({'H2O'}, {'H+', 'OH-'}, 10**-14)  # unit "molar" assumed
 ammonia_prot = Equilibrium({'NH4+'}, {'NH3', 'H+'}, 10**-9.24)  # same here
 from chempy.equilibria import EqSystem
 substances = map(Species.from_formula, 'H2O OH- H+ NH3 NH4+'.split())
 eqsys = EqSystem([water_autop, ammonia_prot], substances)
 print('\n'.join(map(str, eqsys.rxns)))  # "rxns" short for "reactions"
H2O = H+ + OH-; 1e-14
NH4+ = H+ + NH3; 5.75e-10
 from collections import defaultdict
 init_conc = defaultdict(float, {'H2O': 1, 'NH3': 0.1})
 x, sol, sane = eqsys.root(init_conc)
 assert sol['success'] and sane
 print(sorted(sol.keys()))  # see package "pyneqsys" for more info
['fun', 'intermediate_info', 'internal_x_vecs', 'nfev', 'njev', 'success', 'x', 'x_vecs']
 print(', '.join('%.2g' % v for v in x))
1, 0.0013, 7.6e-12, 0.099, 0.0013
```

```
 from chempy.electrolytes import ionic_strength
     ionic_strength({'Fe+3': 0.050, 'ClO4-': 0.150}) == .3
    True
```

```
from chempy import ReactionSystem  # The rate constants below are arbitrary
 rsys = ReactionSystem.from_string("""2 Fe+2 + H2O2 -> 2 Fe+3 + 2 OH-; 42
     2 Fe+3 + H2O2 -> 2 Fe+2 + O2 + 2 H+; 17
     H+ + OH- -> H2O; 1e10
     H2O -> H+ + OH-; 1e-4
     Fe+3 + 2 H2O -> FeOOH(s) + 3 H+; 1
     FeOOH(s) + 3 H+ -> Fe+3 + 2 H2O; 2.5""")  # "[H2O]" = 1.0 (actually 55.4 at RT)
 from chempy.kinetics.ode import get_odesys
 odesys, extra = get_odesys(rsys)
 from collections import defaultdict
 import numpy as np
 tout = sorted(np.concatenate((np.linspace(0, 23), np.logspace(-8, 1))))
 c0 = defaultdict(float, {'Fe+2': 0.05, 'H2O2': 0.1, 'H2O': 1.0, 'H+': 1e-7, 'OH-': 1e-7})
 result = odesys.integrate(tout, c0, atol=1e-12, rtol=1e-14)
 import matplotlib.pyplot as plt
 _ = plt.subplot(1, 2, 1)
 _ = result.plot(names=[k for k in rsys.substances if k != 'H2O'])
 _ = plt.legend(loc='best', prop={'size': 9}); _ = plt.xlabel('Time'); _ =
plt.ylabel('Concentration')
 _ = plt.subplot(1, 2, 2)
 _ = result.plot(names=[k for k in rsys.substances if k != 'H2O'], xscale='log', yscale='log')
 _ = plt.legend(loc='best', prop={'size': 9}); _ = plt.xlabel('Time'); _ =
plt.ylabel('Concentration')
 _ = plt.tight_layout()
 plt.show()
```

ChemPy - python https://riptutorial.com/zh-TW/python/topic/10625/chempy-----python

9: configparser

ConfigParserINI。 Python。

- =
-
- INI[]

`ConfigParser.ConfigParser().get。 eval`

Examples

config.ini

```
[DEFAULT]
debug = True
name = Test
password = password

[FILES]
path = /path/to/file
```

Python

```
[DEFAULT]
debug = True
name = Test
password = password

[FILES]
path = /path/to/file
```

。 configparser。 -

```
import configparser
config = configparser.ConfigParser()
config['settings']={'resolution':'320x240',
                    'color':'blue'}
with open('example.ini', 'w') as configfile:
    config.write(configfile)
```


```
import configparser
config = configparser.ConfigParser()
config['settings']={'resolution':'320x240',
                    'color':'blue'}
with open('example.ini', 'w') as configfile:
    config.write(configfile)
```


```
import configparser
config = configparser.ConfigParser()
```

```
config['settings']={'resolution':'320x240',
                    'color':'blue'}
with open('example.ini', 'w') as configfile:
    config.write(configfile)
```

configparser https://riptutorial.com/zh-TW/python/topic/9186/configparser

10: ctypes

ctypespython。

。

Examples

libcntohl。

libc.so

```
>>> from ctypes import *
>>> libc = cdll.LoadLibrary('libc.so.6')
>>> libc
<CDLL 'libc.so.6', handle baadf00d at 0xdeadbeef>
```


```
>>> from ctypes import *
>>> libc = cdll.LoadLibrary('libc.so.6')
>>> libc
<CDLL 'libc.so.6', handle baadf00d at 0xdeadbeef>
```

```
>>> from ctypes import *
>>> libc = cdll.LoadLibrary('libc.so.6')
>>> libc
<CDLL 'libc.so.6', handle baadf00d at 0xdeadbeef>
```


。

。 OSError。

OS

```
>>> cdll.LoadLibrary("foobar.so")
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "/usr/lib/python3.5/ctypes/__init__.py", line 425, in LoadLibrary
    return self._dlltype(name)
File "/usr/lib/python3.5/ctypes/__init__.py", line 347, in __init__
    self._handle = _dlopen(self._name, mode)
OSError: foobar.so: cannot open shared object file: No such file or directory
```

。

。

```
>>> cdll.LoadLibrary("foobar.so")
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in <module>
File "/usr/lib/python3.5/ctypes/__init__.py", line 425, in LoadLibrary
    return self._dlltype(name)
File "/usr/lib/python3.5/ctypes/__init__.py", line 347, in __init__
    self._handle = _dlopen(self._name, mode)
OSError: foobar.so: cannot open shared object file: No such file or directory
```

`.so`。 Linux`.dll`32python64。 。

`.so`。

`AttributeError`

```
>>> cdll.LoadLibrary("foobar.so")
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
File "/usr/lib/python3.5/ctypes/__init__.py", line 425, in LoadLibrary
    return self._dlltype(name)
File "/usr/lib/python3.5/ctypes/__init__.py", line 347, in __init__
    self._handle = _dlopen(self._name, mode)
OSError: foobar.so: cannot open shared object file: No such file or directory
```

ctypes

int

```
>>> obj = ctypes.c_int(12)
>>> obj
c_long(12)
```

`obj`12。

```
>>> obj = ctypes.c_int(12)
>>> obj
c_long(12)
```

`obj`

```
>>> obj = ctypes.c_int(12)
>>> obj
c_long(12)
```

ctypes

C。

```
>>> c_int * 16
<class '__main__.c_long_Array_16'>
```

16`int`。

```
>>> c_int * 16
<class '__main__.c_long_Array_16'>
```

`arr`015。

```
>>> c_int * 16
<class '__main__.c_long_Array_16'>
```

`ctypes`

```
>>> c_int * 16
<class '__main__.c_long_Array_16'>
```

ctypes

C。 `ctypes`python。

```
>>> def max(x, y):
        return x if x >= y else y
```

。 typeint。

```
>>> def max(x, y):
        return x if x >= y else y
```

`c_int` `c_int`。

```
>>> def max(x, y):
        return x if x >= y else y
```

`ctypeslibc.ntohl` 。

```
>>> def max(x, y):
        return x if x >= y else y
```

`libclfind`。

。 。

```
>>> compar_proto = CFUNCTYPE(c_int, POINTER(c_int), POINTER(c_int))
>>> lfind_proto = CFUNCTYPE(c_void_p, c_void_p, c_void_p, POINTER(c_uint), c_uint,
compar_proto)
```

```
>>> compar_proto = CFUNCTYPE(c_int, POINTER(c_int), POINTER(c_int))
>>> lfind_proto = CFUNCTYPE(c_void_p, c_void_p, c_void_p, POINTER(c_uint), c_uint,
compar_proto)
```

```
>>> compar_proto = CFUNCTYPE(c_int, POINTER(c_int), POINTER(c_int))
```

```
>>> lfind_proto = CFUNCTYPE(c_void_p, c_void_p, c_void_p, POINTER(c_uint), c_uint,
compar_proto)
```

xyPOINTER(c_int) ◦

```
>>> compar_proto = CFUNCTYPE(c_int, POINTER(c_int), POINTER(c_int))
>>> lfind_proto = CFUNCTYPE(c_void_p, c_void_p, c_void_p, POINTER(c_uint), c_uint,
compar_proto)
```

ptrvoid◦ arrkey None ◦

```
>>> compar_proto = CFUNCTYPE(c_int, POINTER(c_int), POINTER(c_int))
>>> lfind_proto = CFUNCTYPE(c_void_p, c_void_p, c_void_p, POINTER(c_uint), c_uint,
compar_proto)
```

ptrarr

```
>>> compar_proto = CFUNCTYPE(c_int, POINTER(c_int), POINTER(c_int))
>>> lfind_proto = CFUNCTYPE(c_void_p, c_void_p, c_void_p, POINTER(c_uint), c_uint,
compar_proto)
```

ctypes https://riptutorial.com/zh-TW/python/topic/9050/ctypes

11: Deque

- dq = deque
- dq = dequeiterabledeque
- dq.appendobject
- dq.appendleftobject
- dq.pop - > object
- dq.popleft - > object
- dq.extenditerabledeque
- dq.extendleftiterable

iterable	。
maxlen	。

deque“ ”。

pop appendextendleft。 dequeO1。

Examples

popleftappendleft

```
from collections import deque

d = deque([1, 2, 3])
p = d.popleft()         # p = 1, d = deque([2, 3])
d.appendleft(5)         # d = deque([5, 2, 3])
```

deque

maxlen

```
from collections import deque
d = deque(maxlen=3)  # only holds 3 items
d.append(1)  # deque([1])
d.append(2)  # deque([1, 2])
d.append(3)  # deque([1, 2, 3])
d.append(4)  # deque([2, 3, 4]) (1 is removed because its maxlen is 3)
```

deque

```
dl = deque()  # deque([]) creating empty deque
```

deque

```
dl = deque()  # deque([]) creating empty deque
```

deque

```
dl = deque()  # deque([]) creating empty deque
```

deque

```
dl = deque()  # deque([]) creating empty deque
```

deque

```
dl = deque()  # deque([]) creating empty deque
```

```
dl = deque()  # deque([]) creating empty deque
```

`.pop()`

```
dl = deque()  # deque([]) creating empty deque
```

`.popleft()`

```
dl = deque()  # deque([]) creating empty deque
```

```
dl = deque()  # deque([]) creating empty deque
```

deque

```
dl = deque()  # deque([]) creating empty deque
```

Deque Python。 `queue.Queue`。 Queue 。

```
from collections import deque

def bfs(graph, root):
    distances = {}
    distances[root] = 0
    q = deque([root])
    while q:
        # The oldest seen (but not yet visited) node will be the left most one.
        current = q.popleft()
        for neighbor in graph[current]:
            if neighbor not in distances:
                distances[neighbor] = distances[current] + 1
                # When we see a new node, we add it to the right side of the queue.
                q.append(neighbor)
    return distances
```

```
from collections import deque
```

```
def bfs(graph, root):
    distances = {}
    distances[root] = 0
    q = deque([root])
    while q:
        # The oldest seen (but not yet visited) node will be the left most one.
        current = q.popleft()
        for neighbor in graph[current]:
            if neighbor not in distances:
                distances[neighbor] = distances[current] + 1
                # When we see a new node, we add it to the right side of the queue.
                q.append(neighbor)
    return distances
```

```
from collections import deque

def bfs(graph, root):
    distances = {}
    distances[root] = 0
    q = deque([root])
    while q:
        # The oldest seen (but not yet visited) node will be the left most one.
        current = q.popleft()
        for neighbor in graph[current]:
            if neighbor not in distances:
                distances[neighbor] = distances[current] + 1
                # When we see a new node, we add it to the right side of the queue.
                q.append(neighbor)
    return distances
```

Deque https://riptutorial.com/zh-TW/python/topic/1976/deque

12: dis

Examples

dis

```
EXTENDED_ARG = 145 # All opcodes greater than this have 2 operands
HAVE_ARGUMENT = 90 # All opcodes greater than this have at least 1 operands

cmp_op = ('<', '<=', '==', '!=', '>', '>=', 'in', 'not in', 'is', 'is ...
       # A list of comparator id's. The indecies are used as operands in some opcodes

# All opcodes in these lists have the respective types as there operands
hascompare = [107]
hasconst = [100]
hasfree = [135, 136, 137]
hasjabs = [111, 112, 113, 114, 115, 119]
hasjrel = [93, 110, 120, 121, 122, 143]
haslocal = [124, 125, 126]
hasname = [90, 91, 95, 96, 97, 98, 101, 106, 108, 109, 116]

# A map of opcodes to ids
opmap = {'BINARY_ADD': 23, 'BINARY_AND': 64, 'BINARY_DIVIDE': 21, 'BIN...
# A map of ids to opcodes
opname = ['STOP_CODE', 'POP_TOP', 'ROT_TWO', 'ROT_THREE', 'DUP_TOP', '...
```

Python

Python。 Python*Python* 。 `dis`Python。

```
>>> def hello():
...     print "Hello, World"
...
>>> dis.dis(hello)
  2           0 LOAD_CONST               1 ('Hello, World')
              3 PRINT_ITEM
              4 PRINT_NEWLINE
              5 LOAD_CONST               0 (None)
              8 RETURN_VALUE
```

Python。

Python。 Python。

Python`.pyc`Python。

```
python -m compileall <file>.py
```


```
python -m compileall <file>.py
```

Python`dis`。 。

dis https://riptutorial.com/zh-TW/python/topic/1763/dis

13: Django

DjangoPython Web。 Web。 。

Examples

DjangoHello World

django`Hello World`。

django。

python -c“import django”
- >django。

django。
django-admin startproject HelloWorld

HelloWorld。

| --helloworld
| | - **init** .py
| | --settings.py
| | --urls.py
| | --wsgi.py
| --manage.py

django

PythonWebWeb。 HTML。 views.py。

hello world。 views.py

```
from django.http import HttpResponse

define helloWorld(request):
    return HttpResponse("Hello World!! Django Welcomes You.")
```

。

- django.httpHttpResponse。
- helloWorld。 。 HttpRequestrequest。

  ;Django。 helloWorld。
-

HttpResponse。 HttpResponse。

django

URL
URLURLconf;

django。

- DjangoURLconf。
- DjangoPythonurlpatterns。 django.conf.urls.urlPython。
- DjangoURLURL。
- DjangoPython。

URLconf

```
from django.http import HttpResponse

define helloWorld(request):
    return HttpResponse("Hello World!! Django Welcomes You.")
```

django Urls

HelloWorld。
python manage.py runserver

127.0.0.1:8000

127.0.0.1:8000/helloworld/。 “Hello World !! Django”。

Django https://riptutorial.com/zh-TW/python/topic/8994/django

14: Functools

Examples

partial。 。

```
>>> from functools import partial
>>> unhex = partial(int, base=16)
>>> unhex.__doc__ = 'Convert base16 string to int'
>>> unhex('ca11ab1e')
3390155550
```

partial() 。

```
>>> from functools import partial
>>> unhex = partial(int, base=16)
>>> unhex.__doc__ = 'Convert base16 string to int'
>>> unhex('ca11ab1e')
3390155550
```

g f a, b, c, x a, b, c,。 fg g(2)f 。

partial;。 partial。

total_ordering

__eq()__ __lt__() __le__() __gt__() __le__() __gt__()__ge__() 。

total_ordering__eq__()__lt__() __le__() __gt__()__ge__() 。

```
@total_ordering
class Employee:

    ...

    def __eq__(self, other):
        return ((self.surname, self.name) == (other.surname, other.name))

    def __lt__(self, other):
        return ((self.surname, self.name) < (other.surname, other.name))
```

。 __lt__()__eq()____gt__() not __lt__() and not __eq()__ 。

total_orderingPython 2.7。

Python 3.x reducefunctools。

```
from functools import reduce
def factorial(n):
```

```
    return reduce(lambda a, b: (a*b), range(1, n+1))
```

lru_cache

`@lru_cache`。 。

```
@lru_cache(maxsize=None)  # Boundless cache
def fibonacci(n):
    if n < 2:
        return n
    return fibonacci(n-1) + fibonacci(n-2)

>>> fibonacci(15)
```

`fibonacci(3)` `fibonacci` LRU `fibonacci(3)` 230。 `@lru_cache`。

`@lru_cache`

- `maxsize` 。 `maxsize` LRU。
- `typed` 3.3 `3.03`

```
@lru_cache(maxsize=None)  # Boundless cache
def fibonacci(n):
    if n < 2:
        return n
    return fibonacci(n-1) + fibonacci(n-2)

>>> fibonacci(15)
```

`@lru_cache`。

`@lru_cache` Python 。 `@lru_cache` 3.2。

cmp_to_key

Python。 。

-1,0+1。 。

`functools.cmp_to_key`

```
>>> import functools
>>> import locale
>>> sorted(["A", "S", "F", "D"], key=functools.cmp_to_key(locale.strcoll))
['A', 'D', 'F', 'S']
```

Python。

Functools https://riptutorial.com/zh-TW/python/topic/2492/functools

15: hashlib

hashlib。 FIPSSHA1SHA224SHA256SHA384SHA512。

Examples

MD5

。 FIPSSHA1SHA224SHA256SHA384SHA512FIPS 180-2RSAMD5Internet RFC 1321。

。 。 `sha1()`SHA1。

```
hash.sha1()
```

`md5()` `sha1()` `sha224()` `sha256()` `sha384()``sha512()` 。

`update()`。 `digest()``hexdigest()`digest() 。

```
hash.sha1()
```

arg。 m.updatea; m.updatebm.updatea + b。

```
hash.sha1()
```

update。 digest_sizeASCII。

```
hash.sha1()
```

digest。 。

```
hash.sha1()
```

```
hash.sha1()
```

OpenSSL

`new()`OpenSSL。 `new()`。

`new()`OpenSSL

```
>>> h = hashlib.new('ripemd160')
>>> h.update("Nobody inspects the spammish repetition")
>>> h.hexdigest()
'cc4a5ce1b3df48aec5d22d1f16b894a0b894eccc'
```

hashlib https://riptutorial.com/zh-TW/python/topic/8980/hashlib

16: Heapq

Examples

heapqnlargest

```
import heapq


numbers = [1, 4, 2, 100, 20, 50, 32, 200, 150, 8]
print(heapq.nlargest(4, numbers))  # [200, 150, 100, 50]
```

nsmallest

```
import heapq


numbers = [1, 4, 2, 100, 20, 50, 32, 200, 150, 8]
print(heapq.nlargest(4, numbers))  # [200, 150, 100, 50]
```

nlargestnsmallest。 agepeople

```
import heapq


numbers = [1, 4, 2, 100, 20, 50, 32, 200, 150, 8]
print(heapq.nlargest(4, numbers))  # [200, 150, 100, 50]
```

heap heap[0]

```
import heapq


numbers = [10, 4, 2, 100, 20, 50, 32, 200, 150, 8]

heapq.heapify(numbers)
print(numbers)
# Output: [2, 4, 10, 100, 8, 50, 32, 200, 150, 20]

heapq.heappop(numbers)  # 2
print(numbers)
# Output: [4, 8, 10, 100, 20, 50, 32, 200, 150]

heapq.heappop(numbers)  # 4
print(numbers)
# Output:  [8, 20, 10, 100, 150, 50, 32, 200]
```

Heapq https://riptutorial.com/zh-TW/python/topic/7489/heapq

17: HTML

Examples

BeautifulSoup

HTML

```
<div>
    <label>Name:</label>
    John Smith
</div>
```

`label`“John Smith”。

`label.next_sibling`

```
<div>
    <label>Name:</label>
    John Smith
</div>
```

`John Smith` 。

BeautifulSoupCSS

BeautifulSoup CSS 。 `select()` `select_one()` 。

```
from bs4 import BeautifulSoup

data = """
<ul>
    <li class="item">item1</li>
    <li class="item">item2</li>
    <li class="item">item3</li>
</ul>
"""

soup = BeautifulSoup(data, "html.parser")

for item in soup.select("li.item"):
    print(item.get_text())
```

```
from bs4 import BeautifulSoup

data = """
<ul>
    <li class="item">item1</li>
    <li class="item">item2</li>
    <li class="item">item3</li>
</ul>
```

```
"""

soup = BeautifulSoup(data, "html.parser")

for item in soup.select("li.item"):
    print(item.get_text())
```

PyQuery

pyqueryjquerypython。 css。

```
from pyquery import PyQuery

html = """
<h1>Sales</h1>
<table id="table">
<tr>
    <td>Lorem</td>
    <td>46</td>
</tr>
<tr>
    <td>Ipsum</td>
    <td>12</td>
</tr>
<tr>
    <td>Dolor</td>
    <td>27</td>
</tr>
<tr>
    <td>Sit</td>
    <td>90</td>
</tr>
</table>
"""

doc = PyQuery(html)

title = doc('h1').text()

print title

table_data = []

rows = doc('#table > tr')
for row in rows:
    name = PyQuery(row).find('td').eq(0).text()
    value = PyQuery(row).find('td').eq(1).text()

    print "%s\t  %s" % (name, value)
```

HTML https://riptutorial.com/zh-TW/python/topic/1384/html

18: ijson

ijsonPythonJSON。Python JSON。C。

Examples

```
import ijson

def load_json(filename):
    with open(filename, 'r') as fd:
        parser = ijson.parse(fd)
        ret = {'builders': {}}
        for prefix, event, value in parser:
            if (prefix, event) == ('builders', 'map_key'):
                buildername = value
                ret['builders'][buildername] = {}
            elif prefix.endswith('.shortname'):
                ret['builders'][buildername]['shortname'] = value

        return ret

if __name__ == "__main__":
    load_json('allthethings.json')
```

JSON

ijson https://riptutorial.com/zh-TW/python/topic/8342/ijson

19: IterablesIterators

Examples

Iterator vs Iterable vs Generator

iterable。 `__iter__`。 `__getitem__`。 - `IndexError`。

Python`str``__getitem__` iterable。

Iterator`next(*object*)`。 `__next__`。 `StopIteration`。

Iterable`__iter__`、`__next__`。

```
class MyIterable:

    def __iter__(self):

         return self

    def __next__(self):
         #code

#Classic iterable object in older versions of python, __getitem__ is still supported...
class MySequence:

    def __getitem__(self, index):

         if (condition):
             raise IndexError
         return (item)

 #Can produce a plain `iterator` instance by using iter(MySequence())
```

`collections`。

Python 2.x 2.3

```
class MyIterable:

    def __iter__(self):

         return self

    def __next__(self):
         #code

#Classic iterable object in older versions of python, __getitem__ is still supported...
class MySequence:

    def __getitem__(self, index):

         if (condition):
```

```
            raise IndexError
        return (item)

 #Can produce a plain `iterator` instance by using iter(MySequence())
```

Python 3.x 3.0

```
class MyIterable:

    def __iter__(self):

         return self

    def __next__(self):
         #code

#Classic iterable object in older versions of python, __getitem__ is still supported...
class MySequence:

    def __getitem__(self, index):

         if (condition):
             raise IndexError
         return (item)

 #Can produce a plain `iterator` instance by using iter(MySequence())
```

Python 2Python 3

Python 2.x 2.3

```
class MyIterable:

    def __iter__(self):

         return self

    def __next__(self):
         #code

#Classic iterable object in older versions of python, __getitem__ is still supported...
class MySequence:

    def __getitem__(self, index):

         if (condition):
             raise IndexError
         return (item)

 #Can produce a plain `iterator` instance by using iter(MySequence())
```

```
class MyIterable:

    def __iter__(self):

         return self

    def __next__(self):
```

```
        #code

#Classic iterable object in older versions of python, __getitem__ is still supported...
class MySequence:

    def __getitem__(self, index):

        if (condition):
            raise IndexError
        return (item)

 #Can produce a plain `iterator` instance by using iter(MySequence())
```

。 。

Iterable。 Python

```
[1, 2, 3]     # list, iterate over items
(1, 2, 3)     # tuple
{1, 2, 3}     # set
{1: 2, 3: 4}  # dict, iterate over keys
```


```
[1, 2, 3]     # list, iterate over items
(1, 2, 3)     # tuple
{1, 2, 3}     # set
{1: 2, 3: 4}  # dict, iterate over keys
```


```
s = {1, 2, 3}

# get every element in s
for a in s:
    print a  # prints 1, then 2, then 3

# copy into list
l1 = list(s)  # l1 = [1, 2, 3]

# use list comprehension
l2 = [a * 2 for a in s if a > 2]  # l2 = [6]
```

iterable

```
a, = iterable

def foo():
    yield 1

a, = foo()  # a = 1

nums = [1, 2, 3]
a, = nums  # ValueError: too many values to unpack
```

`iter()` `next()``StopIteration`

```
s = {1, 2}   # or list or generator or even iterator
```

```
i = iter(s)  # get iterator
a = next(i)  # a = 1
b = next(i)  # b = 2
c = next(i)  # raises StopIteration
```

```
def gen():
    yield 1

iterable = gen()
for a in iterable:
    print a

# What was the first item of iterable? No way to get it now.
# Only to get a new iterator
gen()
```

IterablesIterators https://riptutorial.com/zh-TW/python/topic/2343/iterablesiterators

20: Itertools

- `import itertools`

Examples

```
lst = [("a", 5, 6), ("b", 2, 4), ("a", 2, 5), ("c", 2, 6)]
```

```
lst = [("a", 5, 6), ("b", 2, 4), ("a", 2, 5), ("c", 2, 6)]
```

◦ groupby◦

```
lst = [("a", 5, 6), ("b", 2, 4), ("a", 2, 5), ("c", 2, 6)]
```

groupby◦ ◦ 525

```
lst = [("a", 5, 6), ("b", 2, 4), ("a", 2, 5), ("c", 2, 6)]
```

```
lst = [("a", 5, 6), ("b", 2, 4), ("a", 2, 5), ("c", 2, 6)]
```

Itertools“islice”

```
results = fetch_paged_results()  # returns a generator
limit = 20  # Only want the first 20 results
for data in itertools.islice(results, limit):
    print(data)
```

```
results = fetch_paged_results()  # returns a generator
limit = 20  # Only want the first 20 results
for data in itertools.islice(results, limit):
    print(data)
```

```
results = fetch_paged_results()  # returns a generator
limit = 20  # Only want the first 20 results
for data in itertools.islice(results, limit):
    print(data)
```

```
results = fetch_paged_results()  # returns a generator
limit = 20  # Only want the first 20 results
for data in itertools.islice(results, limit):
    print(data)
```

`start stopstep`

```
results = fetch_paged_results()  # returns a generator
limit = 20  # Only want the first 20 results
for data in itertools.islice(results, limit):
```

```
    print(data)
```

itertools.product

。

```
for x, y in itertools.product(xrange(10), xrange(10)):
    print x, y
```

```
for x, y in itertools.product(xrange(10), xrange(10)):
    print x, y
```

python*itertools.product。

```
for x, y in itertools.product(xrange(10), xrange(10)):
    print x, y
```

。

```
for x, y in itertools.product(xrange(10), xrange(10)):
    print x, y
```

itertools.count

。 ...

```
for number in itertools.count():
    if number > 20:
        break
    print(number)
```

```
for number in itertools.count():
    if number > 20:
        break
    print(number)
```

---

count() startstep

```
for number in itertools.count():
    if number > 20:
        break
    print(number)
```

```
for number in itertools.count():
    if number > 20:
        break
    print(number)
```

itertools.takewhile

itertools.takewhile`False`。

```
def is_even(x):
    return x % 2 == 0


lst = [0, 2, 4, 12, 18, 13, 14, 22, 23, 44]
result = list(itertools.takewhile(is_even, lst))

print(result)
```

`[0, 2, 4, 12, 18]`。

`is_even13`。 `takewhileFalse`。

`takewhile`。

```
def is_even(x):
    return x % 2 == 0


lst = [0, 2, 4, 12, 18, 13, 14, 22, 23, 44]
result = list(itertools.takewhile(is_even, lst))

print(result)
```

`takewhiledropwhile`。

`result = list(itertools.takewhile(is_even, lst)) + list(itertools.dropwhile(is_even, lst))`

itertools.dropwhile

itertools.dropwhile`False`。

```
def is_even(x):
    return x % 2 == 0


lst = [0, 2, 4, 12, 18, 13, 14, 22, 23, 44]
result = list(itertools.dropwhile(is_even, lst))

print(result)
```

`[13, 14, 22, 23, 44]`。

`takewhiledropwhile`。

`is_even13`。 。

`dropwhile`。

```
def is_even(x):
    return x % 2 == 0


lst = [0, 2, 4, 12, 18, 13, 14, 22, 23, 44]
result = list(itertools.dropwhile(is_even, lst))

print(result)
```

`takewhiledropwhile`。

```
result = list(itertools.takewhile(is_even, lst)) + list(itertools.dropwhile(is_even, lst))
```

`zip()` `itertools.zip_longest` `itertools.zip_longest`。

```
from itertools import zip_longest
a = [i for i in range(5)] # Length is 5
b = ['a', 'b', 'c', 'd', 'e', 'f', 'g'] # Length is 7
for i in zip_longest(a, b):
    x, y = i  # Note that zip longest returns the values as a tuple
    print(x, y)
```

`fillvalue''`

```
from itertools import zip_longest
a = [i for i in range(5)] # Length is 5
b = ['a', 'b', 'c', 'd', 'e', 'f', 'g'] # Length is 7
for i in zip_longest(a, b):
    x, y = i  # Note that zip longest returns the values as a tuple
    print(x, y)
```

Python 2.62.7`itertools.izip_longest`。

Itertools

`itertools.combinations`*k*。

k-。

```
a = [1,2,3,4,5]
b = list(itertools.combinations(a, 2))
print b
```

`[(1, 2), (1, 3), (1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5)]`

`a`

3

```
a = [1,2,3,4,5]
```

```
b = list(itertools.combinations(a, 2))
print b
```

```
a = [1,2,3,4,5]
b = list(itertools.combinations(a, 2))
print b
```

itertools.chain。

```
from itertools import chain
a = (x for x in ['1', '2', '3', '4'])
b = (x for x in ['x', 'y', 'z'])
' '.join(chain(a, b))
```


```
from itertools import chain
a = (x for x in ['1', '2', '3', '4'])
b = (x for x in ['x', 'y', 'z'])
' '.join(chain(a, b))
```


classmethod `chain.from_iterable` 。


```
from itertools import chain
a = (x for x in ['1', '2', '3', '4'])
b = (x for x in ['x', 'y', 'z'])
' '.join(chain(a, b))
```


chainchain.from_iterable。

itertools.repeat

n

```
>>> import itertools
>>> for i in itertools.repeat('over-and-over', 3):
...     print(i)
over-and-over
over-and-over
over-and-over
```

Python 3.x 3.2

accumulate。

```
>>> import itertools as it
>>> import operator

>>> list(it.accumulate([1,2,3,4,5]))
[1, 3, 6, 10, 15]

>>> list(it.accumulate([1,2,3,4,5], func=operator.mul))
[1, 2, 6, 24, 120]
```

cycle。

```
>>> import itertools as it
>>> it.cycle('ABCD')
A B C D A B C D A B C D ...
```

。


```
>>> import itertools as it
>>> it.cycle('ABCD')
A B C D A B C D A B C D ...
```


itertools.permutations

itertools.permutationsr。

```
a = [1,2,3]
list(itertools.permutations(a))
# [(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)]

list(itertools.permutations(a, 2))
[(1, 2), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2)]
```

aset


```
a = [1,2,3]
list(itertools.permutations(a))
# [(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)]

list(itertools.permutations(a, 2))
[(1, 2), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2)]
```


Itertools https://riptutorial.com/zh-TW/python/topic/1564/itertools

21: JSON

。

json

JSON	
int	INT

jsonNaN Infinity-InfinityJSON。

	JSON
intfloatint / float-derived Enums	

NaN Infinity-Infinity allow_nan=False。 ValueError 。

。 functools.partial。

```
# my_json module

import json
from functools import partial

def serialise_object(obj):
    # Do something to produce json-serialisable data
    return dict_obj

dump = partial(json.dump, default=serialise_object)
dumps = partial(json.dumps, default=serialise_object)
```

jsonobject_hookparse_float。 python 。

```
# my_json module

import json
```

```
from functools import partial

def serialise_object(obj):
    # Do something to produce json-serialisable data
    return dict_obj

dump = partial(json.dump, default=serialise_object)
dumps = partial(json.dumps, default=serialise_object)
```

`json`/`json.JSONEncoder``json.JSONDecoder`◦ ◦ `cls`◦ `functools.partial`cls

```
# my_json module

import json
from functools import partial

def serialise_object(obj):
    # Do something to produce json-serialisable data
    return dict_obj

dump = partial(json.dump, default=serialise_object)
dumps = partial(json.dumps, default=serialise_object)
```

Examples

Python dictJSON

```
import json
d = {
    'foo': 'bar',
    'alice': 1,
    'wonderland': [1, 2, 3]
}
json.dumps(d)
```

```
import json
d = {
    'foo': 'bar',
    'alice': 1,
    'wonderland': [1, 2, 3]
}
json.dumps(d)
```

JSONPython dict

```
import json
s = '{"wonderland": [1, 2, 3], "foo": "bar", "alice": 1}'
json.loads(s)
```

```
import json
s = '{"wonderland": [1, 2, 3], "foo": "bar", "alice": 1}'
json.loads(s)
```

dJSONfilenamefilename。

```
import json

d = {
    'foo': 'bar',
    'alice': 1,
    'wonderland': [1, 2, 3]
}

with open(filename, 'w') as f:
    json.dump(d, f)
```

JSONfilenamefilename。

```
import json

with open(filename, 'r') as f:
    d = json.load(f)
```

`load` vs`load``dump` vs`stacks`

jsonunicode。 s。 StringIO。

```
import json

data = {u"foo": u"bar", u"baz": []}
json_string = json.dumps(data)
# u'{"foo": "bar", "baz": []}'
json.loads(json_string)
# {u"foo": u"bar", u"baz": []}
```


```
import json

data = {u"foo": u"bar", u"baz": []}
json_string = json.dumps(data)
# u'{"foo": "bar", "baz": []}'
json.loads(json_string)
# {u"foo": u"bar", u"baz": []}
```


json。 。 0


```
import json

data = {u"foo": u"bar", u"baz": []}
json_string = json.dumps(data)
# u'{"foo": "bar", "baz": []}'
json.loads(json_string)
# {u"foo": u"bar", u"baz": []}
```


jsonjsonpysparkjson-per-line

```
import json
```

```
data = {u"foo": u"bar", u"baz": []}
json_string = json.dumps(data)
# u'{"foo": "bar", "baz": []}'
json.loads(json_string)
# {u"foo": u"bar", u"baz": []}
```

`json.tool`JSON

JSON“foo.json”

```
{"foo": {"bar": {"baz": 1}}}
```

```
{"foo": {"bar": {"baz": 1}}}
```

STDOUTBash

```
{"foo": {"bar": {"baz": 1}}}
```

JSON

```
>>> data = {"cats": [{"name": "Tubbs", "color": "white"}, {"name": "Pepper", "color":
"black"}]}
```

JSON

```
>>> data = {"cats": [{"name": "Tubbs", "color": "white"}, {"name": "Pepper", "color":
"black"}]}
```

`indent`

```
>>> data = {"cats": [{"name": "Tubbs", "color": "white"}, {"name": "Pepper", "color":
"black"}]}
```

。

```
>>> data = {"cats": [{"name": "Tubbs", "color": "white"}, {"name": "Pepper", "color":
"black"}]}
```

`', '': '`

```
>>> data = {"cats": [{"name": "Tubbs", "color": "white"}, {"name": "Pepper", "color":
"black"}]}
```

JSON

```
import json
from datetime import datetime
data = {'datetime': datetime(2016, 9, 26, 4, 44, 0)}
print(json.dumps(data))
```

```
TypeError: datetime.datetime(2016, 9, 26, 4, 44) is not JSON serializable
```

datetime

```
import json
from datetime import datetime
data = {'datetime': datetime(2016, 9, 26, 4, 44, 0)}
print(json.dumps(data))
```

`json.dumps`

```
import json
from datetime import datetime
data = {'datetime': datetime(2016, 9, 26, 4, 44, 0)}
print(json.dumps(data))
```

JSON https://riptutorial.com/zh-TW/python/topic/272/json

22: kivy - NUIPython

NUINUI。

KivyPython。 。

Examples

kivy

1. **app**
2. 。
3. **run** 。

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

app 。 your_installtion_directory / kivy / app.py

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

ux**Label** 。 uxyour_installation_directory / kivy / uix /。

```
from kivy.app import App
from kivy.uix.label import Label
```

```
class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

- app。

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

app。 。

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

- Label**Hello world** 。

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

python。

```
from kivy.app import App
from kivy.uix.label import Label

class Test(App):
    def build(self):
        return Label(text='Hello world')

if __name__ == '__main__':
    Test().run()
```

Test。 apprun。

。

kivy - NUIPython https://riptutorial.com/zh-TW/python/topic/10743/kivy----nuipython

23: Neo4jCypherPy2Neo

Examples

```
from py2neo import authenticate, Graph, Node, Relationship
authenticate("localhost:7474", "neo4j", "<pass>")
graph = Graph()
```

Neo4jlocalhost7474。

`graph`pythonneo4j。 `__init__`。

Neo4j

```
results = News.objects.todays_news()
for r in results:
    article = graph.merge_one("NewsArticle", "news_id", r)
    article.properties["title"] = results[r]['news_title']
    article.properties["timestamp"] = results[r]['news_timestamp']
    article.push()
    [...]
```

`graph.merge_one`。

`timestamp`neo4jdate。 '05 -06-1989'

`article.push()`neo4j。 。

Neo4j

```
results = News.objects.todays_news()
for r in results:
    article = graph.merge_one("NewsArticle", "news_id", r)
    if 'LOCATION' in results[r].keys():
        for loc in results[r]['LOCATION']:
            loc = graph.merge_one("Location", "name", loc)
            try:
                rel = graph.create_unique(Relationship(article, "about_place", loc))
            except Exception, e:
                print e
```

`create_unique`。 。 。

1

```
def get_autocomplete(text):
    query = """
    start n = node(*) where n.name =~ '(?i)%s.*' return n.name,labels(n) limit 10;
    """
```

```
    query = query % (text)
    obj = []
    for res in graph.cypher.execute(query):
        # print res[0],res[1]
        obj.append({'name':res[0],'entity_type':res[1]})
    return res
```

cypher`textname`。

2

```
def search_news_by_entity(location,timestamp):
    query = """
    MATCH (n)-[]->(l)
    where l.name='%s' and n.timestamp='%s'
    RETURN n.news_id limit 10
    """

    query = query % (location,timestamp)

    news_ids = []
    for res in graph.cypher.execute(query):
        news_ids.append(str(res[0]))

    return news_ids
```

`(l)(n)` 。

Cypher

```
MATCH (n)-[]->(l)
where l.name='Donald Trump'
RETURN n.date,count(*) order by n.date
```

/5。

```
MATCH (n)-[]->(l)
where l.name='Donald Trump'
RETURN n.date,count(*) order by n.date
```

Neo4jCypherPy2Neo https://riptutorial.com/zh-TW/python/topic/5841/neo4jcypherpy2neo

24: os.path

◦ ◦ Unicode◦

- os.path.joina* p
- os.path.basenamep
- os.path.dirnamep
- os.path.splitp
- os.path.splitextp

Examples

pythonos

```
import os
os.path.join('a', 'b', 'c')
```

os.path◦

Windows

```
import os
os.path.join('a', 'b', 'c')
```

Unix

```
import os
os.path.join('a', 'b', 'c')
```

os.path.abspath

```
>>> os.getcwd()
'/Users/csaftoiu/tmp'
>>> os.path.abspath('foo')
'/Users/csaftoiu/tmp/foo'
>>> os.path.abspath('../foo')
'/Users/csaftoiu/foo'
>>> os.path.abspath('/foo')
'/foo'
```

```
>>> p = os.path.join(os.getcwd(), 'foo.txt')
>>> p
'/Users/csaftoiu/tmp/foo.txt'
>>> os.path.dirname(p)
'/Users/csaftoiu/tmp'
>>> os.path.basename(p)
'foo.txt'
>>> os.path.split(os.getcwd())
('/Users/csaftoiu/tmp', 'foo.txt')
```

```
>>> os.path.splitext(os.path.basename(p))
('foo', '.txt')
```

```
os.path.abspath(os.path.join(PATH_TO_GET_THE_PARENT, os.pardir))
```

◦

```
path = '/home/john/temp'
os.path.exists(path)
#this returns false if path doesn't exist or if the path is a broken symbolic link
```

◦

```
dirname = '/home/john/python'
os.path.isdir(dirname)
```

```
dirname = '/home/john/python'
os.path.isdir(dirname)
```

```
dirname = '/home/john/python'
os.path.isdir(dirname)
```

```
dirname = '/home/john/python'
os.path.isdir(dirname)
```

os.path https://riptutorial.com/zh-TW/python/topic/1380/os-path

25: os

◦

-

	◦ os.path.sep◦
	0700

Examples

```
os.mkdir('newdir')
```

mode

```
os.mkdir('newdir')
```

os.getcwd()

```
print(os.getcwd())
```

os◦

```
os.name
```

Python 3

- posix
- nt
- ce
- java

sys.platform

path

```
os.rmdir(path)
```

os.remove()◦ OSError

POSIX

◦ os.readlink

```
print(os.readlink(path_to_symlink))
```

```
os.chmod(path, mode)
```

mode。

makedirs -

```
└── dir1
    ├── subdir1
    └── subdir2
```

dir2subdir1subdir2。

```
└── dir1
    ├── subdir1
    └── subdir2
```

```
└── dir1
    ├── subdir1
    └── subdir2
```

dir2subdir1。

os.mkdir dir2。

```
└── dir1
    ├── subdir1
    └── subdir2
```

os.makedirs。

```
└── dir1
    ├── subdir1
    └── subdir2
```

。

```
└── dir1
    ├── subdir1
    └── subdir2
```

os https://riptutorial.com/zh-TW/python/topic/4127/os

26: Pandas Transform

Examples

nm

```
orders_df = pd.DataFrame()
orders_df['customer_id'] = [1,1,1,1,1,2,2,3,3,3,3,3]
orders_df['order_id'] = [1,1,1,2,2,3,3,4,5,6,6,6]
orders_df['item'] = ['apples', 'chocolate', 'chocolate', 'coffee', 'coffee', 'apples',
                     'bananas', 'coffee', 'milkshake', 'chocolate', 'strawberry',
'strawberry']

# And this is how the dataframe looks like:
print(orders_df)
#     customer_id  order_id        item
# 0             1         1      apples
# 1             1         1   chocolate
# 2             1         1   chocolate
# 3             1         2      coffee
# 4             1         2      coffee
# 5             2         3      apples
# 6             2         3     bananas
# 7             3         4      coffee
# 8             3         5   milkshake
# 9             3         6   chocolate
# 10            3         6  strawberry
# 11            3         6  strawberry
```

。

。

pandas `transform`

```
orders_df = pd.DataFrame()
orders_df['customer_id'] = [1,1,1,1,1,2,2,3,3,3,3,3]
orders_df['order_id'] = [1,1,1,2,2,3,3,4,5,6,6,6]
orders_df['item'] = ['apples', 'chocolate', 'chocolate', 'coffee', 'coffee', 'apples',
                     'bananas', 'coffee', 'milkshake', 'chocolate', 'strawberry',
'strawberry']

# And this is how the dataframe looks like:
print(orders_df)
#     customer_id  order_id        item
# 0             1         1      apples
# 1             1         1   chocolate
# 2             1         1   chocolate
# 3             1         2      coffee
# 4             1         2      coffee
# 5             2         3      apples
# 6             2         3     bananas
# 7             3         4      coffee
# 8             3         5   milkshake
# 9             3         6   chocolate
```

```
# 10            3         6  strawberry
# 11            3         6  strawberry
```

transform

-
-

```
# Create a dummy dataframe
orders_df = pd.DataFrame()
orders_df['customer_id'] = [1,1,1,1,1,2,2,3,3,3,3,3]
orders_df['order_id'] = [1,1,1,2,2,3,3,4,5,6,6,6]
orders_df['item'] = ['apples', 'chocolate', 'chocolate', 'coffee', 'coffee', 'apples',
                     'bananas', 'coffee', 'milkshake', 'chocolate', 'strawberry',
'strawberry']


# Let's try to see if the items were ordered more than once in each orders

# First, we define a fuction that will be applied per group
def multiple_items_per_order(_items):
    # Apply .duplicated, which will return True is the item occurs more than once.
    multiple_item_bool = _items.duplicated(keep=False)
    return(multiple_item_bool)

# Then, we transform each group according to the defined function
orders_df['item_duplicated_per_order'] = (                    # Put the results into a new
column
                        orders_df                             # Take the orders dataframe
                        .groupby(['order_id'])['item']        # Create a seperate group for
each order_id & select the item
                        .transform(multiple_items_per_order)) # Apply the defined function to
each group separately

# Inspecting the results ...
print(orders_df)
#     customer_id  order_id        item  item_duplicated_per_order
# 0             1         1      apples                      False
# 1             1         1   chocolate                       True
# 2             1         1   chocolate                       True
# 3             1         2      coffee                       True
# 4             1         2      coffee                       True
# 5             2         3      apples                      False
# 6             2         3     bananas                      False
# 7             3         4      coffee                      False
# 8             3         5   milkshake                      False
# 9             3         6   chocolate                      False
# 10            3         6  strawberry                       True
# 11            3         6  strawberry                       True
```

Pandas Transform https://riptutorial.com/zh-TW/python/topic/10947/pandas-transform-

27: Pickle

- pickle.dumpobjectfileprotocol#Tutize
- pickle.loadfile
- pickle.dumpsobjectprotocol
- pickle.loadsbuffer

Pickleable

。

- `None` `True` `False`
-
-
- `tuple` S `list` S `set` S`dict`
-
-
- ○ `__dict____getstate__()__getstate__()` 。

Python 。

pickle

。 Internet。

Examples

Pickle

`pickle`Python。 。 。

`dump()` `load()`。

```
import pickle

# An arbitrary collection of objects supported by pickle.
data = {
```

```
    'a': [1, 2.0, 3, 4+6j],
    'b': ("character string", b"byte string"),
    'c': {None, True, False}
}

with open('data.pickle', 'wb') as f:
    # Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(data, f, pickle.HIGHEST_PROTOCOL)
```

```
import pickle

# An arbitrary collection of objects supported by pickle.
data = {
    'a': [1, 2.0, 3, 4+6j],
    'b': ("character string", b"byte string"),
    'c': {None, True, False}
}

with open('data.pickle', 'wb') as f:
    # Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(data, f, pickle.HIGHEST_PROTOCOL)
```

---

picklebyte

dumpsloadsdumpload ◦

```
import pickle

# An arbitrary collection of objects supported by pickle.
data = {
    'a': [1, 2.0, 3, 4+6j],
    'b': ("character string", b"byte string"),
    'c': {None, True, False}
}

with open('data.pickle', 'wb') as f:
    # Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(data, f, pickle.HIGHEST_PROTOCOL)
```

◦ ◦

__getstate____getstate__ ◦ ◦

oposite__setstate__ __getstate__◦

```
class A(object):
    def __init__(self, important_data):
        self.important_data = important_data

        # Add data which cannot be pickled:
```

```
        self.func = lambda: 7

        # Add data which should never be pickled, because it expires quickly:
        self.is_up_to_date = False

    def __getstate__(self):
        return [self.important_data] # only this is needed

    def __setstate__(self, state):
        self.important_data = state[0]

        self.func = lambda: 7  # just some hard-coded unpicklable function

        self.is_up_to_date = False  # even if it was before pickling
```

```
class A(object):
    def __init__(self, important_data):
        self.important_data = important_data

        # Add data which cannot be pickled:
        self.func = lambda: 7

        # Add data which should never be pickled, because it expires quickly:
        self.is_up_to_date = False

    def __getstate__(self):
        return [self.important_data] # only this is needed

    def __setstate__(self, state):
        self.important_data = state[0]

        self.func = lambda: 7  # just some hard-coded unpicklable function

        self.is_up_to_date = False  # even if it was before pickling
```

`[self.important_data]` ◦ `__getstate____getstate____setstate__`oppoisite◦ `{'important_data': self.important_data}` ◦

`a2pickle.loadsA.__init__ A.__setstate____init__`◦

Pickle https://riptutorial.com/zh-TW/python/topic/2606/pickle

28: pipPyPI

pipPython Package IndexPython。

- pip <command> [options]<command>
  - PyPI
  - Zip
  - pybundles

`pip`。 LinuxpythonC。 WindowsVisual Studio / Visual C ++。 。

Python	Visual Studio	Visual C ++
2.6 - 3.2	Visual Studio 2008	Visual C ++ 9.0
3.3 - 3.4	Visual Studio 2010	Visual C ++ 10.0
3.5	Visual Studio 2015	Visual C ++ 14.0

wiki.python.org

Examples

`SomePackage`

```
$ pip install SomePackage
```

```
$ pip install SomePackage
```

```
$ pip install SomePackage
```

Linux / Unix`sudo`

```
$ pip install SomePackage
```

pip install 。

`freeze`

```
$ pip install SomePackage
```

```
$ pip uninstall SomePackage
```

`pip`

```
$ pip list
# example output
docutils (0.9.1)
Jinja2 (2.6)
Pygments (1.5)
Sphinx (1.1.2)
```

```
$ pip list
# example output
docutils (0.9.1)
Jinja2 (2.6)
Pygments (1.5)
Sphinx (1.1.2)
```

```
$ pip install --upgrade SomePackage
```

`SomePackage`。 pip。

pip

```
$ pip install --upgrade SomePackage
```

Unix

```
$ pip install --upgrade SomePackage
```

Windows。

Linux

`pip`。 Linux

```
pip list --outdated --local | grep -v '^\-e' | cut -d = -f 1  | xargs -n1 pip install -U
```

virtualenv。 `pip install -U`。 。

Windows

`pip`。 Windows

```
for /F "delims= " %i in ('pip list --outdated --local') do pip install -U %i
```

virtualenv。 `pip install -U`。 。

requirements.txt

`pipfreezerequirements.txt`。

```
pip freeze > requirements.txt
```

`requirements.txt`。

virtualenvrequirements.txt

`pipfreezerequirements.txt`。

```
pip freeze --local > requirements.txt
```

`--local`virtualenv。 。

Pythonpip

Python 3Python 2Python。 Python 23。

Python 2

```
pip install [package]
```

```
pip install [package]
```

Python 3

```
pip install [package]
```

python

```
pip install [package]
```

OS-X / Linux / Unixpythonpython。 `sudo`。

Windowspython - - `python -c"import sys;print(sys.path);"`python`python -c"import sys;print(sys.path);"py -3.5 -c"import sys;print(sys.path);"pip --versionpip --version`

Windowspython 2python 3python 33.4python`py` 。

```
pip install [package]
```

pythonpython `virtualenvvenv` python。

pythonPython Package Index。 Windowsvcvarsall.bat。

CC ++Python*PyPI*Windows。

Christoph Gohlke。 **-cp *NN* -**pythonWindows 32python **win64** **-win32-**64python**-win_amd64 -** pythonPython 34**-cp *34*-**。 pippypi。

pythonWindowspythonpython。

Python 2.7Visual Studio 2008Python 3.33.4Visual Studio 2010Python 3.5+Visual Studio 2015。

- “ Python 2.7Visual C ++ ”Microsoft
- “ Windows SDK for Windows 7.NET Framework 4 ”v7.1Microsoft
- Visual Studio 2015 **CC ++ - *8***。

。

pip - 。

*pypiChristop*pythonpython。 pythonpython。 python - PIL *6python 3* 。

bug pypi。 python。

Pip。 `V0.98V1.0-rc1` `pip installV0.98` - `--pip install --pre` `--pip install --pre --upgrade` 。 。

pipgithub。

1. 。 `pip install` *path / to / downloaded / file* `pip install`pip。
2. pip `pip install` *URL / of / package / repository* - `--trusted-host` - `--client-cert`/`--proxy`。

```
    > py -3.5-32 -m venv demo-pip
    > demo-pip\Scripts\activate.bat
    > python -m pip install -U pip
    Collecting pip
      Using cached pip-9.0.1-py2.py3-none-any.whl
    Installing collected packages: pip
      Found existing installation: pip 8.1.1
        Uninstalling pip-8.1.1:
          Successfully uninstalled pip-8.1.1
    Successfully installed pip-9.0.1
    > pip install git+https://github.com/sphinx-doc/sphinx/
    Collecting git+https://github.com/sphinx-doc/sphinx/
      Cloning https://github.com/sphinx-doc/sphinx/ to c:\users\steve-
~1\appdata\local\temp\pip-04yn9hpp-build
    Collecting six>=1.5 (from Sphinx==1.7.dev20170506)
      Using cached six-1.10.0-py2.py3-none-any.whl
    Collecting Jinja2>=2.3 (from Sphinx==1.7.dev20170506)
      Using cached Jinja2-2.9.6-py2.py3-none-any.whl
    Collecting Pygments>=2.0 (from Sphinx==1.7.dev20170506)
      Using cached Pygments-2.2.0-py2.py3-none-any.whl
    Collecting docutils>=0.11 (from Sphinx==1.7.dev20170506)
```

```
      Using cached docutils-0.13.1-py3-none-any.whl
    Collecting snowballstemmer>=1.1 (from Sphinx==1.7.dev20170506)
      Using cached snowballstemmer-1.2.1-py2.py3-none-any.whl
    Collecting babel!=2.0,>=1.3 (from Sphinx==1.7.dev20170506)
      Using cached Babel-2.4.0-py2.py3-none-any.whl
    Collecting alabaster<0.8,>=0.7 (from Sphinx==1.7.dev20170506)
      Using cached alabaster-0.7.10-py2.py3-none-any.whl
    Collecting imagesize (from Sphinx==1.7.dev20170506)
      Using cached imagesize-0.7.1-py2.py3-none-any.whl
    Collecting requests>=2.0.0 (from Sphinx==1.7.dev20170506)
      Using cached requests-2.13.0-py2.py3-none-any.whl
    Collecting typing (from Sphinx==1.7.dev20170506)
      Using cached typing-3.6.1.tar.gz
    Requirement already satisfied: setuptools in f:\toolbuild\temp\demo-pip\lib\site-packages
(from Sphinx==1.7.dev20170506)
    Collecting sphinxcontrib-websupport (from Sphinx==1.7.dev20170506)
      Downloading sphinxcontrib_websupport-1.0.0-py2.py3-none-any.whl
    Collecting colorama>=0.3.5 (from Sphinx==1.7.dev20170506)
      Using cached colorama-0.3.9-py2.py3-none-any.whl
    Collecting MarkupSafe>=0.23 (from Jinja2>=2.3->Sphinx==1.7.dev20170506)
      Using cached MarkupSafe-1.0.tar.gz
    Collecting pytz>=0a (from babel!=2.0,>=1.3->Sphinx==1.7.dev20170506)
      Using cached pytz-2017.2-py2.py3-none-any.whl
    Collecting sqlalchemy>=0.9 (from sphinxcontrib-websupport->Sphinx==1.7.dev20170506)
      Downloading SQLAlchemy-1.1.9.tar.gz (5.2MB)
        100% |################################| 5.2MB 220kB/s
    Collecting whoosh>=2.0 (from sphinxcontrib-websupport->Sphinx==1.7.dev20170506)
      Downloading Whoosh-2.7.4-py2.py3-none-any.whl (468kB)
        100% |################################| 471kB 1.1MB/s
    Installing collected packages: six, MarkupSafe, Jinja2, Pygments, docutils,
snowballstemmer, pytz, babel, alabaster, imagesize, requests, typing, sqlalchemy, whoosh,
sphinxcontrib-websupport, colorama, Sphinx
      Running setup.py install for MarkupSafe ... done
      Running setup.py install for typing ... done
      Running setup.py install for sqlalchemy ... done
      Running setup.py install for Sphinx ... done
    Successfully installed Jinja2-2.9.6 MarkupSafe-1.0 Pygments-2.2.0 Sphinx-1.7.dev20170506
alabaster-0.7.10 babel-2.4.0 colorama-0.3.9 docutils-0.13.1 imagesize-0.7.1 pytz-2017.2
requests-2.13.0 six-1.10.0 snowballstemmer-1.2.1 sphinxcontrib-websupport-1.0.0 sqlalchemy-
1.1.9 typing-3.6.1 whoosh-2.7.4
```

URL`git+`。

3. `git` `mercurial` *DVCS*`pip install` /// -requires.text *`pip install -r requires.txtpython setup.py install`*。 git `git pull origin masterpip uninstall` *package-name*`git checkout`。

pipPyPI https://riptutorial.com/zh-TW/python/topic/1781/pip-pypi

29: PostgreSQL

Examples

PostgreSQL。 `psycopg2`。

pip

```
pip install psycopg2
```

`my_databasemy_table`。

ID		
1		

`psycopg2`。

```
import psycopg2

# Establish a connection to the existing database 'my_database' using
# the user 'my_user' with password 'my_password'
con = psycopg2.connect("host=localhost dbname=my_database user=my_user password=my_password")

# Create a cursor
cur = con.cursor()

# Insert a record into 'my_table'
cur.execute("INSERT INTO my_table(id, first_name, last_name) VALUES (2, 'Jane', 'Doe');")

# Commit the current transaction
con.commit()

# Retrieve all records from 'my_table'
cur.execute("SELECT * FROM my_table;")
results = cur.fetchall()

# Close the database connection
con.close()

# Print the results
print(results)

# OUTPUT: [(1, 'John', 'Doe'), (2, 'Jane', 'Doe')]
```

PostgreSQL https://riptutorial.com/zh-TW/python/topic/3374/postgresql

30: py.test

Examples

py.test

py.testPython。 pip

```
pip install pytest
```

projectroot/module/code.py

```
pip install pytest
```

projectroot/tests/test_code.py。 **test_**。

```
pip install pytest
```

projectrootpy.test

```
pip install pytest
```

```
# projectroot/tests/test_code.py
from module import code


def test_add__failing():
    assert code.add(10, 11) == 33
```

```
# projectroot/tests/test_code.py
from module import code


def test_add__failing():
    assert code.add(10, 11) == 33
```

。 。 。

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
```

```
        self.foo = 1
        self.bar = 2
```

`Stuff。 。`

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

`my_stuff = get_prepped_stuff()。`

py.test

`。 。`

`get_prepped_stuffprepped_stuff。 。 @pytest.fixture`fixture。

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

。 。 py.testfixturefixture。 ;

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

`。 my_stuff = prepped_stuffprepped_stuff 。`

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

.....。

。

Stuff。

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
```

```
        self.foo = 1
        self.bar = 2
```

fixtures。 。 fixture。

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

。 **yield fixture**。 `returnyieldyield` 。 `yield_fixture`py.test。

```
# projectroot/module/stuff.py
class Stuff(object):
    def prep(self):
        self.foo = 1
        self.bar = 2
```

py.test fixtureyield

py.test https://riptutorial.com/zh-TW/python/topic/7054/py-test

31: pyaudio

PyAudioI / OPortAudioPython。 PyAudioPython。 PyAudio

1.pyPortAudio / fastaudioPortAudio v18 APIPython。

2.tkSnackTcl / TkPython。

stream_callback。 stream_callback
1。
2。
3.PortAudiopaAbort。
Stream.readStream.write。
PortAudio
http://portaudio.com/docs/v19-doxydocs/portaudio_8h.html#a8a60fb2a5ec9cbade3f54a9c978e2710

Examples

I / O.

```
"""PyAudio Example: Play a wave file (callback version)."""

import pyaudio
import wave
import time
import sys

if len(sys.argv) < 2:
    print("Plays a wave file.\n\nUsage: %s filename.wav" % sys.argv[0])
    sys.exit(-1)

wf = wave.open(sys.argv[1], 'rb')

# instantiate PyAudio (1)
p = pyaudio.PyAudio()

# define callback (2)
def callback(in_data, frame_count, time_info, status):
    data = wf.readframes(frame_count)
    return (data, pyaudio.paContinue)

# open stream using callback (3)
stream = p.open(format=p.get_format_from_width(wf.getsampwidth()),
                channels=wf.getnchannels(),
                rate=wf.getframerate(),
                output=True,
                stream_callback=callback)

# start the stream (4)
stream.start_stream()
```

```
# wait for stream to finish (5)
while stream.is_active():
    time.sleep(0.1)

# stop stream (6)
stream.stop_stream()
stream.close()
wf.close()

# close PyAudio (7)
p.terminate()
```

/PyAudio2。PyAudio。`callback(<input_data>, <frame_count>, <time_info>, <status_flag>)` `frame_count`/。

pyaudio.Stream.start_stream 4**pyaudio.paComplete** 。

5。

I / O.

msgstr“”“PyAudio。”“

```
import pyaudio
import wave
import sys

CHUNK = 1024

if len(sys.argv) < 2:
    print("Plays a wave file.\n\nUsage: %s filename.wav" % sys.argv[0])
    sys.exit(-1)

wf = wave.open(sys.argv[1], 'rb')

# instantiate PyAudio (1)
p = pyaudio.PyAudio()

# open stream (2)
stream = p.open(format=p.get_format_from_width(wf.getsampwidth()),
                channels=wf.getnchannels(),
                rate=wf.getframerate(),
                output=True)

# read data
data = wf.readframes(CHUNK)

# play stream (3)
while len(data) > 0:
    stream.write(data)
    data = wf.readframes(CHUNK)

# stop stream (4)
stream.stop_stream()
stream.close()

# close PyAudio (5)
```

```
p.terminate()
```

PyAudio**pyaudio.PyAudio** 1PyAudioportaudio。

pyaudio.PyAudio.open 2。 **pyaudio.Stream**。

pyaudio.Stream.writepyaudio.Stream.read。 3

“ ”**pyaudio.Stream.writepyaudio.Stream.read**//。 “”

pyaudio.Stream.stop_stream/**pyaudio.Stream.close**。 4

pyaudio.PyAudio.terminate 5portaudio

pyaudio https://riptutorial.com/zh-TW/python/topic/10627/pyaudio

32: pyautogui

pyautogui。 。 0,0。 Python。

Examples

。

```
size()            #gave you the size of the screen
position()      #return current position of mouse
moveTo(200,0,duration=1.5)      #move the cursor  to (200,0) position  with 1.5 second delay

moveRel()           #move the cursor relative to your current position.
click(337,46)            #it will click on the position mention there
dragRel()               #it will drag the mouse relative to position
pyautogui.displayMousePosition()      #gave you the current mouse position but should be done
on terminal.
```

。

```
typewrite('')    #this will type the string on the screen where current window has focused.
typewrite(['a','b','left','left','X','Y'])
pyautogui.KEYBOARD_KEYS    #get the list of all the keyboard_keys.
pyautogui.hotkey('ctrl','o')    #for the combination of keys to enter.
```

ScreenShot

。

```
.screenshot('c:\\path')         #get the screenshot.
.locateOnScreen('c:\\path')     #search that image on screen and get the coordinates for you.
locateCenterOnScreen('c:\\path')       #get the coordinate for the image on screen.
```

pyautogui https://riptutorial.com/zh-TW/python/topic/9432/pyautogui

33: pygame

PygamePython。 http://www.pygame.org/ 。

- pygame.mixer.init= 22050= -16= 2= 4096
- pygame.mixer.pre_init
- pygame.mixer.quit
- pygame.mixer.get_init
- pygame.mixer.stop
- pygame.mixer.pause
- pygame.mixer.unpause
- pygame.mixer.fadeout
- pygame.mixer.set_num_channels
- pygame.mixer.get_num_channels
- pygame.mixer.set_reserved
- pygame.mixer.find_channel
- pygame.mixer.get_busy

	。
	`FalseTrue find_channel()TrueFalse`

Examples

pygame

`pip`

```
pip install pygame
```

`conda`

```
pip install pygame
```

http //www.pygame.org/download.shtml

。

http://www.pygame.org/

Pygame

`pygame.mixerpygame。 mixer`15。

pygame.init()pygamepygame.mixer。

。 。 。

```
pygame.mixer.init(frequency=22050, size=-16, channels=2, buffer=4096)
```

pygame.mixer.get_init() TrueFalse。 /pygame.mixer.quit() 。 。

---

pygame.mixer.pause()。 pygame.mixer.unpause() 。 pygame.mixer.fadeout()。 。

。 8。 pygame.mixer.set_num_channels() 。 。 。

pygame.mixer.get_channels(count) pygame.mixer.get_channels(count) 。 。
pygame.mixer.set_reserved(count)。 。 。

pygame.mixer.find_channel(force)。 。 forceFalseNone 。 forcetrue。

pygame https://riptutorial.com/zh-TW/python/topic/8761/pygame

34: Pyglet

PygletPython。 。 [pyglet.org] [1]。 [1]http//pyglet.org

Examples

PygletHello World

```
import pyglet
window = pyglet.window.Window()
label = pyglet.text.Label('Hello, world',
                      font_name='Times New Roman',
                      font_size=36,
                      x=window.width//2, y=window.height//2,
                      anchor_x='center', anchor_y='center')
@window.event
def on_draw():
    window.clear()
    label.draw()
pyglet.app.run()
```

Pyglet

Python

Python 2

```
pip install pyglet
```

Python 3

```
pip install pyglet
```

Pyglet

```
sound = pyglet.media.load(sound.wav)
sound.play()
```

PygletOpenGL

```
import pyglet
from pyglet.gl import *

win = pyglet.window.Window()

@win.event()
def on_draw():
    #OpenGL goes here. Use OpenGL as normal.
```

```
pyglet.app.run()
```

PygletOpenGL

```
import pyglet
from pyglet.gl import *

win = pyglet.window.Window()
glClear(GL_COLOR_BUFFER_BIT)

@win.event
def on_draw():
    glBegin(GL_POINTS)
    glVertex2f(x, y) #x is desired distance from left side of window, y is desired distance
from bottom of window
    #make as many vertexes as you want
    glEnd
```

GL_POINTSGL_LINE_LOOP ◦

Pyglet https://riptutorial.com/zh-TW/python/topic/8208/pyglet

35: PyInstaller - Python

- pyinstaller [options] script [script ...] | spec

PyInstallerpython。 python。 numpyDjangoOpenCv。

- PyinstallerPython 2.7Python 3.3+
- PyinstallerWindowsLinuxMac OS X。
- 。 WindowsLinux。 WindowsPyInstallerWindows

Examples

Pyinstallerpython。 pip

```
pip install pyinstaller
```

Windows
Windows pywin32pypiwin32。 pippyinstaller。

Mac OS X
PyInstallerMac OS XPython 2.7。 PythonPyQTNumpyMatplotlib。 MacPortsHomebrew 。

pipPyPI。
PyInstaller*develop*。

`setup.py`。 `python setup.py install`PyInstaller。

`pyinstaller`。
`pyinstaller --version`。 pyinstaller。

Pyinstaller

```
pyinstaller myfile.py
```

Pyinstaller

- **myfile.spec**`myfile.py`
- `myfile.py`**build**
- `myfile.py`**dist**
-

dist

pyinstaller。 。

'dist \ myfile \ myfile.exe'。

PyInstaller`myscript.py` `myscript` `myscript`windows`myscript.exe` 。
zip。

`-D--onedir--onedir`

```
pyinstaller myscript.py -D
```

。 。

。 。

。
/。

```
pyinstaller myscript.py -F
```

`-F--onefile` 。 `myscript.exe`。

。 。

PyInstaller - Python https://riptutorial.com/zh-TW/python/topic/2289/pyinstaller----python

36: Python HTTP Server

Examples

HTTP

Python 2.x 2.3

```
python -m SimpleHTTPServer 9000
```

Python 3.x 3.0

```
python -m SimpleHTTPServer 9000
```

```
9000。

8000。

-msys.path.py。
```

localhostPython

```
python -m SimpleHTTPServer 9000
```

Web

Python 2.x 2.3

```
import SimpleHTTPServer
import SocketServer

PORT = 8000

handler = SimpleHTTPServer.SimpleHTTPRequestHandler
httpd = SocketServer.TCPServer(("localhost", PORT), handler)
print "Serving files at port {}".format(PORT)
httpd.serve_forever()
```

Python 3.x 3.0

```
import SimpleHTTPServer
```

```
import SocketServer

PORT = 8000

handler = SimpleHTTPServer.SimpleHTTPRequestHandler
httpd = SocketServer.TCPServer(("localhost", PORT), handler)
print "Serving files at port {}".format(PORT)
httpd.serve_forever()
```

`SocketServer`。

`SocketServerTCPServer`TCP。 IP。

`SimpleHTTPServerSimpleHTTPRequestHandler`。

。

HTTP

Python 2.x 2.3

python -m SimpleHTTPServer 8000

Python 3.x 3.0

python -m http.server 8000

'-m"sys.path".py'。

localhost8000

Directory listing for /

- facade.py
- factory.py
- server.py

SimpleHTTPServerAPI

`python -m SimpleHTTPServer 9000`

SimpleHTTPServer https://hg.python.org/cpython/file/2.7/Lib/SimpleHTTPServer.py
BaseHTTPServer https://hg.python.org/cpython/file 。 /2.7/Lib/BaseHTTPServer.py 。

Python`9000SimpleHTTPServer`。 SimpleHTTPServer

```
def test(HandlerClass = SimpleHTTPRequestHandler,
```

```
         ServerClass = BaseHTTPServer.HTTPServer):
    BaseHTTPServer.test(HandlerClass, ServerClass)


if __name__ == '__main__':
    test()
```

ServerClass。 BaseHTTPServer.test

```
def test(HandlerClass = SimpleHTTPRequestHandler,
         ServerClass = BaseHTTPServer.HTTPServer):
    BaseHTTPServer.test(HandlerClass, ServerClass)


if __name__ == '__main__':
    test()
```

。 。 。

SocketServer

```
def test(HandlerClass = SimpleHTTPRequestHandler,
         ServerClass = BaseHTTPServer.HTTPServer):
    BaseHTTPServer.test(HandlerClass, ServerClass)


if __name__ == '__main__':
    test()
```

https://hg.python.org/cpython/file/2.7/Lib/BaseHTTPServer.py
https://hg.python.org/cpython/file/2.7/Lib/SocketServer.py。

BaseHTTPRequestHandlerGETPOSTPUT

```
# from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer # python2
from http.server import BaseHTTPRequestHandler, HTTPServer # python3
class HandleRequests(BaseHTTPRequestHandler):
    def _set_headers(self):
        self.send_response(200)
        self.send_header('Content-type', 'text/html')
        self.end_headers()

    def do_GET(self):
        self._set_headers()
        self.wfile.write("received get request")

    def do_POST(self):
        '''Reads post request body'''
        self._set_headers()
        content_len = int(self.headers.getheader('content-length', 0))
        post_body = self.rfile.read(content_len)
        self.wfile.write("received post request:<br>{}".format(post_body))

    def do_PUT(self):
        self.do_POST()
```

```
host = ''
port = 80
HTTPServer((host, port), HandleRequests).serve_forever()
```

curl

```
# from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer # python2
from http.server import BaseHTTPRequestHandler, HTTPServer # python3
class HandleRequests(BaseHTTPRequestHandler):
    def _set_headers(self):
        self.send_response(200)
        self.send_header('Content-type', 'text/html')
        self.end_headers()

    def do_GET(self):
        self._set_headers()
        self.wfile.write("received get request")

    def do_POST(self):
        '''Reads post request body'''
        self._set_headers()
        content_len = int(self.headers.getheader('content-length', 0))
        post_body = self.rfile.read(content_len)
        self.wfile.write("received post request:<br>{}".format(post_body))

    def do_PUT(self):
        self.do_POST()

host = ''
port = 80
HTTPServer((host, port), HandleRequests).serve_forever()
```

Python HTTP Server https://riptutorial.com/zh-TW/python/topic/4247/python-http-server

37: Python Lex-Yacc

PLYlexyaccPython。

1.
2. Github

Examples

PLY

python2 / 3PLY

1. 。
2. zip
3. `ply-3.10`
4. `python setup.py install`

PLY。 python`import ply.lex`。

`pip`PLY。

“HelloWorld” PLY -

PLY。

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

tokens = (
    'PLUS',
    'MINUS',
    'TIMES',
    'DIV',
    'LPAREN',
    'RPAREN',
    'NUMBER',
)

t_ignore = ' \t'

t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIV     = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

def t_NUMBER( t ) :
    r'[0-9]+'
    t.value = int( t.value )
    return t
```

```
def t_newline( t ):
  r'\n+'
  t.lexer.lineno += len( t.value )

def t_error( t ):
  print("Invalid Token:",t.value[0])
  t.lexer.skip( 1 )

lexer = lex.lex()

precedence = (
    ( 'left', 'PLUS', 'MINUS' ),
    ( 'left', 'TIMES', 'DIV' ),
    ( 'nonassoc', 'UMINUS' )
)

def p_add( p ) :
    'expr : expr PLUS expr'
    p[0] = p[1] + p[3]

def p_sub( p ) :
    'expr : expr MINUS expr'
    p[0] = p[1] - p[3]

def p_expr2uminus( p ) :
    'expr : MINUS expr %prec UMINUS'
    p[0] = - p[2]

def p_mult_div( p ) :
    '''expr : expr TIMES expr
            | expr DIV expr'''

    if p[2] == '*' :
        p[0] = p[1] * p[3]
    else :
        if p[3] == 0 :
            print("Can't divide by 0")
            raise ZeroDivisionError('integer division by 0')
        p[0] = p[1] / p[3]

def p_expr2NUM( p ) :
    'expr : NUMBER'
    p[0] = p[1]

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

def p_error( p ):
    print("Syntax error in input!")

parser = yacc.yacc()

res = parser.parse("-4*-(3-5)") # the input
print(res)
```

calc.py◦

```
from ply import lex
```

```
import ply.yacc as yacc

tokens = (
    'PLUS',
    'MINUS',
    'TIMES',
    'DIV',
    'LPAREN',
    'RPAREN',
    'NUMBER',
)

t_ignore = ' \t'

t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIV     = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

def t_NUMBER( t ) :
    r'[0-9]+'
    t.value = int( t.value )
    return t

def t_newline( t ):
  r'\n+'
  t.lexer.lineno += len( t.value )

def t_error( t ):
  print("Invalid Token:",t.value[0])
  t.lexer.skip( 1 )

lexer = lex.lex()

precedence = (
    ( 'left', 'PLUS', 'MINUS' ),
    ( 'left', 'TIMES', 'DIV' ),
    ( 'nonassoc', 'UMINUS' )
)

def p_add( p ) :
    'expr : expr PLUS expr'
    p[0] = p[1] + p[3]

def p_sub( p ) :
    'expr : expr MINUS expr'
    p[0] = p[1] - p[3]

def p_expr2uminus( p ) :
    'expr : MINUS expr %prec UMINUS'
    p[0] = - p[2]

def p_mult_div( p ) :
    '''expr : expr TIMES expr
            | expr DIV expr'''

    if p[2] == '*' :
        p[0] = p[1] * p[3]
    else :
```

```
        if p[3] == 0 :
            print("Can't divide by 0")
            raise ZeroDivisionError('integer division by 0')
        p[0] = p[1] / p[3]

def p_expr2NUM( p ) :
    'expr : NUMBER'
    p[0] = p[1]

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

def p_error( p ):
    print("Syntax error in input!")

parser = yacc.yacc()

res = parser.parse("-4*-(3-5)") # the input
print(res)
```

```
-4 * - (3 - 5)。
```

1Lex

1 。

。

```
    import ply.lex as lex

    # List of token names. This is always required
    tokens = [
       'NUMBER',
       'PLUS',
       'MINUS',
       'TIMES',
       'DIVIDE',
       'LPAREN',
       'RPAREN',
    ]

    # Regular expression rules for simple tokens
    t_PLUS    = r'\+'
    t_MINUS   = r'-'
    t_TIMES   = r'\*'
    t_DIVIDE  = r'/'
    t_LPAREN  = r'\('
    t_RPAREN  = r'\)'

    # A regular expression rule with some action code
    def t_NUMBER(t):
        r'\d+'
        t.value = int(t.value)
        return t

    # Define a rule so we can track line numbers
    def t_newline(t):
```

```
        r'\n+'
        t.lexer.lineno += len(t.value)

    # A string containing ignored characters (spaces and tabs)
    t_ignore  = ' \t'

    # Error handling rule
    def t_error(t):
        print("Illegal character '%s'" % t.value[0])
        t.lexer.skip(1)

    # Build the lexer
    lexer = lex.lex()

    # Give the lexer some input
    lexer.input(data)

    # Tokenize
    while True:
        tok = lexer.token()
        if not tok:
            break      # No more input
        print(tok)
```

`calclex.py` ◦ Yacc◦

---

1. `import ply.lex`

2. `tokens`◦ ◦

```
    import ply.lex as lex

    # List of token names. This is always required
    tokens = [
       'NUMBER',
       'PLUS',
       'MINUS',
       'TIMES',
       'DIVIDE',
       'LPAREN',
       'RPAREN',
    ]

    # Regular expression rules for simple tokens
    t_PLUS    = r'\+'
    t_MINUS   = r'-'
    t_TIMES   = r'\*'
    t_DIVIDE  = r'/'
    t_LPAREN  = r'\('
    t_RPAREN  = r'\)'

    # A regular expression rule with some action code
    def t_NUMBER(t):
        r'\d+'
        t.value = int(t.value)
        return t
```

```
# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

tokens。

3. 。 t_。

- t_PLUS = r'\+'

- 。

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
```

```
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

doc。 `LexToken`。

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t
```

```
# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

- `LexToken`

  1. `t.type` `'NUMBER'` `'PLUS'`。 `t.type``t_`。
  2. `t.value`lexeme
  3. `t.lineno`。 `t.lineno` `t_newline`。 `t_newline`。

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t
```

```
# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

4. `t.lexpos` 。

- 。 t_ignore_。

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
```

```
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

...

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)
```

```
    # A string containing ignored characters (spaces and tabs)
    t_ignore  = ' \t'

    # Error handling rule
    def t_error(t):
        print("Illegal character '%s'" % t.value[0])
        t.lexer.skip(1)

    # Build the lexer
    lexer = lex.lex()

    # Give the lexer some input
    lexer.input(data)

    # Tokenize
    while True:
        tok = lexer.token()
        if not tok:
            break      # No more input
        print(tok)
```

。 。

```
t_ignore = "<characters to ignore>"
```

```
    import ply.lex as lex

    # List of token names. This is always required
    tokens = [
       'NUMBER',
       'PLUS',
       'MINUS',
       'TIMES',
       'DIVIDE',
       'LPAREN',
       'RPAREN',
    ]

    # Regular expression rules for simple tokens
    t_PLUS    = r'\+'
    t_MINUS   = r'-'
    t_TIMES   = r'\*'
    t_DIVIDE  = r'/'
    t_LPAREN  = r'\('
    t_RPAREN  = r'\)'

    # A regular expression rule with some action code
    def t_NUMBER(t):
        r'\d+'
        t.value = int(t.value)
        return t

    # Define a rule so we can track line numbers
    def t_newline(t):
        r'\n+'
        t.lexer.lineno += len(t.value)

    # A string containing ignored characters (spaces and tabs)
    t_ignore  = ' \t'
```

```
# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

- lex

  1. ◦
  2. ◦

=== ◦

- ◦ `t.typet.value`◦

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
```

```
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
```

```
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

- -

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)
```

```
# Build the lexer
lexer = lex.lex()

# Give the lexer some input
lexer.input(data)

# Tokenize
while True:
    tok = lexer.token()
    if not tok:
        break      # No more input
    print(tok)
```

- t_error。

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()

# Give the lexer some input
```

```
    lexer.input(data)

    # Tokenize
    while True:
        tok = lexer.token()
        if not tok:
            break      # No more input
        print(tok)
```

`t.lexer.skip(n)`n。

4. `lexer = lex.lex()`lexer = lex.lex() 。

。

```
import ply.lex as lex

# List of token names. This is always required
tokens = [
   'NUMBER',
   'PLUS',
   'MINUS',
   'TIMES',
   'DIVIDE',
   'LPAREN',
   'RPAREN',
]

# Regular expression rules for simple tokens
t_PLUS    = r'\+'
t_MINUS   = r'-'
t_TIMES   = r'\*'
t_DIVIDE  = r'/'
t_LPAREN  = r'\('
t_RPAREN  = r'\)'

# A regular expression rule with some action code
def t_NUMBER(t):
    r'\d+'
    t.value = int(t.value)
    return t

# Define a rule so we can track line numbers
def t_newline(t):
    r'\n+'
    t.lexer.lineno += len(t.value)

# A string containing ignored characters (spaces and tabs)
t_ignore  = ' \t'

# Error handling rule
def t_error(t):
    print("Illegal character '%s'" % t.value[0])
    t.lexer.skip(1)

# Build the lexer
lexer = lex.lex()
```

```
    # Give the lexer some input
    lexer.input(data)

    # Tokenize
    while True:
        tok = lexer.token()
        if not tok:
            break      # No more input
        print(tok)
```

`lexer.input(data)`data

`lexer.token()` ◦

```
    import ply.lex as lex

    # List of token names. This is always required
    tokens = [
       'NUMBER',
       'PLUS',
       'MINUS',
       'TIMES',
       'DIVIDE',
       'LPAREN',
       'RPAREN',
    ]

    # Regular expression rules for simple tokens
    t_PLUS    = r'\+'
    t_MINUS   = r'-'
    t_TIMES   = r'\*'
    t_DIVIDE  = r'/'
    t_LPAREN  = r'\('
    t_RPAREN  = r'\)'

    # A regular expression rule with some action code
    def t_NUMBER(t):
        r'\d+'
        t.value = int(t.value)
        return t

    # Define a rule so we can track line numbers
    def t_newline(t):
        r'\n+'
        t.lexer.lineno += len(t.value)

    # A string containing ignored characters (spaces and tabs)
    t_ignore  = ' \t'

    # Error handling rule
    def t_error(t):
        print("Illegal character '%s'" % t.value[0])
        t.lexer.skip(1)

    # Build the lexer
    lexer = lex.lex()

    # Give the lexer some input
    lexer.input(data)
```

```
    # Tokenize
    while True:
        tok = lexer.token()
        if not tok:
            break      # No more input
        print(tok)
```

2Yacc

1 - Context Free GrammarsCFG。 。 LALR。

```
# Yacc example

import ply.yacc as yacc

# Get the token map from the lexer. This is required.
from calclex import tokens

def p_expression_plus(p):
    'expression : expression PLUS term'
    p[0] = p[1] + p[3]

def p_expression_minus(p):
    'expression : expression MINUS term'
    p[0] = p[1] - p[3]

def p_expression_term(p):
    'expression : term'
    p[0] = p[1]

def p_term_times(p):
    'term : term TIMES factor'
    p[0] = p[1] * p[3]

def p_term_div(p):
    'term : term DIVIDE factor'
    p[0] = p[1] / p[3]

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

def p_factor_num(p):
    'factor : NUMBER'
    p[0] = p[1]

def p_factor_expr(p):
    'factor : LPAREN expression RPAREN'
    p[0] = p[2]

# Error rule for syntax errors
def p_error(p):
    print("Syntax error in input!")

# Build the parser
parser = yacc.yacc()

while True:
```

```
try:
    s = raw_input('calc > ')
except EOFError:
    break
if not s: continue
result = parser.parse(s)
print(result)
```

- ◦ ◦ p◦ `p[i]`

```
# Yacc example

import ply.yacc as yacc

# Get the token map from the lexer. This is required.
from calclex import tokens

def p_expression_plus(p):
    'expression : expression PLUS term'
    p[0] = p[1] + p[3]

def p_expression_minus(p):
    'expression : expression MINUS term'
    p[0] = p[1] - p[3]

def p_expression_term(p):
    'expression : term'
    p[0] = p[1]

def p_term_times(p):
    'term : term TIMES factor'
    p[0] = p[1] * p[3]

def p_term_div(p):
    'term : term DIVIDE factor'
    p[0] = p[1] / p[3]

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

def p_factor_num(p):
    'factor : NUMBER'
    p[0] = p[1]

def p_factor_expr(p):
    'factor : LPAREN expression RPAREN'
    p[0] = p[2]

# Error rule for syntax errors
def p_error(p):
    print("Syntax error in input!")

# Build the parser
parser = yacc.yacc()

while True:
```

```
    try:
        s = raw_input('calc > ')
    except EOFError:
        break
    if not s: continue
    result = parser.parse(s)
    print(result)
```

- `p[i]`“”`p.value`。 `PLUS+`。
- `p[0]`。 None。 `p[-1]``p[3]` `p` `p[-1]`。

`p_`。

- `p_error(p)` yacc / bison `yyerror`。
- 。

```
# Yacc example

import ply.yacc as yacc

# Get the token map from the lexer. This is required.
from calclex import tokens

def p_expression_plus(p):
    'expression : expression PLUS term'
    p[0] = p[1] + p[3]

def p_expression_minus(p):
    'expression : expression MINUS term'
    p[0] = p[1] - p[3]

def p_expression_term(p):
    'expression : term'
    p[0] = p[1]

def p_term_times(p):
    'term : term TIMES factor'
    p[0] = p[1] * p[3]

def p_term_div(p):
    'term : term DIVIDE factor'
    p[0] = p[1] / p[3]

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

def p_factor_num(p):
    'factor : NUMBER'
    p[0] = p[1]

def p_factor_expr(p):
    'factor : LPAREN expression RPAREN'
    p[0] = p[2]

# Error rule for syntax errors
```

```
def p_error(p):
    print("Syntax error in input!")

# Build the parser
parser = yacc.yacc()

while True:
   try:
       s = raw_input('calc > ')
   except EOFError:
       break
   if not s: continue
   result = parser.parse(s)
   print(result)
```

- 。

```
# Yacc example

import ply.yacc as yacc

# Get the token map from the lexer. This is required.
from calclex import tokens

def p_expression_plus(p):
    'expression : expression PLUS term'
    p[0] = p[1] + p[3]

def p_expression_minus(p):
    'expression : expression MINUS term'
    p[0] = p[1] - p[3]

def p_expression_term(p):
    'expression : term'
    p[0] = p[1]

def p_term_times(p):
    'term : term TIMES factor'
    p[0] = p[1] * p[3]

def p_term_div(p):
    'term : term DIVIDE factor'
    p[0] = p[1] / p[3]

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

def p_factor_num(p):
    'factor : NUMBER'
    p[0] = p[1]

def p_factor_expr(p):
    'factor : LPAREN expression RPAREN'
    p[0] = p[2]

# Error rule for syntax errors
def p_error(p):
    print("Syntax error in input!")
```

```
# Build the parser
parser = yacc.yacc()

while True:
   try:
       s = raw_input('calc > ')
   except EOFError:
       break
   if not s: continue
   result = parser.parse(s)
   print(result)
```

。

- '''symbol : '''
- start = 'foo' foo。
- 。

```
# Yacc example

import ply.yacc as yacc

# Get the token map from the lexer. This is required.
from calclex import tokens

def p_expression_plus(p):
    'expression : expression PLUS term'
    p[0] = p[1] + p[3]

def p_expression_minus(p):
    'expression : expression MINUS term'
    p[0] = p[1] - p[3]

def p_expression_term(p):
    'expression : term'
    p[0] = p[1]

def p_term_times(p):
    'term : term TIMES factor'
    p[0] = p[1] * p[3]

def p_term_div(p):
    'term : term DIVIDE factor'
    p[0] = p[1] / p[3]

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

def p_factor_num(p):
    'factor : NUMBER'
    p[0] = p[1]

def p_factor_expr(p):
    'factor : LPAREN expression RPAREN'
```

```
    p[0] = p[2]

# Error rule for syntax errors
def p_error(p):
    print("Syntax error in input!")

# Build the parser
parser = yacc.yacc()

while True:
   try:
       s = raw_input('calc > ')
   except EOFError:
       break
   if not s: continue
   result = parser.parse(s)
   print(result)
```

◦ `nonassoc`◦ `a < b < ca < b`◦

- `parser.out`yacc◦ /◦

Python Lex-Yacc https://riptutorial.com/zh-TW/python/topic/10510/python-lex-yacc

38: Python

Python`threadingmultiprocessing`API。

Examples

```
from __future__ import print_function
import threading
def counter(count):
    while count > 0:
        print("Count value", count)
        count -= 1
    return

t1 = threading.Thread(target=countdown,args=(10,))
t1.start()
t2 = threading.Thread(target=countdown,args=(20,))
t2.start()
```

PythonCPythonGIL**G** nterpreter 。

Python

David BeazleyYouTubePython

```
from __future__ import print_function
import multiprocessing


def countdown(count):
    while count > 0:
        print("Count value", count)
        count -= 1
    return

if __name__ == "__main__":
    p1 = multiprocessing.Process(target=countdown, args=(10,))
    p1.start()

    p2 = multiprocessing.Process(target=countdown, args=(20,))
    p2.start()

    p1.join()
    p2.join()
```

。 Python VM`GIL`。

`Process.startargstarget`。 `Process.joinp1p2`。

python

- Windows`spawn`。
-

unix3.3`fork`。
forkPOSIX。

- unix3.4+`fork` `forkserverspawnmultiprocessing.set_start_method`。 `forkserverspawn`。

POSIX fork

forkexecve。

fork`MainThread`。

- `MainThreadLock`。 `fork`。

python`multiprocessing`macOSnumpy / accelerated。

。 Queue。

```
import multiprocessing
import queue
my_Queue=multiprocessing.Queue()
#Creates a queue with an undefined maximum size
#this can be dangerous as the queue becomes increasingly large
#it will take a long time to copy data to/from each read/write thread
```

tryexceptblockempty。 `queue.Empty==Truequeue.Empty==False` Iftry'if'。

```
import multiprocessing
import queue
my_Queue=multiprocessing.Queue()
#Creates a queue with an undefined maximum size
#this can be dangerous as the queue becomes increasingly large
#it will take a long time to copy data to/from each read/write thread
```

Python https://riptutorial.com/zh-TW/python/topic/3357/python

39: PythonMutable vs ImmutableHashable

Examples

Python。 。

Immutables

。 。

frozensets。

id。 。 id。 。 .....

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

13 .....。 。

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

。 stack。 。

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
```

```
>>> a
3
>>> id(a)
140128142243328
```

。

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

Mutables

。 。

。

id。

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

ascii。 。

bytearray`id`。 。

。 。

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

...

```
>>> a = 1
```

```
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

Waiiit.....

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

◦ cb◦ cb ◦

```
>>> a = 1
>>> id(a)
140128142243264
>>> a += 2
>>> a
3
>>> id(a)
140128142243328
```

◦ ◦ ◦

```
>>> def list_add3(lin):
    lin += [3]
    return lin

>>> a = [1, 2, 3]
>>> b = list_add3(a)
>>> b
[1, 2, 3, 3]
>>> a
[1, 2, 3, 3]
```

lin◦ lina◦ lina◦ lina ◦ ab◦

◦

```
>>> def list_add3(lin):
    lin += [3]
    return lin

>>> a = [1, 2, 3]
>>> b = list_add3(a)
>>> b
[1, 2, 3, 3]
```

```
>>> a
[1, 2, 3, 3]
```

tina。 。 tina。 tin。

```
>>> def list_add3(lin):
    lin += [3]
    return lin

>>> a = [1, 2, 3]
>>> b = list_add3(a)
>>> b
[1, 2, 3, 3]
>>> a
[1, 2, 3, 3]
```

reverse。

saying focused

PythonMutable vs ImmutableHashable https://riptutorial.com/zh-TW/python/topic/9182/python mutable-vs-immutable-hashable-

40: Python

。 。 lambdamapreduce。

Examples

Lambda

lambda。 lambda。 。 。

```
s=lambda x:x*x
s(2)    =>4
```

Map。 。 。

```
name_lengths = map(len, ["Mary", "Isla", "Sam"])
print(name_lengths)    =>[4, 4, 3]
```

Reduce。 。

。 。

```
total = reduce(lambda a, x: a + x, [0, 1, 2, 3, 4])
print(total)    =>10
```

。 True。

```
arr=[1,2,3,4,5,6]
[i for i in filter(lambda x:x>4,arr)]    # outputs[5,6]
```

Python https://riptutorial.com/zh-TW/python/topic/9552/python

41: PythonShell

Examples

ssh

```
from paramiko import client
ssh = client.SSHClient() # create a new SSHClient object
ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy()) #auto-accept unknown host keys
ssh.connect(hostname, username=username, port=port, password=password) #connect with a host
stdin, stdout, stderr = ssh.exec_command(command) # submit a command to ssh
print stdout.channel.recv_exit_status() #tells the status  1 - job failed
```

PythonShell https://riptutorial.com/zh-TW/python/topic/5709/pythonshell

42: python

Examples

```
from string import Template

data = dict(item = "candy", price = 8, qty = 2)

# define the template
t = Template("Simon bought $qty $item for $price dollar")
print(t.substitute(data))
```

```
from string import Template

data = dict(item = "candy", price = 8, qty = 2)

# define the template
t = Template("Simon bought $qty $item for $price dollar")
print(t.substitute(data))
```

$。 。

。 priceqty。 。 。

“$”。

```
from string import Template

class MyOtherTemplate(Template):
    delimiter = "#"


data = dict(id = 1, name = "Ricardo")
t = MyOtherTemplate("My name is #name and I have the id: #id")
print(t.substitute(data))
```

de docs

python https://riptutorial.com/zh-TW/python/topic/6029/python

43: Pythonpyserial

- ser.read= 1
- ser.readline
- ser.write

	GNU / Linux/ dev / ttyUSB0WindowsCOM3。
	int9600 50,75,110,134,150,200,300,600,1200,1800,2400,4800,9600,19200,38400,57600,115200

pyserial

Examples

```
import serial
#Serial takes these two parameters: serial device and baudrate
ser = serial.Serial('/dev/ttyUSB0', 9600)
```

```
import serial
#Serial takes two parameters: serial device and baudrate
ser = serial.Serial('/dev/ttyUSB0', 9600)
```

```
import serial
#Serial takes two parameters: serial device and baudrate
ser = serial.Serial('/dev/ttyUSB0', 9600)
```

```
import serial
#Serial takes two parameters: serial device and baudrate
ser = serial.Serial('/dev/ttyUSB0', 9600)
```

。

```
import serial
#Serial takes two parameters: serial device and baudrate
ser = serial.Serial('/dev/ttyUSB0', 9600)
```

。

```
import serial
#Serial takes two parameters: serial device and baudrate
ser = serial.Serial('/dev/ttyUSB0', 9600)
```

```
python -m serial.tools.list_ports
```


```
python -m serial.tools.list_ports
```


Python shell。

Pythonpyserial https://riptutorial.com/zh-TW/python/topic/5744/python-pyserial-

44: Python

Examples

except。

```
try:
    res = get_result()
    res = res[0]
    log('got result: %r' % res)
except:
    if not res:
        res = ''
    print('got exception')
```

3

1. except 5KeyboardInterrupt。 。
2. exceptget_resultres。
3. 。 get_result resexceptresNameError。

。 。

```
try:
    res = get_result()
    res = res[0]
    log('got result: %r' % res)
except:
    if not res:
        res = ''
    print('got exception')
```

。 。

。

valueNone

```
def intensive_f(value): # int -> Optional[int]
   # complex, and time-consuming code
   if process_has_failed:
       return None
   return integer_output
```

```
def intensive_f(value): # int -> Optional[int]
   # complex, and time-consuming code
   if process_has_failed:
       return None
   return integer_output
```

`intensive_f`。 。

```
def intensive_f(value): # int -> Optional[int]
   # complex, and time-consuming code
   if process_has_failed:
       return None
   return integer_output
```

pythonic

```
def intensive_f(value): # int -> Optional[int]
   # complex, and time-consuming code
   if process_has_failed:
       return None
   return integer_output
```

。 `assertAssertionError`。

。

```
def intensive_f(value): # int -> Optional[int]
   # complex, and time-consuming code
   if process_has_failed:
       return None
   return integer_output
```

```
def intensive_f(value): # int -> Optional[int]
   # complex, and time-consuming code
   if process_has_failed:
       return None
   return integer_output
```

。 。

`dict.get(key, default)`。

Python https://riptutorial.com/zh-TW/python/topic/4700/python

45: PythonExcel

Examples

Excel。

```
import os, sys
from openpyxl import Workbook
from datetime import datetime

dt = datetime.now()
list_values = [["01/01/2016", "05:00:00", 3], \
               ["01/02/2016", "06:00:00", 4], \
               ["01/03/2016", "07:00:00", 5], \
               ["01/04/2016", "08:00:00", 6], \
               ["01/05/2016", "09:00:00", 7]]

# Create a Workbook on Excel:
wb = Workbook()
sheet = wb.active
sheet.title = 'data'

# Print the titles into Excel Workbook:
row = 1
sheet['A'+str(row)] = 'Date'
sheet['B'+str(row)] = 'Hour'
sheet['C'+str(row)] = 'Value'

# Populate with data
for item in list_values:
    row += 1
    sheet['A'+str(row)] = item[0]
    sheet['B'+str(row)] = item[1]
    sheet['C'+str(row)] = item[2]

# Save a file by date:
filename = 'data_' + dt.strftime("%Y%m%d_%I%M%S") + '.xlsx'
wb.save(filename)

# Open the file for the user:
os.chdir(sys.path[0])
os.system('start excel.exe "%s\\%s"' % (sys.path[0], filename, ))
```

OpenPyXL

OpenPyXL`xlsx/xlsm/xltx/xltm`。

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

`load_workbook()` `read_only` `True` read_only `xlsx`

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

`workbook.sheetsworkbook.sheets`

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

`workbook.get_sheet_names()` ◦

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

`sheet.rows`◦

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

`rowrowsCellCell.value`Cell◦

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

openpyxl`tabColor`

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

```
import openpyxl as opx
#To change an existing wookbook we located it by referencing its path
workbook = opx.load_workbook(workbook_path)
```

xlsxwriterexcel

```
import xlsxwriter

# sample data
chart_data = [
    {'name': 'Lorem', 'value': 23},
    {'name': 'Ipsum', 'value': 48},
    {'name': 'Dolor', 'value': 15},
    {'name': 'Sit', 'value': 8},
    {'name': 'Amet', 'value': 32}
]

# excel file path
```

```
xls_file = 'chart.xlsx'

# the workbook
workbook = xlsxwriter.Workbook(xls_file)

# add worksheet to workbook
worksheet = workbook.add_worksheet()

row_ = 0
col_ = 0

# write headers
worksheet.write(row_, col_, 'NAME')
col_ += 1
worksheet.write(row_, col_, 'VALUE')
row_ += 1

# write sample data
for item in chart_data:
    col_ = 0
    worksheet.write(row_, col_, item['name'])
    col_ += 1
    worksheet.write(row_, col_, item['value'])
    row_ += 1

# create pie chart
pie_chart = workbook.add_chart({'type': 'pie'})

# add series to pie chart
pie_chart.add_series({
    'name': 'Series Name',
    'categories': '=Sheet1!$A$3:$A$%s' % row_,
    'values': '=Sheet1!$B$3:$B$%s' % row_,
    'marker': {'type': 'circle'}
})
# insert pie chart
worksheet.insert_chart('D2', pie_chart)

# create column chart
column_chart = workbook.add_chart({'type': 'column'})

# add serie to column chart
column_chart.add_series({
    'name': 'Series Name',
    'categories': '=Sheet1!$A$3:$A$%s' % row_,
    'values': '=Sheet1!$B$3:$B$%s' % row_,
    'marker': {'type': 'circle'}
})
# insert column chart
worksheet.insert_chart('D20', column_chart)

workbook.close()
```

xlrdexcel

Python xlrdMicrosoft Exceltm。

-

```
pip install xlrd
```

pypisetup.py

https://pypi.python.org/pypi/xlrd

Excel -open_workbookxlrdexcel。

```
pip install xlrd
```

excel

```
pip install xlrd
```

```
pip install xlrd
```

```
pip install xlrd
```

```
pip install xlrd
```

Excel

```
pip install xlrd
```

Excel

```
pip install xlrd
```

xlsxwriterExcel

```
import xlsxwriter

# create a new file
workbook = xlsxwriter.Workbook('your_file.xlsx')

# add some new formats to be used by the workbook
percent_format = workbook.add_format({'num_format': '0%'})
percent_with_decimal = workbook.add_format({'num_format': '0.0%'})
bold = workbook.add_format({'bold': True})
red_font = workbook.add_format({'font_color': 'red'})
remove_format = workbook.add_format()

# add a new sheet
worksheet = workbook.add_worksheet()

# set the width of column A
worksheet.set_column('A:A', 30, )

# set column B to 20 and include the percent format we created earlier
worksheet.set_column('B:B', 20, percent_format)

# remove formatting from the first row (change in height=None)
worksheet.set_row('0:0', None, remove_format)

workbook.close()
```

PythonExcel https://riptutorial.com/zh-TW/python/topic/2986/pythonexcel

46: Python

- pickle.dumpobjfileprotocol = None*fix_imports = True
- pickle.loadfile*fix_imports = Trueencoding =“ASCII”errors =“strict”

OBJ	obj
	pickler `0` -ASCII `1` -
	file*dump*write`wb`read`rb`

Examples

Python

。

pickle。

pickle`wb`rb`pickle。

```
data={'a':'some_value',
     'b':[9,4,7],
     'c':['some_str','another_str','spam','ham'],
     'd':{'key':'nested_dictionary'},
     }
```

```
data={'a':'some_value',
     'b':[9,4,7],
     'c':['some_str','another_str','spam','ham'],
     'd':{'key':'nested_dictionary'},
     }
```

```
data={'a':'some_value',
     'b':[9,4,7],
     'c':['some_str','another_str','spam','ham'],
     'd':{'key':'nested_dictionary'},
     }
```

1.
2.
3.
4.
5. deflambda
6.
7.
8.

dictgetstate

```
import pickle
def save(filename,object):
    file=open(filename,'wb')
    pickle.dump(object,file)
    file.close()

def load(filename):
    file=open(filename,'rb')
    object=pickle.load(file)
    file.close()
    return object


>>>list_object=[1,1,2,3,5,8,'a','e','i','o','u']
>>>save(list_file,list_object)
>>>new_list=load(list_file)
>>>new_list
[1, 1, 2, 3, 5, 8, 'a', 'e', 'i', 'o', 'u'
```

Python https://riptutorial.com/zh-TW/python/topic/7810/python

47: Python

。 Python。 。

Examples

Python。 Intfloatcomplexlong。

```
int_num = 10    #int value
float_num = 10.2    #float value
complex_num = 3.14j    #complex value
long_num = 1234567L    #long value
```

。 Python。 。

```
a_str = 'Hello World'
print(a_str)    #output will be whole string. Hello World
print(a_str[0])    #output will be first character. H
print(a_str[0:5])    #output will be first five characters. Hello
```

[]。 C。 。

```
list = [123,'abcd',10.2,'d']    #can be a array of any data type or single data type.
list1 = ['hello','world']
print(list)    #will ouput whole list. [123,'abcd',10.2,'d']
print(list[0:2])    #will output first two element of list. [123,'abcd']
print(list1 * 2)    #will gave list1 two times. ['hello','world','hello','world']
print(list + list1)    #will gave concatenation of both the lists.
[123,'abcd',10.2,'d','hello','world']
```

[]。 。

```
tuple = (123,'hello')
tuple1 = ('world')
print(tuple)    #will output whole tuple. (123,'hello')
print(tuple[0])    #will output first value. (123)
print(tuple + tuple1)    #will output (123,'hello','world')
tuple[1]='update'    #this will give you error.
```

。 {}[]。

```
dic={'name':'red','age':10}
print(dic)    #will output all the key-value pairs. {'name':'red','age':10}
print(dic['name'])    #will output only value with 'name' key. 'red'
print(dic.values())    #will output list of values in dic. ['red',10]
print(dic.keys())    #will output list of keys. ['name','age']
```

1. -

```
basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}
print(basket)             # duplicates will be removed
> {'orange', 'banana', 'pear', 'apple'}
a = set('abracadabra')
print(a)                  # unique letters in a
> {'a', 'r', 'b', 'c', 'd'}
a.add('z')
print(a)
> {'a', 'c', 'r', 'b', 'z', 'd'}
```

2. - 。

```
basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}
print(basket)             # duplicates will be removed
> {'orange', 'banana', 'pear', 'apple'}
a = set('abracadabra')
print(a)                  # unique letters in a
> {'a', 'r', 'b', 'c', 'd'}
a.add('z')
print(a)
> {'a', 'c', 'r', 'b', 'z', 'd'}
```

Python https://riptutorial.com/zh-TW/python/topic/9366/python

48: Python

SSEWeb“”。 websockets。 SSEwebsocketsSSEwebsockets。 SSEwebsockets/。

Examples

SSE

```
@route("/stream")
def stream():
    def event_stream():
        while True:
            if message_to_send:
                yield "data:
                    {}\n\n".format(message_to_send)"

    return Response(event_stream(), mimetype="text/event-stream")
```

Asyncio SSE

asyncio SSE https //github.com/brutasse/asyncio-sse

```
import asyncio
import sse

class Handler(sse.Handler):
    @asyncio.coroutine
    def handle_request(self):
        yield from asyncio.sleep(2)
        self.send('foo')
        yield from asyncio.sleep(2)
        self.send('bar', event='wakeup')

start_server = sse.serve(Handler, 'localhost', 8888)
asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

Python https://riptutorial.com/zh-TW/python/topic/9100/python

49: PythonCurses

CursesPython。 TUI。

C'ncurses'python

Examples

```
import curses
import traceback

try:
    # -- Initialize --
    stdscr = curses.initscr()   # initialize curses screen
    curses.noecho()             # turn off auto echoing of keypress on to screen
    curses.cbreak()             # enter break mode where pressing Enter key
                                #  after keystroke is not required for it to register
    stdscr.keypad(1)            # enable special Key values such as curses.KEY_LEFT etc

    # -- Perform an action with Screen --
    stdscr.border(0)
    stdscr.addstr(5, 5, 'Hello from Curses!', curses.A_BOLD)
    stdscr.addstr(6, 5, 'Press q to close this screen', curses.A_NORMAL)

    while True:
        # stay in this loop till the user presses 'q'
        ch = stdscr.getch()
        if ch == ord('q'):
            break

    # -- End of user code --

except:
    traceback.print_exc()     # print trace back log of the error

finally:
    # --- Cleanup on exit ---
    stdscr.keypad(0)
    curses.echo()
    curses.nocbreak()
    curses.endwin()
```

wrapper。

curses`wrapper(func, ...)`。

```
main(scr, *args):
    # -- Perform an action with Screen --
    scr.border(0)
    scr.addstr(5, 5, 'Hello from Curses!', curses.A_BOLD)
    scr.addstr(6, 5, 'Press q to close this screen', curses.A_NORMAL)

    while True:
        # stay in this loop till the user presses 'q'
```

```
        ch = scr.getch()
        if ch == ord('q'):

curses.wrapper(main)
```

wrappercurses`stdscr` WindowObjectstdscr`func` ◦ `func wrapper`◦

PythonCurses https://riptutorial.com/zh-TW/python/topic/5851/pythoncurses

50: Python

Examples

PythonBig-O。

```
def list_check(to_check, the_list):
    for item in the_list:
        if to_check == item:
          return True
    return False
```

。 On。 “O”OOrder。

On - n

Ok - k

O1

On

DelOn

On

On

O1

O1

On

Ok

On + k

Ok

On log n

Onk

x inOn

minsmaxsOn

O1

Deque

。

```
class Deque:
def __init__(self):
    self.items = []

def isEmpty(self):
    return self.items == []

def addFront(self, item):
    self.items.append(item)

def addRear(self, item):
    self.items.insert(0,item)

def removeFront(self):
    return self.items.pop()

def removeRear(self):
    return self.items.pop(0)

def size(self):
    return len(self.items)
```

O1

O1

On

Ok

ExtendleftOk

O1

PopleftO1

On

Ok

xxO1

s - tOlens

stOminlenslentOlens* lent

s1s2s3...sn ::n-1* Ollmaxlens1...lensn

s.difference_updatetOlentOlent* lens

s.symetric_difference_updatetOlent

s ^ tOlensOlens* lent

s | tOlens+ lent

Python。 。 。

80/20 802090/10 - 。 80。

“”“” 。

On log nOn2。 Olog nO1PythonOn。

... Python

3。

1. **Θ** theta。 Theta。 。 3n3 + 6n2 + 6000 =Θn3n0Θn3Θn2。 gnΘgn。 Θgn= {fnc1c2n00 <= c1 *gn<= fn<= c2* gnn> = n0}fngnθnfnc1 *gnc2* gnn> = n0。 θn0nfn。

2. **Big O** Big O。 。 。 On ^ 2。 On ^ 2。 Θ

1. Θn ^ 2。
2. Θn。

Big O。 。 Ogn= {fncn0n> = n00 <= fn<= cgn

3. **Ω** Big OΩ。 Ω<。 Omega。 gnΩgn。 Ωgn= {fncn0n> = n0}0 <= cgn<= fn。 。 Ωn。

Python https://riptutorial.com/zh-TW/python/topic/9185/python

51: Python

Python

Examples

Python -

```
import socket

serversocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
serversocket.bind(('localhost', 8089))
serversocket.listen(5) # become a server socket, maximum 5 connections

while True:
    connection, address = serversocket.accept()
    buf = connection.recv(64)
    if len(buf) > 0:
        print(buf)
    break
```

```
import socket

serversocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
serversocket.bind(('localhost', 8089))
serversocket.listen(5) # become a server socket, maximum 5 connections

while True:
    connection, address = serversocket.accept()
    buf = connection.recv(64)
    if len(buf) > 0:
        print(buf)
    break
```

SocketServer.py/.....;

Http

httpjavascriptPythonSimpleHTTPServer。 PythonPath。

python 2

```
$ python -m SimpleHTTPServer <portnumber>
```

python 3

```
$ python -m SimpleHTTPServer <portnumber>
```

8000。

0.0.0.08000HTTP ...

`http://hostipaddress:8000/`。

`hostipaddress`IP`192.168.xx`。

`ctrl+c`.

TCP

`socketserver`TCP。 echo。

```
from sockerserver import BaseRequestHandler, TCPServer

class EchoHandler(BaseRequestHandler):
    def handle(self):
        print('connection from:', self.client_address)
        while True:
            msg = self.request.recv(8192)
            if not msg:
                break
            self.request.send(msg)

if __name__ == '__main__':
    server = TCPServer(('', 5000), EchoHandler)
    server.serve_forever()
```

```
from sockerserver import BaseRequestHandler, TCPServer

class EchoHandler(BaseRequestHandler):
    def handle(self):
        print('connection from:', self.client_address)
        while True:
            msg = self.request.recv(8192)
            if not msg:
                break
            self.request.send(msg)

if __name__ == '__main__':
    server = TCPServer(('', 5000), EchoHandler)
    server.serve_forever()
```

`socketserver`TCP。 。 `ThreadingTCPServer` 。

```
from sockerserver import BaseRequestHandler, TCPServer

class EchoHandler(BaseRequestHandler):
    def handle(self):
        print('connection from:', self.client_address)
        while True:
            msg = self.request.recv(8192)
            if not msg:
                break
            self.request.send(msg)

if __name__ == '__main__':
```

```
    server = TCPServer(('', 5000), EchoHandler)
    server.serve_forever()
```

UDP

`socketserver`UDP。

```
import time
from socketserver import BaseRequestHandler, UDPServer

class CtimeHandler(BaseRequestHandler):
    def handle(self):
    print('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__':
    server = UDPServer(('', 5000), CtimeHandler)
    server.serve_forever()
```

```
import time
from socketserver import BaseRequestHandler, UDPServer

class CtimeHandler(BaseRequestHandler):
    def handle(self):
    print('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__':
    server = UDPServer(('', 5000), CtimeHandler)
    server.serve_forever()
```

Simple HttpServer

。

```
from http.server import HTTPServer, CGIHTTPRequestHandler
import webbrowser
import threading

def start_server(path, port=8000):
    '''Start a simple webserver serving path on port'''
    os.chdir(path)
    httpd = HTTPServer(('', port), CGIHTTPRequestHandler)
    httpd.serve_forever()

# Start the server in a new thread
port = 8000
daemon = threading.Thread(name='daemon_server',
                          target=start_server,
                          args=('.', port)
```

```
daemon.setDaemon(True) # Set as a daemon so it will be killed once the main thread is dead.
daemon.start()

# Open the web browser
webbrowser.open('http://localhost:{}'.format(port))
```

Python https://riptutorial.com/zh-TW/python/topic/1309/python

52: Python - virtualenv

“virtualenv”Python。 Python env。 “A2.xxxB2.xxx”。

“virtualenv”Python。

Examples

pip /apt-getvirtualenv

```
pip install virtualenv
```


```
pip install virtualenv
```


sudo。

```
$ cd test_proj
```


```
$ cd test_proj
```

```
$ cd test_proj
```


virtualenv“deactivate”


```
$ cd test_proj
```


Virtualenv

virtualenvbineasy_installeggvirtualenvsite-packages。

```
$ source test_project/bin/activate
$ pip install flask
```

sudovirtualenv site-packages。 。

virtualenv

lsvirtualenv 。

cdvirtualenv 。

cdsitepackages site-packages。

lssitepackages site-packages。

Python - virtualenv https://riptutorial.com/zh-TW/python/topic/9782/python----virtualenv

53: Python

HTTP POSTRequestsPython Requests

Examples

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

HTTP POST。 。

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

SSL

SSL。

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

httphttps

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

```
from requests import post

foo = post('http://httpbin.org/post', data = {'key':'value'})
```

```
from requests import post
```

```
foo = post('http://httpbin.org/post', data = {'key':'value'})
```

```
from requests import post

payload = {'key1' : 'value1',
           'key2' : 'value2'
           }

foo = post('http://httpbin.org/post', data=payload)
```

post。

data。

json

```
from requests import post

payload = {'key1' : 'value1',
           'key2' : 'value2'
           }

foo = post('http://httpbin.org/post', data=payload)
```

Requests.`read()`

```
from requests import post

files = {'file' : open('data.txt', 'rb')}

foo = post('http://http.org/post', files=files)
```

Filenamecontent_typeheaders

```
from requests import post

files = {'file' : open('data.txt', 'rb')}

foo = post('http://http.org/post', files=files)
```

`files`。

```
from requests import post

files = {'file' : open('data.txt', 'rb')}

foo = post('http://http.org/post', files=files)
```

```
from requests import post

foo = post('http://httpbin.org/post', data={'data' : 'value'})
print(foo.status_code)
```

```
from requests import post

foo = post('http://httpbin.org/post', data={'data' : 'value'})
print(foo.status_code)
```

urllib3 response.HTTPResponse

```
from requests import post

foo = post('http://httpbin.org/post', data={'data' : 'value'})
print(foo.status_code)
```

HTTP

HTTP

```
from requests import post

foo = post('http://natas0.natas.labs.overthewire.org', auth=('natas0', 'natas0'))
```

```
from requests import post

foo = post('http://natas0.natas.labs.overthewire.org', auth=('natas0', 'natas0'))
```

HTTP

HTTP

```
from requests import post

foo = post('http://natas0.natas.labs.overthewire.org', auth=('natas0', 'natas0'))
```

HTTP。AuthBaseRequests

```
from requests import post

foo = post('http://natas0.natas.labs.overthewire.org', auth=('natas0', 'natas0'))
```

```
from requests import post

foo = post('http://natas0.natas.labs.overthewire.org', auth=('natas0', 'natas0'))
```

POST

HTTP / S

```
from requests import post

proxies = {
  'http': 'http://192.168.0.128:3128',
  'https': 'http://192.168.0.127:1080',
```

```
   }

foo = requests.post('http://httpbin.org/post', proxies=proxies)
```

HTTP

```
from requests import post

proxies = {
  'http': 'http://192.168.0.128:3128',
  'https': 'http://192.168.0.127:1080',
   }

foo = requests.post('http://httpbin.org/post', proxies=proxies)
```

SOCKS

socks`requests[socks]` socksHTTPBasicAuth

```
from requests import post

proxies = {
  'http': 'http://192.168.0.128:3128',
  'https': 'http://192.168.0.127:1080',
   }

foo = requests.post('http://httpbin.org/post', proxies=proxies)
```

Python https://riptutorial.com/zh-TW/python/topic/10021/python

54: setup.py

name	。
version	。
packages	Python。 setuptools.find_packages()。
py_modules	Python.py。

python

。

Examples

setup.py

Distutils。 。

foofoo.py

```
from distutils.core import setup

setup(name='foo',
      version='1.0',
      py_modules=['foo'],
      )
```

setup.py

```
from distutils.core import setup

setup(name='foo',
      version='1.0',
      py_modules=['foo'],
      )
```

sdistUnixtarballWindowsZIPsetup.pyfoo.py.foo-1.0.tar.gz.zipfoo-1.0。

foofoo-1.0.tar.gz.zipfoo-1.0

```
from distutils.core import setup

setup(name='foo',
      version='1.0',
      py_modules=['foo'],
```

```
      )
```

python

python。 。

`greetingshello_world.py` 。

```
greetings/
   greetings/
      __init__.py
      hello_world.py
```

```
greetings/
   greetings/
      __init__.py
      hello_world.py
```

```
greetings/
   greetings/
      __init__.py
      hello_world.py
```

`setup.py` `setup()` `scripts`

```
greetings/
   greetings/
      __init__.py
      hello_world.py
```

`hello_world.py`。

```
greetings/
   greetings/
      __init__.py
      hello_world.py
```

```
greetings/
   greetings/
      __init__.py
      hello_world.py
```

setup.py

`setuptools_scm`GitMercurialPython。

```
from setuptools import setup, find_packages

setup(
    setup_requires=['setuptools_scm'],
```

```
    use_scm_version=True,
    packages=find_packages(),
    include_package_data=True,
)
```

;SCM`find_packages()`` use_scm_version=True`。

```
python setup.py install
```

。

```
python setup.py install
```

Sphinxfortran

```
python setup.py install
```

`initialize_options`` finalize_options`` run`。

```
python setup.py install
```

setup.py https://riptutorial.com/zh-TW/python/topic/1444/setup-py

55: Sqlite3

Examples

Sqlite3 - 。

sqlite3GerhardHäring。 Connection。 example.db

```
import sqlite3
conn = sqlite3.connect('example.db')
```

memoryRAM。 ConnectionCursorexecuteSQL

```
import sqlite3
conn = sqlite3.connect('example.db')
```

SQLite3。

select

```
import sqlite3
conn = sqlite3.connect('example.db')
c = conn.cursor()
c.execute("SELECT * from table_name where id=cust_id")
for row in c:
    print row # will be a list
```

fetchone

```
import sqlite3
conn = sqlite3.connect('example.db')
c = conn.cursor()
c.execute("SELECT * from table_name where id=cust_id")
for row in c:
    print row # will be a list
```

fetchall

```
import sqlite3
conn = sqlite3.connect('example.db')
c = conn.cursor()
c.execute("SELECT * from table_name where id=cust_id")
for row in c:
    print row # will be a list
```

sqlite3.Error

```
import sqlite3
conn = sqlite3.connect('example.db')
```

```
c = conn.cursor()
c.execute("SELECT * from table_name where id=cust_id")
for row in c:
    print row # will be a list
```

Sqlite3 https://riptutorial.com/zh-TW/python/topic/7754/sqlite3

56: SYS

sys`sys.argvsys.exit()`。

。

- sys

```
import sys
```

- sys

```
import sys
```

sys 。

Examples

```
if len(sys.argv) != 4:          # The script name needs to be accounted for as well.
    raise RuntimeError("expected 3 command line arguments")

f = open(sys.argv[1], 'rb')     # Use first command line argument.
start_line = int(sys.argv[2])   # All arguments come as strings, so need to be
end_line = int(sys.argv[3])     # converted explicitly if other types are required.
```

click。

```
# The name of the executed script is at the beginning of the argv list.
print('usage:', sys.argv[0], '<filename> <start> <end>')

# You can use it to generate the path prefix of the executed program
# (as opposed to the current module) to access files relative to that,
# which would be good for assets of a game, for instance.
program_file = sys.argv[0]

import pathlib
program_path = pathlib.Path(program_file).resolve().parent
```

```
# Error messages should not go to standard output, if possible.
print('ERROR: We have no cheese at all.', file=sys.stderr)

try:
    f = open('nonexistent-file.xyz', 'rb')
except OSError as e:
    print(e, file=sys.stderr)
```

```
def main():
    if len(sys.argv) != 4 or '--help' in sys.argv[1:]:
        print('usage: my_program <arg1> <arg2> <arg3>', file=sys.stderr)
```

```
        sys.exit(1)    # use an exit code to signal the program was unsuccessful

    process_data()
```

SYS https://riptutorial.com/zh-TW/python/topic/9847/sys

57: tempfile NamedTemporaryFile

PARAM	
	= w + b
	= True
	="
	='tmp'
DIR	tempfiledirname= None
BUFFSIZE	default = -1

Examples

`name`。 unix`delete` paramTrue。

“Hello World”。 `namepath`。 。

```
import tempfile

with tempfile.NamedTemporaryFile(delete=False) as t:
    t.write('Hello World!')
    path = t.name
    print path

with open(path) as t:
    print t.read()
```


```
import tempfile

with tempfile.NamedTemporaryFile(delete=False) as t:
    t.write('Hello World!')
    path = t.name
    print path

with open(path) as t:
    print t.read()
```


tempfile NamedTemporaryFile https://riptutorial.com/zh-TW/python/topic/3666/tempfile-namedtemporaryfile

58: Tkinter

TkinterPythonGUI。 。

tkinterPython 23。 Python 2

```
from Tkinter import *  # Capitalized
```

Python 3

```
from Tkinter import *  # Capitalized
```

Python 23

```
from Tkinter import *  # Capitalized
```

```
from Tkinter import *  # Capitalized
```

tkinter

Examples

tkinter

`tkinter`GUITk / Tcl GUIPython。 `tkinter`。

Python 2“”。

```
import tkinter as tk

# GUI window is a subclass of the basic tkinter Frame object
class HelloWorldFrame(tk.Frame):
    def __init__(self, master):
        # Call superclass constructor
        tk.Frame.__init__(self, master)
        # Place frame into main window
        self.grid()
        # Create text box with "Hello World" text
        hello = tk.Label(self, text="Hello World! This label can hold strings!")
        # Place text box into frame
        hello.grid(row=0, column=0)

# Spawn window
if __name__ == "__main__":
    # Create main window object
    root = tk.Tk()
    # Set title of window
    root.title("Hello World!")
    # Instantiate HelloWorldFrame object
```

```
    hello_frame = HelloWorldFrame(root)
    # Start GUI
    hello_frame.mainloop()
```

Tkinter `place` `packgrid` ◦

`place`◦

`pack`4◦ ◦

`grid`◦

`widget.place`

- `x` x
- `y` y
- `height`
- `width`

`place`

```
class PlaceExample(Frame):
    def __init__(self,master):
        Frame.__init__(self,master)
        self.grid()
        top_text=Label(master,text="This is on top at the origin")
        #top_text.pack()
        top_text.place(x=0,y=0,height=50,width=200)
        bottom_right_text=Label(master,text="This is at position 200,400")
        #top_text.pack()
        bottom_right_text.place(x=200,y=400,height=50,width=200)
# Spawn Window
if __name__=="__main__":
    root=Tk()
    place_frame=PlaceExample(root)
    place_frame.mainloop()
```

`widget.pack`

- `expand`
- `fill` NONEXYBOTH
- `side` TOPBOTTOMLEFT

`widget.grid`

- `row`
- `rowspan` colums1
- `column` 0
- `columnspan` 1

- `sticky` NNEESESSWWNW

◦ ◦

`grid`

```
class PlaceExample(Frame):
    def __init__(self,master):
        Frame.__init__(self,master)
        self.grid()
        top_text=Label(master,text="This is on top at the origin")
        #top_text.pack()
        top_text.place(x=0,y=0,height=50,width=200)
        bottom_right_text=Label(master,text="This is at position 200,400")
        #top_text.pack()
        bottom_right_text.place(x=200,y=400,height=50,width=200)
# Spawn Window
if __name__=="__main__":
    root=Tk()
    place_frame=PlaceExample(root)
    place_frame.mainloop()
```

packgrid

Tkinter https://riptutorial.com/zh-TW/python/topic/7574/tkinter

59: Unicode

- str.encodeencodingerrors ='strict'
- bytes.decodeencodingerrors ='strict'
- =

	`'ascii' 'utf8'...`
	`'replace' 'ignore'.....`

Examples

Python 3 `str`unicode`bytes`。

```
type("f") == type(u"f")  # True, <class 'str'>
type(b"f")               # <class 'bytes'>
```

Python 2unicode“u”。

```
type("f") == type(u"f")  # True, <class 'str'>
type(b"f")               # <class 'bytes'>
```

Unicode

`.encode(encoding)`Unicode。

Python 3

```
type("f") == type(u"f")  # True, <class 'str'>
type(b"f")               # <class 'bytes'>
```

Python 2

py2`sys.getdefaultencoding() == 'ascii'`py3`utf-8` 。

```
type("f") == type(u"f")  # True, <class 'str'>
type(b"f")               # <class 'bytes'>
```

“UnicodeEncodeError”

```
type("f") == type(u"f")  # True, <class 'str'>
```

```
type(b"f")                # <class 'bytes'>
```

unicode

`.decode(encoding)` unicode。

unicode

```
type("f") == type(u"f")  # True, <class 'str'>
type(b"f")               # <class 'bytes'>
```

`UnicodeDecodeError`

```
type("f") == type(u"f")  # True, <class 'str'>
type(b"f")               # <class 'bytes'>
```

/

`.encode.decode`。

`'strict'` 。 。

```
>>> "£13.55".encode('ascii', errors='replace')
b'?13.55'
>>> "£13.55".encode('ascii', errors='ignore')
b'13.55'
>>> "£13.55".encode('ascii', errors='namereplace')
b'\\N{POUND SIGN}13.55'
>>> "£13.55".encode('ascii', errors='xmlcharrefreplace')
b'&#163;13.55'
>>> "£13.55".encode('ascii', errors='backslashreplace')
b'\\xa313.55'
```


```
>>> "£13.55".encode('ascii', errors='replace')
b'?13.55'
>>> "£13.55".encode('ascii', errors='ignore')
b'13.55'
>>> "£13.55".encode('ascii', errors='namereplace')
b'\\N{POUND SIGN}13.55'
>>> "£13.55".encode('ascii', errors='xmlcharrefreplace')
b'&#163;13.55'
>>> "£13.55".encode('ascii', errors='backslashreplace')
b'\\xa313.55'
```

unicode。

I / O.

`'r''w'` 。 deafult`'utf8'` 。

```
open(fn, mode='r')                      # opens file for reading in utf8
open(fn, mode='r', encoding='utf16')  # opens file for reading utf16

# ERROR: cannot write bytes when a string is expected:
open("foo.txt", "w").write(b"foo")
```

`'rb''wb'` 。 。


```
open(fn, mode='r')                      # opens file for reading in utf8
open(fn, mode='r', encoding='utf16')  # opens file for reading utf16

# ERROR: cannot write bytes when a string is expected:
open("foo.txt", "w").write(b"foo")
```


Unicode https://riptutorial.com/zh-TW/python/topic/1216/unicode

60: WebWSGI

start_response

Examples

'application'。 。

```
import os, sys

def run(application):
    environ['wsgi.input']        = sys.stdin
    environ['wsgi.errors']       = sys.stderr

    headers_set = []
    headers_sent = []

    def write (data):
        """
        Writes header data from 'start_response()' as well as body data from 'response'
        to system standard output.
        """
        if not headers_set:
            raise AssertionError("write() before start_response()")

        elif not headers_sent:
            status, response_headers = headers_sent[:] = headers_set
            sys.stdout.write('Status: %s\r\n' % status)
            for header in response_headers:
                sys.stdout.write('%s: %s\r\n' % header)
            sys.stdout.write('\r\n')

        sys.stdout.write(data)
        sys.stdout.flush()

    def start_response(status, response_headers):
        """ Sets headers for the response returned by this server."""
        if headers_set:
            raise AssertionError("Headers already set!")

        headers_set[:] = [status, response_headers]
        return write

    # This is the most important piece of the 'server object'
    # Our result will be generated by the 'application' given to this method as a parameter
    result = application(environ, start_response)
    try:
        for data in result:
            if data:
                write(data)          # Body isn't empty send its data to 'write()'
        if not headers_sent:
            write('')                # Body is empty, send empty string to 'write()'
```

WebWSGI https://riptutorial.com/zh-TW/python/topic/5315/web-wsgi-

61: Web

PythonwebbrowserWeb。webbrowser。

- `webbrowser.open(url, new=0, autoraise=False)`
- `webbrowser.open_new(url)`
- `webbrowser.open_new_tab(url)`
- `webbrowser.get(usage=None)`
- `webbrowser.register(name, constructor, instance=None)`

`webbrowser.open()`	
	WebURL
	0URL12
autoraise	True
`webbrowser.open_new()`	
	WebURL
`webbrowser.open_new_tab()`	
	WebURL
`webbrowser.get()`	
`webbrowser.register()`	
	`webbrowser.get()`Web

。 `webbrowser.get()`。

`'mozilla'`	`Mozilla('mozilla')`
`'firefox'`	`Mozilla('mozilla')`
`'netscape'`	`Mozilla('netscape')`
`'galeon'`	`Galeon('galeon')`
`'epiphany'`	`Galeon('epiphany')`
`'skipstone'`	`BackgroundBrowser('skipstone')`
`'kfmclient'`	`Konqueror()`
`'konqueror'`	`Konqueror()`

'kfm'	Konqueror()
'mosaic'	BackgroundBrowser('mosaic')
'opera'	Opera()
'grail'	Grail()
'links'	GenericBrowser('links')
'elinks'	Elinks('elinks')
'lynx'	GenericBrowser('lynx')
'w3m'	GenericBrowser('w3m')
'windows-default'	WindowsDefault
'macosx'	MacOSX('default')
'safari'	MacOSX('safari')
'google-chrome'	Chrome('google-chrome')
'chrome'	Chrome('chrome')
'chromium'	Chromium('chromium')
'chromium-browser'	Chromium('chromium-browser')

Examples

URL

URL`webbrowser.open()`

```
import webbrowser
webbrowser.open("http://stackoverflow.com")
```

URL。URL。 open

- `url` - WebURL **[]**
- `new` - 012 **[0]**
- `autoraise` - True **[False]**

`newautoraise`。

Webbrowser`open_new`URL

```
import webbrowser
webbrowser.open("http://stackoverflow.com")
```

URL。 `open_new_tab`

```
import webbrowser
webbrowser.open("http://stackoverflow.com")
```

URL

webbrowser`register()get()`。 get。

```
import webbrowser
ff_path = webbrowser.get("C:/Program Files/Mozilla Firefox/firefox.exe")
ff = webbrowser.get(ff_path)
ff.open("http://stackoverflow.com/")
```


```
import webbrowser
ff_path = webbrowser.get("C:/Program Files/Mozilla Firefox/firefox.exe")
ff = webbrowser.get(ff_path)
ff.open("http://stackoverflow.com/")
```


Web https://riptutorial.com/zh-TW/python/topic/8676/web

62: “with”Statement

Python。 。 Python。

- “context_manager”“”“context_manager”“”*

PEP 343。 `try ... finally`。 。

PEP`__exit__()` `__enter__()``__exit__()`with。

`with`。

```
with EXPR as VAR:
    BLOCK
```

```
with EXPR as VAR:
    BLOCK
```

Examples

with

。 `with`。 。

。

```
open_file = open(filename)
with open_file:
    file_contents = open_file.read()

# the open_file object has automatically been closed.
```

`as`

```
open_file = open(filename)
with open_file:
    file_contents = open_file.read()

# the open_file object has automatically been closed.
```

exit。 。 /。 。 Python 2.5。

。 `with`。

`with`

```
with database_connection as cursor:
    cursor.execute(sql_query)
```

```
with database_connection as cursor:
    cursor.execute(sql_query)
```

__enter__()__exit__()

```
class AContextManager():

    def __enter__(self):
        print("Entered")
        # optionally return an object
        return "A-instance"

    def __exit__(self, exc_type, exc_value, traceback):
        print("Exited" + (" (with an exception)" if exc_type else ""))
        # return True if you want to suppress the exception
```

exc_type exc_value tracebacksys.exc_info()。 None 。

__exit__True__exit__。

```
class AContextManager():

    def __enter__(self):
        print("Entered")
        # optionally return an object
        return "A-instance"

    def __exit__(self, exc_type, exc_value, traceback):
        print("Exited" + (" (with an exception)" if exc_type else ""))
        # return True if you want to suppress the exception
```

with__exit__。

__exit__

```
class AContextManager():

    def __enter__(self):
        print("Entered")
        # optionally return an object
        return "A-instance"

    def __exit__(self, exc_type, exc_value, traceback):
        print("Exited" + (" (with an exception)" if exc_type else ""))
        # return True if you want to suppress the exception
```

contextlib.contextmanager

```
import contextlib

@contextlib.contextmanager
def context_manager(num):
    print('Enter')
    yield num + 1
    print('Exit')
```

```
with context_manager(2) as cm:
    # the following instructions are run when the 'yield' point of the context
    # manager is reached.
    # 'cm' will have the value that was yielded
    print('Right in the middle with cm = {}'.format(cm))
```

```
import contextlib

@contextlib.contextmanager
def context_manager(num):
    print('Enter')
    yield num + 1
    print('Exit')

with context_manager(2) as cm:
    # the following instructions are run when the 'yield' point of the context
    # manager is reached.
    # 'cm' will have the value that was yielded
    print('Right in the middle with cm = {}'.format(cm))
```

- yield `__enter__` withyield `__exit__`。

`try..except..finally` -`with` -。

```
import contextlib

@contextlib.contextmanager
def context_manager(num):
    print('Enter')
    yield num + 1
    print('Exit')

with context_manager(2) as cm:
    # the following instructions are run when the 'yield' point of the context
    # manager is reached.
    # 'cm' will have the value that was yielded
    print('Right in the middle with cm = {}'.format(cm))
```

```
import contextlib

@contextlib.contextmanager
def context_manager(num):
    print('Enter')
    yield num + 1
    print('Exit')

with context_manager(2) as cm:
    # the following instructions are run when the 'yield' point of the context
    # manager is reached.
    # 'cm' will have the value that was yielded
    print('Right in the middle with cm = {}'.format(cm))
```

```
with open(input_path) as input_file, open(output_path, 'w') as output_file:

    # do something with both files.
```

```
    # e.g. copy the contents of input_file into output_file
    for line in input_file:
        output_file.write(line + '\n')
```

```
with open(input_path) as input_file, open(output_path, 'w') as output_file:

    # do something with both files.

    # e.g. copy the contents of input_file into output_file
    for line in input_file:
        output_file.write(line + '\n')
```

```
class File():
    def __init__(self, filename, mode):
        self.filename = filename
        self.mode = mode

    def __enter__(self):
        self.open_file = open(self.filename, self.mode)
        return self.open_file

    def __exit__(self, *args):
        self.open_file.close()
```

`__init__()`。 `__enter__()` `__exit__()`。

`__enter__` `__exit__` with`with`。

File

```
class File():
    def __init__(self, filename, mode):
        self.filename = filename
        self.mode = mode

    def __enter__(self):
        self.open_file = open(self.filename, self.mode)
        return self.open_file

    def __exit__(self, *args):
        self.open_file.close()
```

“with”Statement https://riptutorial.com/zh-TW/python/topic/928/--with-statement-

63: Python 2Python 3

Python。 2008Python 3。 Python 2Python 3。

Python2.7Python 23.6Python 3。 3.33.4。

Python 2.7Python1.x2.xPythonPython。 。 CPython。 CPython2020。

Python Enhancement Proposal 373 2016625Python 22020。 2020Python2.

Python 3。 Python 3。 。

Python 3.0Python 3Python 2.6Python 3Python 2。 PythonPython 2Python 3。

```
from __future__ import print_function
# other imports and instructions go after __future__
print('Hello world')
```

`__future__`Python 。

2to3PythonPython 2.xPython 3.xPython 。

sixPython 2/3

-
- / unicode
-

Python 2Python 3。

Examples

Python 2 `print`

Python 2.x 2.7

```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
print >>sys.stderr, "Error"     # print to stderr
print("hello")                  # print "hello", since ("hello") == "hello"
print()                         # print an empty tuple "()"
print 1, 2, 3                   # print space-separated arguments: "1 2 3"
print(1, 2, 3)                  # print tuple "(1, 2, 3)"
```

Python 3 `print()`

Python 3.x 3.0

```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
print >>sys.stderr, "Error"     # print to stderr
print("hello")                  # print "hello", since ("hello") == "hello"
print()                         # print an empty tuple "()"
print 1, 2, 3                   # print space-separated arguments: "1 2 3"
print(1, 2, 3)                  # print tuple "(1, 2, 3)"
```

```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
print >>sys.stderr, "Error"     # print to stderr
print("hello")                  # print "hello", since ("hello") == "hello"
print()                         # print an empty tuple "()"
print 1, 2, 3                   # print space-separated arguments: "1 2 3"
print(1, 2, 3)                  # print tuple "(1, 2, 3)"
```


`sep`。


```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
print >>sys.stderr, "Error"     # print to stderr
print("hello")                  # print "hello", since ("hello") == "hello"
print()                         # print an empty tuple "()"
print 1, 2, 3                   # print space-separated arguments: "1 2 3"
print(1, 2, 3)                  # print tuple "(1, 2, 3)"
```


`end`print。


```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
print >>sys.stderr, "Error"     # print to stderr
print("hello")                  # print "hello", since ("hello") == "hello"
print()                         # print an empty tuple "()"
print 1, 2, 3                   # print space-separated arguments: "1 2 3"
print(1, 2, 3)                  # print tuple "(1, 2, 3)"
```



```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
print >>sys.stderr, "Error"     # print to stderr
print("hello")                  # print "hello", since ("hello") == "hello"
print()                         # print an empty tuple "()"
print 1, 2, 3                   # print space-separated arguments: "1 2 3"
print(1, 2, 3)                  # print tuple "(1, 2, 3)"
```


`print` Python 2.6;`print`


```
print "Hello World"
print                           # print a newline
print "No newline",             # add trailing comma to remove newline
```

```
print >>sys.stderr, "Error"   # print to stderr
print("hello")                # print "hello", since ("hello") == "hello"
print()                       # print an empty tuple "()"
print 1, 2, 3                 # print space-separated arguments: "1 2 3"
print(1, 2, 3)                # print tuple "(1, 2, 3)"
```

Python 3`flush`。

PEP 3105 。

Unicode

Python 2.x 2.7

Python 2 `str` `str` `unicode` 。

Python 2 `str`。

。

```
s = 'Cafe'    # type(s) == str
```

`u`*Unicode*

```
s = 'Cafe'    # type(s) == str
```

Unicode

```
s = 'Cafe'    # type(s) == str
```

Unicode

```
s = 'Cafe'    # type(s) == str
```

Python 3.x 3.0

Python 3 `str`Unicode。

```
s = 'Cafe'    # type(s) == str
```

Python 3`bytes` “blob”。 `bencode`

```
s = 'Cafe'    # type(s) == str
```

```
s = 'Cafe'    # type(s) == str
```

Python 3.x 3.3

`u`

Python 2Python 3。 Python 3Unicode`u`

```
s = 'Cafe'    # type(s) == str
```

Python 2Unicode`ur`

```
s = 'Cafe'    # type(s) == str
```

`encode` Python 3 text `str` `encode`bytes。 UTF-8 。

`decode`bytesUnicode

```
s = 'Cafe'    # type(s) == str
```

Python 2.x 2.6

Python 23`bytesunicode`Python 2。 Python 2Python 3Unicode

```
s = 'Cafe'    # type(s) == str
```

Python 3.x 3.0

Python 3`int`

```
s = 'Cafe'    # type(s) == str
```

11

```
s = 'Cafe'    # type(s) == str
```

Python 3unicode Python 2。

```
s = 'Cafe'    # type(s) == str
```

`/` Python 3Python 2。

Python 3`x / y` `__truediv__`。 Python 2 。

	Python 2	Python 3
`3 / 2`	1	1.5
`2 / 3`	0	0.6666666666666666
`-3 / 2`	-2	-1.5

Python 2.2Python 2.7Python 3。

Python 2*float*。 Python 2`02/32 / 3.02.0 / 32.0/3.00.6666666666666666`

	Python 2	Python 3
`3.0 / 2.0`	1.5	1.5
`2 / 3.0`	0.6666666666666666	0.6666666666666666
`-3.0 / 2`	-1.5	-1.5

---

floor division `//` 。 `//` `__floordiv__` 。

	Python 2	Python 3
`3 // 2`	1	1
`2 // 3`	0	0
`-3 // 2`	-2	-2
`3.0 // 2.0`	1.0	1.0
`2.0 // 3`	0.0	0.0
`-3 // 2.0`	-2.0	-2.0

`operator`

```
from operator import truediv, floordiv
assert truediv(10, 8) == 1.25            # equivalent to `/` in Python 3
assert floordiv(10, 8) == 1              # equivalent to `//`
```

。 `/`。 `from __future__ import division`

```
from operator import truediv, floordiv
assert truediv(10, 8) == 1.25            # equivalent to `/` in Python 3
assert floordiv(10, 8) == 1              # equivalent to `//`
```

	Python 2	Python 3
`3 / 2`	1.5	1.5
`2 / 3`	0.6666666666666666	0.6666666666666666
`-3 / 2`	-1.5	-1.5

`from __future__ import division`/`__future__`。

Python `-3 / 2 == -1` 。 。

---

float `from __future__ import division`/float `3 / 2.0 == 1.5` 。 。 `average = sum(items) / len(items)` float。 `float`int。 Python 3`3 / 2 == 1`True。

Python 3PEP 238 。

Simple Math 。

Reduce

Python 2 `reduce`functools2.6Python 3 `reduce`functools 。 `reduce`Python2Python3 `reduce(function_to_reduce, list_to_reduce)`

。 `operator`truediv。

Python 2.x

Python 2.x 2.3

```
>>> my_list = [1, 2, 3, 4, 5]
>>> import operator
>>> reduce(operator.truediv, my_list)
0.008333333333333333
```

Python 3.x

Python 3.x 3.0

```
>>> my_list = [1, 2, 3, 4, 5]
>>> import operator
>>> reduce(operator.truediv, my_list)
0.008333333333333333
```

`from functools import reduce`reduce 。

xrange

Python 2 `range`xrangexrangeindexcount

Python 2.x 2.3

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)
```

```
print(isinstance(xrange(1, 10), xrange))
# Out: True
```

Python 3 `xrangerangerange。 xrange`

Python 3.x 3.0

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)

print(isinstance(xrange(1, 10), xrange))
# Out: True
```

Python 3.2 `range indexcount`

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)

print(isinstance(xrange(1, 10), xrange))
# Out: True
```

Python 2.x 2.3

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)

print(isinstance(xrange(1, 10), xrange))
# Out: True
```

Python 3。 Python 3`list() rangelist()`

Python 3.x 3.0

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)

print(isinstance(xrange(1, 10), xrange))
# Out: True
```

Python 2.xPython 3.x`builtinsfuture`

Python 2.x 2.0

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)

print(isinstance(xrange(1, 10), xrange))
# Out: True
```

Python 3.x 3.0

```
print(range(1, 10))
# Out: [1, 2, 3, 4, 5, 6, 7, 8, 9]

print(isinstance(range(1, 10), list))
# Out: True

print(xrange(1, 10))
# Out: xrange(1, 10)

print(isinstance(xrange(1, 10), xrange))
# Out: True
```

`futurerange`Python `indexcount` Python 3.2+。

Iterables

Python 3.x 3.0

Python 3。 。 。 `list`

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
```

```
# Out: 5
```

*variable variable。 。

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

str

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

date; _year

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

** -

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

Python 3.x 3.5

。 *** Python 3.5 。

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

Python 2.x 2.0

iterable

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

Python 3.x 3.5

** PEP 448

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

。

```
first, second, *tail, last = [1, 2, 3, 4, 5]
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

Python 3.x 3.0

Python 3。 Python 3

```
first, second, *tail, last = [1, 2, 3, 4, 5]
```

```
print(first)
# Out: 1
print(second)
# Out: 2
print(tail)
# Out: [3, 4]
print(last)
# Out: 5
```

PEP 3113 。

Python2, `raiseexcept`

Python 2.x 2.3

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

Python 3, `as`

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

Python 3Python 2.6。

Python 3.x 3.0

Python 3 。

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

`exceptDatabaseError__cause__`。

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

`except`

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

Python 2.x 2.0

Python 2.x;except。

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

Python 3.x 3.3

“”“ `raise from None` ”“ `raise from None`

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

Python 23

```
try:
    raise IOError, "input/output error"
except IOError, exc:
    print exc
```

.next

Python 2`next`

Python 2.x 2.3

```
g = (i for i in range(0, 3))
g.next()  # Yields 0
g.next()  # Yields 1
g.next()  # Yields 2
```

Python 3 `.next.__next__` “”.`nextAttributeError`。 Python 2Python 3`next` 。

Python 3.x 3.0

```
g = (i for i in range(0, 3))
g.next()  # Yields 0
```

```
g.next()  # Yields 1
g.next()  # Yields 2
```

2.6。

Python 2.x 2.3

。 。 None。 intstr tuplelist

```
[1, 2] > 'foo'
# Out: False
(1, 2) > 'foo'
# Out: True
[1, 2] > (1, 2)
# Out: False
100 < [1, 'x'] < 'xyz' < (1, 'x')
# Out: True
```

```
[1, 2] > 'foo'
# Out: False
(1, 2) > 'foo'
# Out: True
[1, 2] > (1, 2)
# Out: False
100 < [1, 'x'] < 'xyz' < (1, 'x')
# Out: True
```

Python 3.x 3.0

```
[1, 2] > 'foo'
# Out: False
(1, 2) > 'foo'
# Out: True
[1, 2] > (1, 2)
# Out: False
100 < [1, 'x'] < 'xyz' < (1, 'x')
# Out: True
```

Python 3

```
[1, 2] > 'foo'
# Out: False
(1, 2) > 'foo'
# Out: True
[1, 2] > (1, 2)
# Out: False
100 < [1, 'x'] < 'xyz' < (1, 'x')
# Out: True
```

strkey。 [' {0-9。

Python 2`raw_input`

Python 2.x 2.3

```
user_input = raw_input()
```

Python 3`input`。

Python 3.x 3.0

```
user_input = raw_input()
```

Python 2 `input` 。 Python 3。 `eval(input())` 。

Python

```
user_input = raw_input()
```

Python 3Python 2`has_key` `has_key` `iter*view*`。 `d.has_key(key)` `key in d`key in d 。

Python 2`keys` `values``items`。 Python 3;

- `len`
-

。

Python 2.7Python 3;`viewkeys` `viewvalues``viewitems` 。 Python 2Python 3

- `d.keys()` Python 2`d.keys()` `d.values()``d.items()``list(d.keys())` `list(d.values())` `list(d.items())`
- `d.iterkeys()` `d.itervalues()``d.iteritems()``iter(d.keys())` `iter(d)` ;`iter(d.values())` `iter(d.items())`
- Python 2.7`d.viewkeys()` `d.viewvalues()``d.viewitems()``d.keys()` `d.values()``d.items()` 。

Python 2 。

```
d = {'a': 0, 'b': 1, 'c': 2, '!': 3}
for key in d.keys():
    if key.isalpha():
        del d[key]
```

Python 3`keys`Python 3`RuntimeError: dictionary changed size during iteration`RuntimeError: dictionary changed size during iteration 。 `for key in list(d)`。

`next()` ;;Python 2`d.iterkeys()` `d.itervalues()``d.iteritems()` `iter(d)` `iter(d.values())` Python 3 `iter(d.values())``iter(d.items())` 。

execPython 3

Python 2 `exec` `exec code [in globals[, locals]]`.Python 3 `exec` `exec(code, [, globals[, locals]])` Python 2`SyntaxError` 。

`print`statement`__future__` import。 `from __future__ import exec_function`Python 2execPython 3`exec`。

Python 2.x 2.3

```
exec 'code'
exec 'code' in global_vars
exec 'code' in global_vars, local_vars
```

Python 3.x 3.0

```
exec 'code'
exec 'code' in global_vars
exec 'code' in global_vars, local_vars
```

Python 2Python 3。

Python 2hasattr

Python 2 `hasattr`False 。

```
class A(object):
    @property
    def get(self):
        raise IOError


class B(object):
    @property
    def get(self):
        return 'get in b'

a = A()
b = B()

print 'a hasattr get: ', hasattr(a, 'get')
# output False in Python 2 (fixed, True in Python 3)
print 'b hasattr get', hasattr(b, 'get')
# output True in Python 2 and Python 3
```

Python3。 Python 2

```
class A(object):
    @property
    def get(self):
        raise IOError


class B(object):
    @property
    def get(self):
        return 'get in b'

a = A()
b = B()
```

```
print 'a hasattr get: ', hasattr(a, 'get')
# output False in Python 2 (fixed, True in Python 3)
print 'b hasattr get', hasattr(b, 'get')
# output True in Python 2 and Python 3
```

`getattr`

```
class A(object):
    @property
    def get(self):
        raise IOError


class B(object):
    @property
    def get(self):
        return 'get in b'

a = A()
b = B()

print 'a hasattr get: ', hasattr(a, 'get')
# output False in Python 2 (fixed, True in Python 3)
print 'b hasattr get', hasattr(b, 'get')
# output True in Python 2 and Python 3
```

_winreg	WinReg
ConfigParser	configparser
copy_reg	copyreg
SocketServer	SocketServer
_markupbase	markupbase
	reprlib
test.test_support	test.support
Tkinter	Tkinter
tkFileDialog	tkinter.filedialog
urllib / urllib2	urlliburllib.parseurllib.errorurllib.responseurllib.requesturllib.robotparser

- tkinterurllib。

Python 2.x3.x`future`Python 2.xPython 3.x。

Python 2

```
>>> 0755  # only Python 2
```


```
>>> 0755  # only Python 2
```


Python 3“”。

Python `3.x` ;python`object` 。 `objectclass`

Python 3.x 3.0

```
class X: pass
class Y(object): pass
```

`mroobject`

Python 3.x 3.0

```
class X: pass
class Y(object): pass
```

Python `2.x`;`object`。 `objectclass`

Python 2.x 2.3

```
class X: pass
class Y(object): pass
```

`Y__mro__` Python `3.x`

Python 2.x 2.3

```
class X: pass
class Y(object): pass
```

`Y` `class Y(object): pass` 。 `X` `__mro__``AttributeError` 。

Python `object`

```
class X: pass
class Y(object): pass
```

`__metaclass__``type` `object`

```
class X: pass
class Y(object): pass
```

<>``=repr

Python 2 `<>` `!=`; `` `foo` `` `repr(foo)`。

Python 2.x 2.7

```
>>> 1 <> 2
True
>>> 1 <> 1
False
>>> foo = 'hello world'
>>> repr(foo)
"'hello world'"
>>> `foo`
"'hello world'"
```

Python 3.x 3.0

```
>>> 1 <> 2
True
>>> 1 <> 1
False
>>> foo = 'hello world'
>>> repr(foo)
"'hello world'"
>>> `foo`
"'hello world'"
```

/

Python 2.x 2.7

```
"1deadbeef3".decode('hex')
# Out: '\x1d\xea\xdb\xee\xf3'
'\x1d\xea\xdb\xee\xf3'.encode('hex')
# Out: 1deadbeef3
```

Python 3.x 3.0

```
"1deadbeef3".decode('hex')
# Out: '\x1d\xea\xdb\xee\xf3'
'\x1d\xea\xdb\xee\xf3'.encode('hex')
# Out: 1deadbeef3
```

`codecs`

```
"1deadbeef3".decode('hex')
# Out: '\x1d\xea\xdb\xee\xf3'
'\x1d\xea\xdb\xee\xf3'.encode('hex')
# Out: 1deadbeef3
```

`codecs.encode`bytes。 `str`decodeASCII

```
"1deadbeef3".decode('hex')
# Out: '\x1d\xea\xdb\xee\xf3'
'\x1d\xea\xdb\xee\xf3'.encode('hex')
```

```
# Out: 1deadbeef3
```

Python 3cmp

Python 3 `cmp__cmp__`。

> `cmp()__cmp__()`。 `__lt__()` `__eq__()` `__hash__()__hash__()` 。 `cmp()(a > b) - (a < b)cmp(a, b)`。

`cmpkey` keyword only。

`functoolscmp_to_key(func)` `cmp` style`key` style

> 。 `sorted()` `min()` `max()` `heapq.nlargest()` `heapq.nsmallest()` `itertools.groupby()` 。 Python 2。

Python 2.x 2.3

```
x = 'hello world!'
vowels = [x for x in 'AEIOU']

print (vowels)
# Out: ['A', 'E', 'I', 'O', 'U']
print(x)
# Out: 'U'
```

Python 3.x 3.0

```
x = 'hello world!'
vowels = [x for x in 'AEIOU']

print (vowels)
# Out: ['A', 'E', 'I', 'O', 'U']
print(x)
# Out: 'U'
```

Python 2`xhello world!U x`。

Python 3 `xhello world!` 。

2.5PythonPython 3Python 2.7Python 2。

Python 2Python 3for

```
x = 'hello world!'
vowels = [x for x in 'AEIOU']

print (vowels)
# Out: ['A', 'E', 'I', 'O', 'U']
print(x)
# Out: 'U'
```

`map()`iterable。 Python 2 `map`。 Python 3 `map`*map* 。

```
# Python 2.X
>>> map(str, [1, 2, 3, 4, 5])
['1', '2', '3', '4', '5']
>>> type(_)
>>> <class 'list'>

# Python 3.X
>>> map(str, [1, 2, 3, 4, 5])
<map object at 0x*>
>>> type(_)
<class 'map'>

# We need to apply map again because we "consumed" the previous map....
>>> map(str, [1, 2, 3, 4, 5])
>>> list(_)
['1', '2', '3', '4', '5']
```

Python 2`None`。 Python 3。

Python 2.x 2.3

```
# Python 2.X
>>> map(str, [1, 2, 3, 4, 5])
['1', '2', '3', '4', '5']
>>> type(_)
>>> <class 'list'>

# Python 3.X
>>> map(str, [1, 2, 3, 4, 5])
<map object at 0x*>
>>> type(_)
<class 'map'>

# We need to apply map again because we "consumed" the previous map....
>>> map(str, [1, 2, 3, 4, 5])
>>> list(_)
['1', '2', '3', '4', '5']
```

Python 3.x 3.0

```
# Python 2.X
>>> map(str, [1, 2, 3, 4, 5])
['1', '2', '3', '4', '5']
>>> type(_)
>>> <class 'list'>

# Python 3.X
>>> map(str, [1, 2, 3, 4, 5])
<map object at 0x*>
>>> type(_)
<class 'map'>

# We need to apply map again because we "consumed" the previous map....
>>> map(str, [1, 2, 3, 4, 5])
>>> list(_)
['1', '2', '3', '4', '5']
```

Python 2`None` `mapitertools.izip_longest` 。 Python 3。

Python 2

Python 2.x 2.3

```
# Python 2.X
>>> map(str, [1, 2, 3, 4, 5])
['1', '2', '3', '4', '5']
>>> type(_)
>>> <class 'list'>

# Python 3.X
>>> map(str, [1, 2, 3, 4, 5])
<map object at 0x*>
>>> type(_)
<class 'map'>

# We need to apply map again because we "consumed" the previous map....
>>> map(str, [1, 2, 3, 4, 5])
>>> list(_)
['1', '2', '3', '4', '5']
```

Python 3

Python 3.x 3.0

```
# Python 2.X
>>> map(str, [1, 2, 3, 4, 5])
['1', '2', '3', '4', '5']
>>> type(_)
>>> <class 'list'>

# Python 3.X
>>> map(str, [1, 2, 3, 4, 5])
<map object at 0x*>
>>> type(_)
<class 'map'>

# We need to apply map again because we "consumed" the previous map....
>>> map(str, [1, 2, 3, 4, 5])
>>> list(_)
['1', '2', '3', '4', '5']
```

Python 2/3`map` 。 `map(str, [1, 2, 3, 4, 5])`

```
# Python 2.X
>>> map(str, [1, 2, 3, 4, 5])
['1', '2', '3', '4', '5']
>>> type(_)
>>> <class 'list'>

# Python 3.X
>>> map(str, [1, 2, 3, 4, 5])
<map object at 0x*>
>>> type(_)
<class 'map'>
```

```
# We need to apply map again because we "consumed" the previous map....
>>> map(str, [1, 2, 3, 4, 5])
>>> list(_)
['1', '2', '3', '4', '5']
```

filtermapzip

Python 2.x 2.7

Python 2 `filter` `mapzip`。 `mapzipfilter`

```
>>> s = filter(lambda x: x.isalpha(), 'a1b2c3')
>>> s
'abc'
>>> s = map(lambda x: x * x, [0, 1, 2])
>>> s
[0, 1, 4]
>>> s = zip([0, 1, 2], [3, 4, 5])
>>> s
[(0, 3), (1, 4), (2, 5)]
```

Python 3.x 3.0

Python 3 `filter` `mapzip`

```
>>> s = filter(lambda x: x.isalpha(), 'a1b2c3')
>>> s
'abc'
>>> s = map(lambda x: x * x, [0, 1, 2])
>>> s
[0, 1, 4]
>>> s = zip([0, 1, 2], [3, 4, 5])
>>> s
[(0, 3), (1, 4), (2, 5)]
```

Python 2 `itertools.izip`Python 3`zip` `izip`Python 3。

/

Python 3 PEP 404Python2.`from ... import *`。

Python 2Python 3

- `from __future__ import absolute_import`
-

Python 2

```
import foo
```

import`foo`。

```
import foo
```

.。

---

shapes 。

```
import foo
```

circle.py square.pytriangle.pyutil.py。

```
import foo
```

```
import foo
```

.。

shapes

```
import foo
```

3util.py

```
import foo
```

```
import foo
```

..。 . s。

I / O.

file3.x open。

I / OioStringIO

```
import io
assert io.open is open # the builtin is an alias
buffer = io.StringIO()
buffer.write('hello, ') # returns number of characters written
buffer.write('world!\n')
buffer.getvalue() # 'hello, world!\n'
```

```
import io
assert io.open is open # the builtin is an alias
buffer = io.StringIO()
buffer.write('hello, ') # returns number of characters written
buffer.write('world!\n')
buffer.getvalue() # 'hello, world!\n'
```

`locale.getpreferredencoding(False)`locale.getpreferredencoding(False) ◦ encoding

```
import io
assert io.open is open # the builtin is an alias
buffer = io.StringIO()
buffer.write('hello, ') # returns number of characters written
buffer.write('world!\n')
buffer.getvalue() # 'hello, world!\n'
```

round

Python 2`round()`0◦

Python 2.x 2.7

```
round(1.5)  # Out: 2.0
round(0.5)  # Out: 1.0
round(-0.5)  # Out: -1.0
round(-1.5)  # Out: -2.0
```

Python 3 `round()` ◦

Python 3.x 3.0

```
round(1.5)  # Out: 2.0
round(0.5)  # Out: 1.0
round(-0.5)  # Out: -1.0
round(-1.5)  # Out: -2.0
```

round*halfeven* `round(2.5)`23.0◦

◦

IEEE 754.NET◦

◦ ◦ ◦

---

round

`round()`Python 2.7`float`

Python 2.x 2.7

```
round(1.5)  # Out: 2.0
round(0.5)  # Out: 1.0
round(-0.5)  # Out: -1.0
round(-1.5)  # Out: -2.0
```

Python 3.0`int` ◦

Python 3.x 3.0

```
round(1.5)  # Out: 2.0
round(0.5)  # Out: 1.0
round(-0.5)  # Out: -1.0
round(-1.5)  # Out: -2.0
```

Python 2 `True` `False``None`。 。

Python 2.x 2.0

```
True, False = False, True
True   # False
False  # True
```

Python 2.4`None`。

Python 2.x 2.4

```
True, False = False, True
True   # False
False  # True
```

Python 3 `True` `False``None`。

Python 3.x 3.0

```
True, False = False, True
True   # False
False  # True
```

Python 2`None`

Python 2.x 2.3

```
hi = sys.stdout.write('hello world\n')
# Out: hello world
type(hi)
# Out: <type 'NoneType'>
```

Python 3

Python 3.x 3.0

```
hi = sys.stdout.write('hello world\n')
# Out: hello world
type(hi)
# Out: <type 'NoneType'>
```

Python 2C `ssize_t``long`L。 32Python

Python 2.x 2.7

```
>>> 2**31
```

```
2147483648L
>>> type(2**31)
<type 'long'>
>>> 2**30
1073741824
>>> type(2**30)
<type 'int'>
>>> 2**31 - 1  # 2**31 is long and long - int is long
2147483647L
```

Python 3`long`;`int`。

Python 3.x 3.0

```
>>> 2**31
2147483648L
>>> type(2**31)
<type 'long'>
>>> 2**30
1073741824
>>> type(2**30)
<type 'int'>
>>> 2**31 - 1  # 2**31 is long and long - int is long
2147483647L
```

Python 2.x 2.7

Python 2`__nonzero__`。 True。

```
class MyClass:
    def __nonzero__(self):
        return False

my_instance = MyClass()
print bool(MyClass)        # True
print bool(my_instance)    # False
```

Python 3.x 3.0

Python 3`__bool__``__nonzero__`

```
class MyClass:
    def __nonzero__(self):
        return False

my_instance = MyClass()
print bool(MyClass)        # True
print bool(my_instance)    # False
```

Python 2Python 3 https://riptutorial.com/zh-TW/python/topic/809/python-2python-3

64: intfloatstrtuplefrozensets

Examples

```
foo = "bar"
foo[0] = "c" # Error
```

。

```
foo = ("bar", 1, "Hello!",)
foo[1] = 2 # ERROR!!
```

。 。

Frozenset

```
foo = frozenset(["bar", 1, "Hello!"])
foo[2] = 7 # ERROR
foo.add(3) # ERROR
```

。 frozensets。

intfloatstrtuplefrozensets https://riptutorial.com/zh-TW/python/topic/4806/-int-float-str-tuple frozensets-

65:

GILpython。 IOCPU。 CPU`multiprocessing`。

GILCPythonPythonPyPy。 JythonIronPythonGIL 。

Examples

```
import multiprocessing

def fib(n):
    """computing the Fibonacci in an inefficient way
    was chosen to slow down the CPU."""
    if n <= 2:
        return 1
    else:
        return fib(n-1)+fib(n-2)
p = multiprocessing.Pool()
print(p.map(fib,[38,37,36,35,34,33]))

# Out: [39088169, 24157817, 14930352, 9227465, 5702887, 3524578]
```

`fib`**1.8** 。

Python 2.2+

ParentChildren

child.py

```
import time

def main():
    print "starting work"
    time.sleep(1)
    print "work work work work work"
    time.sleep(1)
    print "done working"

if __name__ == '__main__':
    main()
```

parent.py

```
import time

def main():
    print "starting work"
    time.sleep(1)
    print "work work work work work"
    time.sleep(1)
    print "done working"
```

```
if __name__ == '__main__':
    main()
```

HTTP//。 **child.py**。 Redis。

C

CPythonCGIL。

```
#include "Python.h"
...
PyObject *pyfunc(PyObject *self, PyObject *args) {
    ...
    Py_BEGIN_ALLOW_THREADS
    // Threaded C code
    ...
    Py_END_ALLOW_THREADS
    ...
}
```

PyPar

PyParMPIPython。 PyParhttps://github.com/daleroberts/pypar

```
import pypar as pp

ncpus = pp.size()
rank = pp.rank()
node = pp.get_processor_name()

print 'I am rank %d of %d on node %s' % (rank, ncpus, node)

if rank == 0:
  msh = 'P0'
  pp.send(msg, destination=1)
  msg = pp.receive(source=rank-1)
  print 'Processor 0 received message "%s" from rank %d' % (msg, rank-1)
else:
  source = rank-1
  destination = (rank+1) % ncpus
  msg = pp.receive(source)
  msg = msg + 'P' + str(rank)
  pypar.send(msg, destination)
pp.finalize()
```

https://riptutorial.com/zh-TW/python/topic/542/

66:

- packfmtv1v2...
- unpackfmtbuffer

Examples

```
from struct import pack

print(pack('I3c', 123, b'a', b'b', b'c'))  # b'{\x00\x00\x00abc'
```

```
from struct import unpack

print(unpack('I3c', b'{\x00\x00\x00abc'))  # (123, b'a', b'b', b'c')
```

“ **struct** ”pythonpython。

pack。 。

```
import struct
import sys
print "Native byteorder: ", sys.byteorder
# If no byteorder is specified, native byteorder is used
buffer = struct.pack("ihb", 3, 4, 5)
print "Byte chunk: ", repr(buffer)
print "Byte chunk unpacked: ", struct.unpack("ihb", buffer)
# Last element as unsigned short instead of unsigned char ( 2 Bytes)
buffer = struct.pack("ihh", 3, 4, 5)
print "Byte chunk: ", repr(buffer)
```

> '\ x03 \ x00 \ x00 \ x00 \ x04 \ x00 \ x05'3,4,5'\ x03 \ x00 \ x00 \ x00 \ x04 \ x00 \ X05 \ X00“

。

```
import struct
import sys
print "Native byteorder: ", sys.byteorder
# If no byteorder is specified, native byteorder is used
buffer = struct.pack("ihb", 3, 4, 5)
print "Byte chunk: ", repr(buffer)
print "Byte chunk unpacked: ", struct.unpack("ihb", buffer)
# Last element as unsigned short instead of unsigned char ( 2 Bytes)
buffer = struct.pack("ihh", 3, 4, 5)
print "Byte chunk: ", repr(buffer)
```

> '\ x03 \ x00 \ x04 \ x00 \ x05 \ x00'
>
> '\ x00 \ x03 \ x00 \ x04 \ x00 \ x05'

。

```
import struct
import sys
print "Native byteorder: ", sys.byteorder
# If no byteorder is specified, native byteorder is used
buffer = struct.pack("ihb", 3, 4, 5)
print "Byte chunk: ", repr(buffer)
print "Byte chunk unpacked: ", struct.unpack("ihb", buffer)
# Last element as unsigned short instead of unsigned char ( 2 Bytes)
buffer = struct.pack("ihh", 3, 4, 5)
print "Byte chunk: ", repr(buffer)
```

'\ x03 \ x00 \ x04 \ x00 \ x05 \ x00 \ x00 \ x00'

'\ x00 \ x00 \ x03 \ x00 \ x04 \ x00 \ x05 \ x00'

https://riptutorial.com/zh-TW/python/topic/2978/

67:

Python。 。 。 。

Examples

	ns	
	ns`__main__`	
	ns`__main__`	
	ns	
	ns	
`exec`	ns	
`eval()`		ns
`execfile()execfile()`		ns
`input()`		ns

https://riptutorial.com/zh-TW/python/topic/10741/-

68: “pip”PyPI

pythonpip。 `ImportError`。 Windows `Python_root/Scripts/pip.exePython_root/Scripts/pip.exe` `__main__.pypipmain`。 pippip。 pip pipPyPI Package Manager

- pip。 <function | attribute | class>function
  -
    - 。 shellbashzshfish。
  - check_isolated
    - param args {list}
    - {boolean}
  - create_main_parser
    - {pip.baseparser.ConfigOptionParser}
  - =
    - param args {list}
    - {integer}0
  - parseopts
    - param args {list}
  - get_installed_distributions
    - {list}
  - get_similar_commands
    - 。
    - {string}
    - {boolean}
  - get_summaries=
    - 。
  - get_prog
    - {string}
  - dist_is_editableDIST
    -
    - param dist {object}
    - {boolean}
  - commands_dict
    - {}

Examples

```
import pip

command = 'install'
parameter = 'selenium'
second_param = 'numpy' # You can give as many package names as needed
switch = '--upgrade'

pip.main([command, parameter, second_param, switch])
```

`pip.main(['freeze'])pip.main(['freeze', '', ''])`。

/“1”。

```
import pip

command = 'install'
parameter = 'selenium'
second_param = 'numpy' # You can give as many package names as needed
switch = '--upgrade'

pip.main([command, parameter, second_param, switch])
```

。

```
import pip

command = 'install'
parameter = 'selenium'
second_param = 'numpy' # You can give as many package names as needed
switch = '--upgrade'

pip.main([command, parameter, second_param, switch])
```

ImportError

python。

```
if __name__ == '__main__':
    try:
        import requests
    except ImportError:
        print("To use this module you need 'requests' module")
        t = input('Install requests? y/n: ')
        if t == 'y':
            import pip
            pip.main(['install', 'requests'])
            import requests
            import os
            import sys
            pass
        else:
            import os
            import sys
            print('Some functionality can be unavailable.')
else:
    import requests
    import os
    import sys
```

3.43.6。 .whlwheel。 。 `scikit_learn-0.18.1-cp36-cp36m-win_amd64.whl` [package_name] - [version] - [python interpreter] - [python-interpreter] - [Operating System] .whl。 python。 pip`kjhfkjdf.whl is not a valid wheel filename.` 。

.whl7-zip。 - 。 `site-packges`。

“pip”PyPI https://riptutorial.com/zh-TW/python/topic/10730/-pip--pypi

69: AMQPStormRabbitMQ

AMQPStormpypipip

```
pip install amqpstorm
```

Examples

RabbitMQ

。

```
from amqpstorm import Connection
```

。 `start_consumingto_tuple`。

。 RabbitMQ。

```
from amqpstorm import Connection
```

RabbitMQ。

```
from amqpstorm import Connection
```

。 。

```
from amqpstorm import Connection
```

RabbitMQ。 `on_message`。

RabbitMQ`simple_queue` RabbitMQ。

```
from amqpstorm import Connection
```

IORabbitMQ。

```
from amqpstorm import Connection
```

RabbitMQ

。

```
from amqpstorm import Connection
from amqpstorm import Message
```

RabbitMQ。

```
from amqpstorm import Connection
from amqpstorm import Message
```

。 。

```
from amqpstorm import Connection
from amqpstorm import Message
```

。

```
from amqpstorm import Connection
from amqpstorm import Message
```

publishrouting_key。 simple_queue。

```
from amqpstorm import Connection
from amqpstorm import Message
```

RabbitMQ

。 ;confirm delivery delivery_modedurable。 RabbitMQ 。

。

```
channel.queue.bind(exchange='amq.direct', routing_key='hello', queue='hello')
```

。

```
channel.queue.bind(exchange='amq.direct', routing_key='hello', queue='hello')
```

- x-message-ttl -

  。

- X-

  。

- X-

  hello_delayhelloExchange。

Pika。

```
channel.queue.bind(exchange='amq.direct', routing_key='hello', queue='hello')
```

RabbitMQ。

Overview					Messages			Messa	
Name	**Exclusive**	Parameters	**Policy**	**Status**	**Ready**	**Unacked**	**Total**	**incoming**	**deliv**
hello		D		Idle	1	0	1		
hello_delay		TTL DLX DLK D		Idle	0	0	0	0.00/s	

。

```
channel.queue.bind(exchange='amq.direct', routing_key='hello', queue='hello')
```

AMQPStormRabbitMQ https://riptutorial.com/zh-TW/python/topic/3373/amqpstormrabbitmq

70: Pandas

pandas。

Examples

Pandas

```
# Print the working directory
import os
print os.getcwd()
# C:\Python27\Scripts

# Set the working directory
os.chdir('C:/Users/general1/Documents/simple Python files')
print os.getcwd()
# C:\Users\general1\Documents\simple Python files

# load pandas
import pandas as pd

# read a csv data file named 'small_dataset.csv' containing 4 lines and 3 variables
my_data = pd.read_csv("small_dataset.csv")
my_data
#     x   y   z
# 0   1   2   3
# 1   4   5   6
# 2   7   8   9
# 3  10  11  12

my_data.shape       # number of rows and columns in data set
# (4, 3)

my_data.shape[0]    # number of rows in data set
# 4

my_data.shape[1]    # number of columns in data set
# 3

# Python uses 0-based indexing.  The first row or column in a data set is located
# at position 0.  In R the first row or column in a data set is located
# at position 1.

# Select the first two rows
my_data[0:2]
#    x   y   z
#0   1   2   3
#1   4   5   6

# Select the second and third rows
my_data[1:3]
#    x  y  z
# 1  4  5  6
# 2  7  8  9

# Select the third row
```

```
my_data[2:3]
#    x   y   z
#2   7   8   9

# Select the first two elements of the first column
my_data.iloc[0:2, 0:1]
#    x
# 0  1
# 1  4

# Select the first element of the variables y and z
my_data.loc[0, ['y', 'z']]
# y    2
# z    3

# Select the first three elements of the variables y and z
my_data.loc[0:2, ['y', 'z']]
#    y  z
# 0  2  3
# 1  5  6
# 2  8  9

# Write the first three elements of the variables y and z
# to an external file.  Here index = 0 means do not write row names.

my_data2 = my_data.loc[0:2, ['y', 'z']]

my_data2.to_csv('my.output.csv', index = 0)
```

Pandas https://riptutorial.com/zh-TW/python/topic/8854/pandas-

71: PythonWindows

WindowsUI。 WindowsPowershell“”。 Web。 WindowsPython。

Examples

Python

pywin32 Python for Windows。 Python。

```
import win32serviceutil
import win32service
import win32event
import servicemanager
import socket


class AppServerSvc (win32serviceutil.ServiceFramework):
    _svc_name_ = "TestService"
    _svc_display_name_ = "Test Service"

    def __init__(self,args):
        win32serviceutil.ServiceFramework.__init__(self,args)
        self.hWaitStop = win32event.CreateEvent(None,0,0,None)
        socket.setdefaulttimeout(60)

    def SvcStop(self):
        self.ReportServiceStatus(win32service.SERVICE_STOP_PENDING)
        win32event.SetEvent(self.hWaitStop)

    def SvcDoRun(self):
        servicemanager.LogMsg(servicemanager.EVENTLOG_INFORMATION_TYPE,
                              servicemanager.PYS_SERVICE_STARTED,
                              (self._svc_name_,''))
        self.main()

    def main(self):
        pass

if __name__ == '__main__':
    win32serviceutil.HandleCommandLine(AppServerSvc)
```

。 main。

。 。 GUI。 Python

```
import win32serviceutil
import win32service
import win32event
import servicemanager
import socket


class AppServerSvc (win32serviceutil.ServiceFramework):
```

```
    _svc_name_ = "TestService"
    _svc_display_name_ = "Test Service"

    def __init__(self,args):
        win32serviceutil.ServiceFramework.__init__(self,args)
        self.hWaitStop = win32event.CreateEvent(None,0,0,None)
        socket.setdefaulttimeout(60)

    def SvcStop(self):
        self.ReportServiceStatus(win32service.SERVICE_STOP_PENDING)
        win32event.SetEvent(self.hWaitStop)

    def SvcDoRun(self):
        servicemanager.LogMsg(servicemanager.EVENTLOG_INFORMATION_TYPE,
                              servicemanager.PYS_SERVICE_STARTED,
                              (self._svc_name_,''))
        self.main()

    def main(self):
        pass

if __name__ == '__main__':
    win32serviceutil.HandleCommandLine(AppServerSvc)
```

my_script.pymain。 PythonPython。

Windows Server Resource Kit。

Flask Web

。 main()run()。 WSGIRequestHandler。

```
import win32serviceutil
import win32service
import win32event
import servicemanager
from multiprocessing import Process

from app import app


class Service(win32serviceutil.ServiceFramework):
    _svc_name_ = "TestService"
    _svc_display_name_ = "Test Service"
    _svc_description_ = "Tests Python service framework by receiving and echoing messages over
a named pipe"

    def __init__(self, *args):
        super().__init__(*args)

    def SvcStop(self):
        self.ReportServiceStatus(win32service.SERVICE_STOP_PENDING)
        self.process.terminate()
        self.ReportServiceStatus(win32service.SERVICE_STOPPED)

    def SvcDoRun(self):
        self.process = Process(target=self.main)
        self.process.start()
```

```
        self.process.run()

    def main(self):
        app.run()


if __name__ == '__main__':
    win32serviceutil.HandleCommandLine(Service)
```

http://stackoverflow.com/a/25130524/318488

PythonWindows https://riptutorial.com/zh-TW/python/topic/9065/pythonwindows

72: PythonRaspberry PI

Examples

-

DS18B20Raspberry pi

DS18B20Raspberry pi

1. VCC
2. GND
3.

R14.7k

1. **Vcc**Raspberry pi5v3.3vPIN01,02,04,17。
2. **Gnd**Raspberry piGndPIN06,09,14,20,25。
3. PIN07

RPi

4. puttylinux / unixRaspberry pi。
5. /boot/config.txt。

   nano /boot/config.txt

6. `dtoverlay=w1-gpio`。
7. Raspberry pi `sudo reboot` 。
8. Raspberry pi`sudo modprobe g1-gpio`
9. `sudo modprobe w1-therm`
10. / sys / bus / w1 / devices `cd /sys/bus/w1/devices`
11. 28 - ********。
12. `cd 28-********`

13. **w1-slave** CRC。 `cat w1-slave`。

python

```
import glob
import time

RATE = 30
sensor_dirs = glob.glob("/sys/bus/w1/devices/28*")

if len(sensor_dirs) != 0:
    while True:
        time.sleep(RATE)
        for directories in sensor_dirs:
            temperature_file = open(directories + "/w1_slave")
            # Reading the files
            text = temperature_file.read()
            temperature_file.close()
            # Split the text with new lines (\n) and select the second line.
            second_line = text.split("\n")[1]
            # Split the line into words, and select the 10th word
            temperature_data = second_line.split(" ")[9]
            # We will read after ignoring first two character.
            temperature = float(temperature_data[2:])
            # Now normalise the temperature by dividing 1000.
            temperature = temperature / 1000
            print 'Address : '+str(directories.split('/')[-1])+', Temperature :
'+str(temperature)
```

python。 RATE。

GPIO

1. [https://www.element14.com/community/servlet/JiveServlet/previewBody/73950-102-11-339300/pi3_gpio.png] [3]

PythonRaspberry PI https://riptutorial.com/zh-TW/python/topic/10735/pythonraspberry-pi

73: PythonWeb

“”。 。 。

WebPython

requests

HTTP。

requests-cache

requests ;。 。 .....。

scrapy

Webrequests。

selenium

Selenium WebDriverPython。 requestsHTTP。 requestsJavaScript。

HTML

BeautifulSoup

HTMLXMLPythonHTML Parser html5lib lxmllxml.html

lxml

HTMLXML。 CSSXPathHTML。

Examples

lxml

```
# For Python 2 compatibility.
from __future__ import print_function

import lxml.html
import requests


def main():
    r = requests.get("https://httpbin.org")
    html_source = r.text
    root_element = lxml.html.fromstring(html_source)
```

```
    # Note root_element.xpath() gives a *list* of results.
    # XPath specifies a path to the element we want.
    page_title = root_element.xpath('/html/head/title/text()')[0]
    print(page_title)

if __name__ == '__main__':
    main()
```

Web

cookie。 `requests.Session`TCP

```
import requests

with requests.Session() as session:
    # all requests through session now have User-Agent header set
    session.headers = {'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_4)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36'}

    # set cookies
    session.get('http://httpbin.org/cookies/set?key=value')

    # get cookies
    response = session.get('http://httpbin.org/cookies')
    print(response.text)
```

Scrapy

Scrapy。

```
scrapy startproject projectName
```

。 。 StackOverflow

```
scrapy startproject projectName
```

`projectName\spiders`。 - `projectName\spiders\stackoverflow_spider.py` 。

。

```
scrapy startproject projectName
```

Scrapy

Scrapy `"Scrapy/VERSION (+http://scrapy.org)"` 。 **settings.py** 。

```
#USER_AGENT = 'projectName (+http://www.yourdomain.com)'
```


```
#USER_AGENT = 'projectName (+http://www.yourdomain.com)'
```

BeautifulSoup4

```
from bs4 import BeautifulSoup
import requests

# Use the requests module to obtain a page
res = requests.get('https://www.codechef.com/problems/easy')

# Create a BeautifulSoup object
page = BeautifulSoup(res.text, 'lxml')   # the text field contains the source of the page

# Now use a CSS selector in order to get the table containing the list of problems
datatable_tags = page.select('table.dataTable')  # The problems are in the <table> tag,
                                                 # with class "dataTable"
# We extract the first tag from the list, since that's what we desire
datatable = datatable_tags[0]
# Now since we want problem names, they are contained in <b> tags, which are
# directly nested under <a> tags
prob_tags = datatable.select('a > b')
prob_names = [tag.getText().strip() for tag in prob_tags]

print prob_names
```

Selenium WebDriver

- - SeleniumWeb。

```
from selenium import webdriver

browser = webdriver.Firefox()  # launch firefox browser

browser.get('http://stackoverflow.com/questions?sort=votes')  # load url

title = browser.find_element_by_css_selector('h1').text  # page title (first h1 element)

questions = browser.find_elements_by_css_selector('.question-summary')  # question list

for question in questions:  # iterate over questions
    question_title = question.find_element_by_css_selector('.summary h3 a').text
    question_excerpt = question.find_element_by_css_selector('.summary .excerpt').text
    question_vote = question.find_element_by_css_selector('.stats .vote .votes .vote-count-
post').text

    print "%s\n%s\n%s votes\n-----------\n" % (question_title, question_excerpt,
question_vote)
```

- cookieJavaScript。

urllib.requestWeb

`urllib.request`Web

```
from urllib.request import urlopen

response = urlopen('http://stackoverflow.com/questions?sort=votes')
```

```
data = response.read()

# The received bytes should usually be decoded according the response's character set
encoding = response.info().get_content_charset()
html = data.decode(encoding)
```

Python 2。

```
from subprocess import Popen, PIPE
from lxml import etree
from io import StringIO
```

```
from subprocess import Popen, PIPE
from lxml import etree
from io import StringIO
```

```
-s

-A
```

```
from subprocess import Popen, PIPE
from lxml import etree
from io import StringIO
```

PythonWeb https://riptutorial.com/zh-TW/python/topic/1792/pythonweb

74: Python

Examples

Matplotlib

MatplotlibPython。

matplotlibSO Docs 。

Matplotlib

- matplotlib`pyplot`MATLAB。
- matplotlib。 。

`pyplot`

```
import matplotlib.pyplot as plt

# Generate some data for plotting.
x = [0, 1, 2, 3, 4, 5, 6]
y = [i**2 for i in x]

# Plot the data x, y with some keyword arguments that control the plot style.
# Use two different plot commands to plot both points (scatter) and a line (plot).

plt.scatter(x, y, c='blue', marker='x', s=100) # Create blue markers of shape "x" and size 100
plt.plot(x, y, color='red', linewidth=2) # Create a red line with linewidth 2.

# Add some text to the axes and a title.
plt.xlabel('x data')
plt.ylabel('y data')
plt.title('An example plot')

# Generate the plot and show to the user.
plt.show()
```

`matplotlib.pyplot` `plt.show()` `plt.show(block=True)`。 `plt.show(block=True)` 。

Seaborn

SeabornMatplotlib。 。 SeabornAPI 。

Seaborn。 。

```
import numpy as np  # numpy used to create data from plotting
import seaborn as sns  # common form of importing seaborn

# Generate normally distributed data
data = np.random.randn(1000)

# Plot a histogram with both a rugplot and kde graph superimposed
sns.distplot(data, kde=True, rug=True)
```

。

```
import numpy as np  # numpy used to create data from plotting
import seaborn as sns  # common form of importing seaborn

# Generate normally distributed data
data = np.random.randn(1000)

# Plot a histogram with both a rugplot and kde graph superimposed
sns.distplot(data, kde=True, rug=True)
```

matplotlibSeaborn。 。

```
import numpy as np  # numpy used to create data from plotting
import seaborn as sns  # common form of importing seaborn

# Generate normally distributed data
data = np.random.randn(1000)

# Plot a histogram with both a rugplot and kde graph superimposed
sns.distplot(data, kde=True, rug=True)
```

Mayavi

MayaVI。 VTK 。 VTK **MayaVI**。 。 Matplotlib**VTK** 。

MayaVIPython 2.7xPython 3-x Python 3

。

MayaVI。

```
# Author: Gael Varoquaux <gael.varoquaux@normalesup.org>
# Copyright (c) 2007, Enthought, Inc.
# License: BSD Style.


from numpy import sin, cos, mgrid, pi, sqrt
from mayavi import mlab

mlab.figure(fgcolor=(0, 0, 0), bgcolor=(1, 1, 1))
u, v = mgrid[- 0.035:pi:0.01, - 0.035:pi:0.01]

X = 2 / 3. * (cos(u) * cos(2 * v)
        + sqrt(2) * sin(u) * cos(v)) * cos(u) / (sqrt(2) -
```

```
                                                sin(2 * u) * sin(3 * v))
Y = 2 / 3. * (cos(u) * sin(2 * v) -
        sqrt(2) * sin(u) * sin(v)) * cos(u) / (sqrt(2)
        - sin(2 * u) * sin(3 * v))
Z = -sqrt(2) * cos(u) * cos(u) / (sqrt(2) - sin(2 * u) * sin(3 * v))
S = sin(u)

mlab.mesh(X, Y, Z, scalars=S, colormap='YlGnBu', )

# Nice view from the front
mlab.view(.0, - 5.0, 4)
mlab.show()
```

Plotly

Plotly。 **Plotly Python R JavaScript JuliaMATLAB** 。 Web。

plotly。 。 **Jupyter**。

。 。

250。 。 。

```
import plotly.graph_objs as go
import plotly as ply

# Create random data with numpy
import numpy as np

N = 100
random_x = np.linspace(0, 1, N)
random_y0 = np.random.randn(N)+5
random_y1 = np.random.randn(N)
random_y2 = np.random.randn(N)-5

# Create traces
trace0 = go.Scatter(
    x = random_x,
```

```
y = random_y0,
mode = 'lines',
name = 'lines'
)
trace1 = go.Scatter(
    x = random_x,
    y = random_y1,
    mode = 'lines+markers',
    name = 'lines+markers'
)
trace2 = go.Scatter(
    x = random_x,
    y = random_y2,
    mode = 'markers',
    name = 'markers'
)
data = [trace0, trace1, trace2]

ply.offline.plot(data, filename='line-mode')
```

Python https://riptutorial.com/zh-TW/python/topic/2388/python

75: ZIP

- zipfile
- *class* zipfile。 **ZipFile** *filemode ='r'compression = ZIP_STOREDallowZip64 = True*

ZIP`zipfile.BadZipFile`。

Python 2.7`zipfile.BadZipfile` Python 3.2+

Examples

Zip

`zipfile`。

```
import zipfile
filename = 'zipfile.zip'
```

zipzip 。

```
import zipfile
filename = 'zipfile.zip'
```

Python 2.73.2Python 3`with` context manager。 “”

```
import zipfile
filename = 'zipfile.zip'
```

Zipfile

zipfile。 `printdirstdout`

```
with zipfile.ZipFile(filename) as zip:
    zip.printdir()

    # Out:
    # File Name                                             Modified             Size
    # pyexpat.pyd                                   2016-06-25 22:13:34       157336
    # python.exe                                    2016-06-25 22:13:34        39576
    # python3.dll                                   2016-06-25 22:13:34        51864
    # python35.dll                                  2016-06-25 22:13:34      3127960
    # etc.
```

`namelistnamelist`。

```
with zipfile.ZipFile(filename) as zip:
    zip.printdir()
```

```
    # Out:
    # File Name                                             Modified             Size
    # pyexpat.pyd                                     2016-06-25 22:13:34       157336
    # python.exe                                      2016-06-25 22:13:34        39576
    # python3.dll                                     2016-06-25 22:13:34        51864
    # python35.dll                                    2016-06-25 22:13:34      3127960
    # etc.
```

`infolistnamelist ZipInfo`

```
with zipfile.ZipFile(filename) as zip:
    zip.printdir()

    # Out:
    # File Name                                             Modified             Size
    # pyexpat.pyd                                     2016-06-25 22:13:34       157336
    # python.exe                                      2016-06-25 22:13:34        39576
    # python3.dll                                     2016-06-25 22:13:34        51864
    # python35.dll                                    2016-06-25 22:13:34      3127960
    # etc.
```

zip

zip

```
import zipfile
with zipfile.ZipFile('zipfile.zip','r') as zfile:
    zfile.extractall('path')
```

```
import zipfile
with zipfile.ZipFile('zipfile.zip','r') as zfile:
    zfile.extractall('path')
```

zipfile。

```
import zipfile
new_arch=zipfile.ZipFile("filename.zip",mode="w")
```

write。

```
import zipfile
new_arch=zipfile.ZipFile("filename.zip",mode="w")
```

writestr。

```
import zipfile
new_arch=zipfile.ZipFile("filename.zip",mode="w")
```

ZIP https://riptutorial.com/zh-TW/python/topic/3728/zip

76:

Examples

```
def func(myList):
    for item in myList:
        print(item)
```

```
def func(myList):
    for item in myList:
        print(item)
```

```
def func(myList):
    for item in myList:
        print(item)
```

https://riptutorial.com/zh-TW/python/topic/7744/

77:

。 ;。 Python。 BaseException。

-
- #re-raise
- #Python 3 -
- NonePython 3 -
-
- *[] [ ]*
-
-

Examples

。

```
def even_the_odds(odds):
    if odds % 2 != 1:
        raise ValueError("Did not get an odd number")

    return odds + 1
```

try...except:。

```
try:
    x = 5 / 0
except ZeroDivisionError as e:
    # `e` is the exception object
    print("Got a divide by zero! The exception was:", e)
    # handle exceptional case
    x = 0
finally:
    print "The END"
    # it runs no matter what execute.
```

- ZeroDivisionError - 。

ZeroDivisionErrorArithmeticError

```
try:
    x = 5 / 0
except ZeroDivisionError as e:
    # `e` is the exception object
    print("Got a divide by zero! The exception was:", e)
    # handle exceptional case
    x = 0
finally:
    print "The END"
    # it runs no matter what execute.
```

ZeroDivisionError

```
try:
    x = 5 / 0
except ZeroDivisionError as e:
    # `e` is the exception object
    print("Got a divide by zero! The exception was:", e)
    # handle exceptional case
    x = 0
finally:
    print "The END"
    # it runs no matter what execute.
```

。

tryfinally。

```
resource = allocate_some_expensive_resource()
try:
    do_stuff(resource)
except SomeException as e:
    log_error(e)
    raise  # re-raise the error
finally:
    free_expensive_resource(resource)
```

with 。

。 。

raise。

```
try:
    5 / 0
except ZeroDivisionError:
    print("Got an error")
    raise
```

。 。

```
try:
    5 / 0
except ZeroDivisionError:
    print("Got an error")
    raise
```

raiseraise。

Python 3raise - from

```
try:
    5 / 0
except ZeroDivisionError:
```

```
    print("Got an error")
    raise
```

◦ IOError◦

Python 3.x 3.0

```
>>> try:
    5 / 0
except ZeroDivisionError as e:
    raise ValueError("Division failed") from e

Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
ZeroDivisionError: division by zero

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
ValueError: Division failed
```

◦

IOErrorOSErrorEnvironmentError◦ IOErrorOSError ◦ EnvironmentErrorIOErrorOSError ◦

Python 2.x 2.3

```
BaseException
 +-- SystemExit
 +-- KeyboardInterrupt
 +-- GeneratorExit
 +-- Exception
      +-- StopIteration
      +-- StandardError
      |    +-- BufferError
      |    +-- ArithmeticError
      |    |    +-- FloatingPointError
      |    |    +-- OverflowError
      |    |    +-- ZeroDivisionError
      |    +-- AssertionError
      |    +-- AttributeError
      |    +-- EnvironmentError
      |    |    +-- IOError
      |    |    +-- OSError
      |    |         +-- WindowsError (Windows)
      |    |         +-- VMSError (VMS)
      |    +-- EOFError
      |    +-- ImportError
      |    +-- LookupError
      |    |    +-- IndexError
      |    |    +-- KeyError
      |    +-- MemoryError
      |    +-- NameError
      |    |    +-- UnboundLocalError
      |    +-- ReferenceError
      |    +-- RuntimeError
```

```
      |    |    +-- NotImplementedError
      |    +-- SyntaxError
      |    |    +-- IndentationError
      |    |         +-- TabError
      |    +-- SystemError
      |    +-- TypeError
      |    +-- ValueError
      |         +-- UnicodeError
      |              +-- UnicodeDecodeError
      |              +-- UnicodeEncodeError
      |              +-- UnicodeTranslateError
      +-- Warning
           +-- DeprecationWarning
           +-- PendingDeprecationWarning
           +-- RuntimeWarning
           +-- SyntaxWarning
           +-- UserWarning
           +-- FutureWarning
       +-- ImportWarning
       +-- UnicodeWarning
       +-- BytesWarning
```

Python 3.x 3.0

```
BaseException
 +-- SystemExit
 +-- KeyboardInterrupt
 +-- GeneratorExit
 +-- Exception
      +-- StopIteration
      +-- StandardError
      |    +-- BufferError
      |    +-- ArithmeticError
      |    |    +-- FloatingPointError
      |    |    +-- OverflowError
      |    |    +-- ZeroDivisionError
      |    +-- AssertionError
      |    +-- AttributeError
      |    +-- EnvironmentError
      |    |    +-- IOError
      |    |    +-- OSError
      |    |         +-- WindowsError (Windows)
      |    |         +-- VMSError (VMS)
      |    +-- EOFError
      |    +-- ImportError
      |    +-- LookupError
      |    |    +-- IndexError
      |    |    +-- KeyError
      |    +-- MemoryError
      |    +-- NameError
      |    |    +-- UnboundLocalError
      |    +-- ReferenceError
      |    +-- RuntimeError
      |    |    +-- NotImplementedError
      |    +-- SyntaxError
      |    |    +-- IndentationError
      |    |         +-- TabError
      |    +-- SystemError
      |    +-- TypeError
      |    +-- ValueError
```

```
      |         +-- UnicodeError
      |              +-- UnicodeDecodeError
      |              +-- UnicodeEncodeError
      |              +-- UnicodeTranslateError
      +-- Warning
           +-- DeprecationWarning
           +-- PendingDeprecationWarning
           +-- RuntimeWarning
           +-- SyntaxWarning
           +-- UserWarning
           +-- FutureWarning
       +-- ImportWarning
       +-- UnicodeWarning
       +-- BytesWarning
```

`BaseException`Python。 Python`raise`Python。

```
>>> def failing_function():
...     raise ValueError('Example error!')
>>> failing_function()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 2, in failing_function
ValueError: Example error!
```

`'Example error!'ValueError 'Example error!'failing_function()`。

```
>>> def failing_function():
...     raise ValueError('Example error!')
>>> failing_function()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 2, in failing_function
ValueError: Example error!
```

`except...`

```
>>> def failing_function():
...     raise ValueError('Example error!')
>>> failing_function()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 2, in failing_function
ValueError: Example error!
```

PythonPython https //docs.python.org/3.5/library/exceptions.html。 。

`Exception`

```
class FooException(Exception):
    pass
try:
    raise FooException("insert description here")
except FooException:
    print("A FooException was raised.")
```

```
class FooException(Exception):
    pass
try:
    raise FooException("insert description here")
except FooException:
    print("A FooException was raised.")
```

Exception

```
try:
    very_difficult_function()
except Exception:
    # log / try to reconnect / exit gratiously
finally:
    print "The END"
    # it runs no matter what execute.
```

BaseExceptionException

```
try:
    very_difficult_function()
except Exception:
    # log / try to reconnect / exit gratiously
finally:
    print "The END"
    # it runs no matter what execute.
```

◦ SystemExit KeyboardInterruptMemoryError - ◦ ◦ ◦

“bug”。 ◦

◦

◦

◦ KeyErrorAttributeError◦

```
try:
    d = {}
    a = d[1]
    b = d.non_existing_field
except (KeyError, AttributeError) as e:
    print("A KeyError or an AttributeError exception has been caught.")
```

◦ KeyErrorAttributeError ◦

```
try:
    d = {}
    a = d[1]
    b = d.non_existing_field
except (KeyError, AttributeError) as e:
    print("A KeyError or an AttributeError exception has been caught.")
```

---

input。 。 try / except

Python 3.x 3.0

```
while True:
    try:
        nb = int(input('Enter a number: '))
        break
    except ValueError:
        print('This is not a number, try again.')
```

Python 2.xraw_input ;inputPython 2.x。 input2 + 2。

ValueError 。 except 。 break。 nb。

---

range(n) d *d[i]*。

```
while True:
    try:
        nb = int(input('Enter a number: '))
        break
    except ValueError:
        print('This is not a number, try again.')
```

KeyError 。

tryelse。 。

```
try:
    data = {1: 'one', 2: 'two'}
    print(data[1])
except KeyError as e:
    print('key not found')
else:
    raise ValueError()
# Output: one
# Output: ValueError
```

else:ifelseelif 。 ifelse:

```
try:
    data = {1: 'one', 2: 'two'}
    print(data[1])
except KeyError as e:
    print('key not found')
else:
    raise ValueError()
# Output: one
# Output: ValueError
```

https://riptutorial.com/zh-TW/python/topic/1788/

78:

◦ Python `value = 1, 2, 3` ◦

- 1a“”#a
-
- 1a 1◦ `(1)`◦
- 1,2,331,2,3

◦

◦ ◦

Examples

```
x = (1, 2, 3)
x[0]  # 1
x[1]  # 2
x[2]  # 3
x[3]  # IndexError: tuple index out of range
```

-1

```
x = (1, 2, 3)
x[0]  # 1
x[1]  # 2
x[2]  # 3
x[3]  # IndexError: tuple index out of range
```

```
x = (1, 2, 3)
x[0]  # 1
x[1]  # 2
x[2]  # 3
x[3]  # IndexError: tuple index out of range
```

Python`listtuple`◦

```
>>> t = (1, 4, 9)
>>> t[0] = 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

`list.append.extend`◦ `+=`

```
>>> t = (1, 4, 9)
```

```
>>> t[0] = 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

lists。 。

```
>>> t = (1, 4, 9)
>>> t[0] = 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

```
>>> t = (1, 4, 9)
>>> t[0] = 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

+=“” - “”。

。

```
hash( (1, 2) )  # ok
hash( ([], {"hello"})  # not ok, since lists and sets are not hashabe
```

setdict。

```
hash( (1, 2) )  # ok
hash( ([], {"hello"})  # not ok, since lists and sets are not hashabe
```

```
t = 'a', 'b', 'c', 'd', 'e'
```

```
t = 'a', 'b', 'c', 'd', 'e'
```

```
t = 'a', 'b', 'c', 'd', 'e'
```

```
t = 'a', 'b', 'c', 'd', 'e'
```

```
t = 'a', 'b', 'c', 'd', 'e'
```

。

```
t = 'a', 'b', 'c', 'd', 'e'
```

PEP8 。 PEP8

```
t = 'a', 'b', 'c', 'd', 'e'
```

tuple ◦

```
t = 'a', 'b', 'c', 'd', 'e'
```

Allen B. Downey“ **Think Python** ”。

Python。

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

◦ a = 1, 2, 3◦

◦ Python

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

_

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

Python 3*catch-all*Iterables

Python 3.x 3.0

```
a = 1, 2, 3   # a is the tuple (1, 2, 3)
```

```
colors = "red", "green", "blue"
rev = colors[::-1]
# rev: ("blue", "green", "red")
colors = rev
# colors: ("blue", "green", "red")
```

reverse

```
colors = "red", "green", "blue"
rev = colors[::-1]
# rev: ("blue", "green", "red")
```

```
colors = rev
# colors: ("blue", "green", "red")
```

---

python。 。

- 。
- 。
- 。

“”。 0。

```
tuple1 = ('a', 'b', 'c', 'd', 'e')
tuple2 = ('1','2','3')
tuple3 = ('a', 'b', 'c', 'd', 'e')

cmp(tuple1, tuple2)
Out: 1

cmp(tuple2, tuple1)
Out: -1

cmp(tuple1, tuple3)
Out: 0
```

---

len

```
tuple1 = ('a', 'b', 'c', 'd', 'e')
tuple2 = ('1','2','3')
tuple3 = ('a', 'b', 'c', 'd', 'e')

cmp(tuple1, tuple2)
Out: 1

cmp(tuple2, tuple1)
Out: -1

cmp(tuple1, tuple3)
Out: 0
```

---

max

```
tuple1 = ('a', 'b', 'c', 'd', 'e')
tuple2 = ('1','2','3')
tuple3 = ('a', 'b', 'c', 'd', 'e')

cmp(tuple1, tuple2)
Out: 1

cmp(tuple2, tuple1)
Out: -1
```

```
cmp(tuple1, tuple3)
Out: 0
```

---

min

```
tuple1 = ('a', 'b', 'c', 'd', 'e')
tuple2 = ('1','2','3')
tuple3 = ('a', 'b', 'c', 'd', 'e')

cmp(tuple1, tuple2)
Out: 1

cmp(tuple2, tuple1)
Out: -1

cmp(tuple1, tuple3)
Out: 0
```

---

tuple。

```
tuple1 = ('a', 'b', 'c', 'd', 'e')
tuple2 = ('1','2','3')
tuple3 = ('a', 'b', 'c', 'd', 'e')

cmp(tuple1, tuple2)
Out: 1

cmp(tuple2, tuple1)
Out: -1

cmp(tuple1, tuple3)
Out: 0
```

+

```
tuple1 = ('a', 'b', 'c', 'd', 'e')
tuple2 = ('1','2','3')
tuple3 = ('a', 'b', 'c', 'd', 'e')

cmp(tuple1, tuple2)
Out: 1

cmp(tuple2, tuple1)
Out: -1

cmp(tuple1, tuple3)
Out: 0
```

https://riptutorial.com/zh-TW/python/topic/927/

79:

`type`Python。

- 。
- 。
- Python 3.6`__init_subclass__()` 。

Examples

`type` meta。 /。

```
Dummy = type('OtherDummy', (), dict(x=1))
Dummy.__class__               # <type 'type'>
Dummy().__class__.__class__   # <type 'type'>
```

`type`。

```
Dummy = type('OtherDummy', (), dict(x=1))
Dummy.__class__               # <type 'type'>
Dummy().__class__.__class__   # <type 'type'>
```

`mytypetype` 。

```
Dummy = type('OtherDummy', (), dict(x=1))
Dummy.__class__               # <type 'type'>
Dummy().__class__.__class__   # <type 'type'>
```

`class`。

```
Dummy = type('OtherDummy', (), dict(x=1))
Dummy.__class__               # <type 'type'>
Dummy().__class__.__class__   # <type 'type'>
```

`objectobject type` 。 Python 2Python 3

Python 2.x 2.7

`__metaclass__`。

```
Dummy = type('OtherDummy', (), dict(x=1))
Dummy.__class__               # <type 'type'>
Dummy().__class__.__class__   # <type 'type'>
```

Python 3.x 3.0

`metaclass`。

```
Dummy = type('OtherDummy', (), dict(x=1))
Dummy.__class__               # <type 'type'>
Dummy().__class__.__class__   # <type 'type'>
```

metaclass。 class MyDummy(metaclass=mytype, x=2)x=2mytype。

python。

/。 python singleton 。

```
class SingletonType(type):
    def __call__(cls, *args, **kwargs):
        try:
            return cls.__instance
        except AttributeError:
            cls.__instance = super(SingletonType, cls).__call__(*args, **kwargs)
            return cls.__instance
```

Python 2.x 2.7

```
class SingletonType(type):
    def __call__(cls, *args, **kwargs):
        try:
            return cls.__instance
        except AttributeError:
            cls.__instance = super(SingletonType, cls).__call__(*args, **kwargs)
            return cls.__instance
```

Python 3.x 3.0

```
class SingletonType(type):
    def __call__(cls, *args, **kwargs):
        try:
            return cls.__instance
        except AttributeError:
            cls.__instance = super(SingletonType, cls).__call__(*args, **kwargs)
            return cls.__instance
```

```
class SingletonType(type):
    def __call__(cls, *args, **kwargs):
        try:
            return cls.__instance
        except AttributeError:
            cls.__instance = super(SingletonType, cls).__call__(*args, **kwargs)
            return cls.__instance
```

Python 2.x 2.7

```
class MyClass(object):
    __metaclass__ = SomeMetaclass
```

Python 3.x 3.0

```
class MyClass(object):
    __metaclass__ = SomeMetaclass
```

Python 23six

```
class MyClass(object):
    __metaclass__ = SomeMetaclass
```

。 。

```
class VerboseMetaclass(type):

    def __new__(cls, class_name, class_parents, class_dict):
        print("Creating class ", class_name)
        new_class = super().__new__(cls, class_name, class_parents, class_dict)
        return new_class
```

```
class VerboseMetaclass(type):

    def __new__(cls, class_name, class_parents, class_dict):
        print("Creating class ", class_name)
        new_class = super().__new__(cls, class_name, class_parents, class_dict)
        return new_class
```

```
class VerboseMetaclass(type):

    def __new__(cls, class_name, class_parents, class_dict):
        print("Creating class ", class_name)
        new_class = super().__new__(cls, class_name, class_parents, class_dict)
        return new_class
```

Python。 。

X`type(x)`

```
>>> type(5)
<type 'int'>
>>> type(str)
<type 'type'>
>>> type([1, 2, 3])
<type 'list'>

>>> class C(object):
...     pass
...
>>> type(C)
<type 'type'>
```

python`type`。 `type`。 。

Python `type` 。

```
>>> type(5)
<type 'int'>
>>> type(str)
<type 'type'>
>>> type([1, 2, 3])
<type 'list'>

>>> class C(object):
...     pass
...
>>> type(C)
<type 'type'>
```

MyClassSimplestMetaclass

```
>>> type(5)
<type 'int'>
>>> type(str)
<type 'type'>
>>> type([1, 2, 3])
<type 'list'>

>>> class C(object):
...     pass
...
>>> type(C)
<type 'type'>
```

`__new__type__new__`

```
>>> type(5)
<type 'int'>
>>> type(str)
<type 'type'>
>>> type([1, 2, 3])
<type 'list'>

>>> class C(object):
...     pass
...
>>> type(C)
<type 'type'>
```

Python。

```
>>> type(1)
int
```

1`int`。

```
>>> type(1)
int
```

```
>>> type(1)
```

```
int
```

bar

```
>>> type(1)
int
```

barFoo。 Foo

```
>>> type(1)
int
```

Footype。 type

```
>>> type(1)
int
```

。

- Python
-
- type

Python“hackable”。

https://riptutorial.com/zh-TW/python/topic/286/

80:

- pythonOCR。

Examples

PyTesseract

PyTesseractOCRpython。

PyTesseract

```
try:
    import Image
except ImportError:
    from PIL import Image

import pytesseract

    #Basic OCR
    print(pytesseract.image_to_string(Image.open('test.png')))

    #In French
    print(pytesseract.image_to_string(Image.open('test-european.jpg'), lang='fra’))
```

PyTesseract。

PyOCR

PyOCR 。

PyTesseract。

```
from PIL import Image
import sys

import pyocr
import pyocr.builders

tools = pyocr.get_available_tools()
# The tools are returned in the recommended order of usage
tool = tools[0]

langs = tool.get_available_languages()
lang = langs[0]
# Note that languages are NOT sorted in any way. Please refer
# to the system locale settings for the default language
# to use.
```

```
from PIL import Image
import sys
```

```
import pyocr
import pyocr.builders

tools = pyocr.get_available_tools()
# The tools are returned in the recommended order of usage
tool = tools[0]

langs = tool.get_available_languages()
lang = langs[0]
# Note that languages are NOT sorted in any way. Please refer
# to the system locale settings for the default language
# to use.
```

https://riptutorial.com/zh-TW/python/topic/9302/

81: CLI

hgsvn“”。

。

hgsvn

```
usage: sub <command>

commands:

  status -  show status
  list   -  print list
```

Examples

```
"""
usage: sub <command>

commands:

  status -  show status
  list   -  print list
"""

import sys

def check():
    print("status")
    return 0

if sys.argv[1:] == ['status']:
    sys.exit(check())
elif sys.argv[1:] == ['list']:
    print("list")
else:
    print(__doc__.strip())
```

```
"""
usage: sub <command>

commands:

  status -  show status
  list   -  print list
"""

import sys

def check():
    print("status")
    return 0
```

```
if sys.argv[1:] == ['status']:
    sys.exit(check())
elif sys.argv[1:] == ['list']:
    print("list")
else:
    print(__doc__.strip())
```

- deps
-
-

argparse

```
import argparse
import sys

def check():
    print("status")
    return 0

parser = argparse.ArgumentParser(prog="sub", add_help=False)
subparser = parser.add_subparsers(dest="cmd")

subparser.add_parser('status', help='show status')
subparser.add_parser('list', help='print list')

# hack to show help when no arguments supplied
if len(sys.argv) == 1:
    parser.print_help()
    sys.exit(0)

args = parser.parse_args()

if args.cmd == 'list':
    print('list')
elif args.cmd == 'status':
    sys.exit(check())
```

```
import argparse
import sys

def check():
    print("status")
    return 0

parser = argparse.ArgumentParser(prog="sub", add_help=False)
subparser = parser.add_subparsers(dest="cmd")

subparser.add_parser('status', help='show status')
subparser.add_parser('list', help='print list')

# hack to show help when no arguments supplied
if len(sys.argv) == 1:
    parser.print_help()
    sys.exit(0)

args = parser.parse_args()
```

```
if args.cmd == 'list':
    print('list')
elif args.cmd == 'status':
    sys.exit(check())
```

- Python
-

argparse

http://www.riptutorial.com/python/example/25282/argparse--default-help-formatter-。

```
import argparse
import sys

class CustomHelpFormatter(argparse.HelpFormatter):
    def _format_action(self, action):
        if type(action) == argparse._SubParsersAction:
            # inject new class variable for subcommand formatting
            subactions = action._get_subactions()
            invocations = [self._format_action_invocation(a) for a in subactions]
            self._subcommand_max_length = max(len(i) for i in invocations)

        if type(action) == argparse._SubParsersAction._ChoicesPseudoAction:
            # format subcommand help line
            subcommand = self._format_action_invocation(action) # type: str
            width = self._subcommand_max_length
            help_text = ""
            if action.help:
                help_text = self._expand_help(action)
            return "  {:{width}} -  {}\n".format(subcommand, help_text, width=width)

        elif type(action) == argparse._SubParsersAction:
            # process subcommand help section
            msg = '\n'
            for subaction in action._get_subactions():
                msg += self._format_action(subaction)
            return msg
        else:
            return super(CustomHelpFormatter, self)._format_action(action)


def check():
    print("status")
    return 0

parser = argparse.ArgumentParser(usage="sub <command>", add_help=False,
             formatter_class=CustomHelpFormatter)

subparser = parser.add_subparsers(dest="cmd")
subparser.add_parser('status', help='show status')
subparser.add_parser('list', help='print list')

# custom help messge
parser._positionals.title = "commands"

# hack to show help when no arguments supplied
if len(sys.argv) == 1:
```

```
    parser.print_help()
    sys.exit(0)

args = parser.parse_args()

if args.cmd == 'list':
    print('list')
elif args.cmd == 'status':
    sys.exit(check())
```

```
import argparse
import sys

class CustomHelpFormatter(argparse.HelpFormatter):
    def _format_action(self, action):
        if type(action) == argparse._SubParsersAction:
            # inject new class variable for subcommand formatting
            subactions = action._get_subactions()
            invocations = [self._format_action_invocation(a) for a in subactions]
            self._subcommand_max_length = max(len(i) for i in invocations)

        if type(action) == argparse._SubParsersAction._ChoicesPseudoAction:
            # format subcommand help line
            subcommand = self._format_action_invocation(action) # type: str
            width = self._subcommand_max_length
            help_text = ""
            if action.help:
                help_text = self._expand_help(action)
            return "  {:{width}} -  {}\n".format(subcommand, help_text, width=width)

        elif type(action) == argparse._SubParsersAction:
            # process subcommand help section
            msg = '\n'
            for subaction in action._get_subactions():
                msg += self._format_action(subaction)
            return msg
        else:
            return super(CustomHelpFormatter, self)._format_action(action)


def check():
    print("status")
    return 0

parser = argparse.ArgumentParser(usage="sub <command>", add_help=False,
             formatter_class=CustomHelpFormatter)

subparser = parser.add_subparsers(dest="cmd")
subparser.add_parser('status', help='show status')
subparser.add_parser('list', help='print list')

# custom help messge
parser._positionals.title = "commands"

# hack to show help when no arguments supplied
if len(sys.argv) == 1:
    parser.print_help()
    sys.exit(0)

args = parser.parse_args()
```

```
if args.cmd == 'list':
    print('list')
elif args.cmd == 'status':
    sys.exit(check())
```

CLI https://riptutorial.com/zh-TW/python/topic/7701/cli

82:

- value1 ** value2
- powvalue1value2 [value3]
- value1 .__ pow __value2 [value3]
- 2 .__ rpow __1
- operator.powvalue1value2
- operator .__ pow __value1value2
- math.powvalue1value2
- math.sqrt
- math.exp
- cmath.exp
- math.expm1

Examples

math.sqrtcmath.sqrt

`mathmath.sqrt()float float`

```
import math

math.sqrt(9)                # 3.0
math.sqrt(11.11)            # 3.3331666624997918
math.sqrt(Decimal('6.25'))  # 2.5
```

`complex math.sqrt()ValueError`

```
import math

math.sqrt(9)                # 3.0
math.sqrt(11.11)            # 3.3331666624997918
math.sqrt(Decimal('6.25'))  # 2.5
```

ValueError

`math.sqrt(x)math.pow(x, 0.5)x ** 0.5 math.pow(x, 0.5)。 cmathmath`+ bi。 `.sqrt()`

```
import math

math.sqrt(9)                # 3.0
math.sqrt(11.11)            # 3.3331666624997918
math.sqrt(Decimal('6.25'))  # 2.5
```

`j j`-1。 a + bia + bj。 `a2+0j2+0j。 b`0. `b2j`2。 `2j 2j0 + 2j 。`

**pow

`pow` -function`**`

```
2 ** 3    # 8
pow(2, 3) # 8
```

Python 2.x。 `**` ;

- Base `int` exponent `int < 0`

  ```
  2 ** 3    # 8
  pow(2, 3) # 8
  ```

- Python 3.x。
- Python 2.2.0`ValueError` 。
- Base `int < 0float < 0` exponent `float != int`

  ```
  2 ** 3    # 8
  pow(2, 3) # 8
  ```

- python 3.0.0`ValueError` 。

`operator**` `operator`

```
2 ** 3    # 8
pow(2, 3) # 8
```

`__pow__`

```
2 ** 3    # 8
pow(2, 3) # 8
```

math.pow

`math` -module`math.pow()` 。 `pow()**float`

```
import math
math.pow(2, 2)    # 4.0
math.pow(-2., 2)  # 4.0
```

```
import math
math.pow(2, 2)    # 4.0
math.pow(-2., 2)  # 4.0
```

TypeErrorcomplexfloat

```
import math
math.pow(2, 2)    # 4.0
```

```
math.pow(-2., 2)  # 4.0
```

ValueError

math.expcmath.exp

`mathcmath` -moduleEulere`pow()` `** math.exp()` `math.exp()`

```
import math

math.e ** 2  # 7.3890560989306495
math.exp(2)  # 7.38905609893065

import cmath
cmath.e ** 2 # 7.3890560989306495
cmath.exp(2) # (7.38905609893065+0j)
```

`math.e`

```
import math

math.e ** 2  # 7.3890560989306495
math.exp(2)  # 7.38905609893065

import cmath
cmath.e ** 2 # 7.3890560989306495
cmath.exp(2) # (7.38905609893065+0j)
```

1math.expm1

`mathexpm1()` `xmath.e ** x - 1` `math.exp(x)` `cmath.exp(x)`

```
import math

print(math.e ** 1e-3 - 1)  # 0.0010005001667083846
print(math.exp(1e-3) - 1)  # 0.0010005001667083846
print(math.expm1(1e-3))    # 0.0010005001667083417
#                            ------------------^
```

x

```
import math

print(math.e ** 1e-3 - 1)  # 0.0010005001667083846
print(math.exp(1e-3) - 1)  # 0.0010005001667083846
print(math.expm1(1e-3))    # 0.0010005001667083417
#                            ------------------^
```

- 1

```
import math

print(math.e ** 1e-3 - 1)  # 0.0010005001667083846
```

```
print(math.exp(1e-3) - 1)  # 0.0010005001667083846
print(math.expm1(1e-3))    # 0.0010005001667083417
#                            ------------------^
```

cmath

```
class Integer(object):
    def __init__(self, value):
        self.value = int(value) # Cast to an integer

    def __repr__(self):
        return '{cls}({val})'.format(cls=self.__class__.__name__,
                                     val=self.value)

    def __pow__(self, other, modulo=None):
        if modulo is None:
            print('Using __pow__')
            return self.__class__(self.value ** other)
        else:
            print('Using __pow__ with modulo')
            return self.__class__(pow(self.value, other, modulo))

    def __float__(self):
        print('Using __float__')
        return float(self.value)

    def __complex__(self):
        print('Using __complex__')
        return complex(self.value, 0)
```

`pow**__pow__`

```
class Integer(object):
    def __init__(self, value):
        self.value = int(value) # Cast to an integer

    def __repr__(self):
        return '{cls}({val})'.format(cls=self.__class__.__name__,
                                     val=self.value)

    def __pow__(self, other, modulo=None):
        if modulo is None:
            print('Using __pow__')
            return self.__class__(self.value ** other)
        else:
            print('Using __pow__ with modulo')
            return self.__class__(pow(self.value, other, modulo))

    def __float__(self):
        print('Using __float__')
        return float(self.value)

    def __complex__(self):
        print('Using __complex__')
        return complex(self.value, 0)
```

`__pow__()` builtin- `pow()`

```
class Integer(object):
    def __init__(self, value):
        self.value = int(value) # Cast to an integer

    def __repr__(self):
        return '{cls}({val})'.format(cls=self.__class__.__name__,
                                     val=self.value)

    def __pow__(self, other, modulo=None):
        if modulo is None:
            print('Using __pow__')
            return self.__class__(self.value ** other)
        else:
            print('Using __pow__ with modulo')
            return self.__class__(pow(self.value, other, modulo))

    def __float__(self):
        print('Using __float__')
        return float(self.value)

    def __complex__(self):
        print('Using __complex__')
        return complex(self.value, 0)
```

`mathfloat`

```
class Integer(object):
    def __init__(self, value):
        self.value = int(value) # Cast to an integer

    def __repr__(self):
        return '{cls}({val})'.format(cls=self.__class__.__name__,
                                     val=self.value)

    def __pow__(self, other, modulo=None):
        if modulo is None:
            print('Using __pow__')
            return self.__class__(self.value ** other)
        else:
            print('Using __pow__ with modulo')
            return self.__class__(pow(self.value, other, modulo))

    def __float__(self):
        print('Using __float__')
        return float(self.value)

    def __complex__(self):
        print('Using __complex__')
        return complex(self.value, 0)
```

`cmath` -functions`complexcomplexfloat`

```
class Integer(object):
    def __init__(self, value):
        self.value = int(value) # Cast to an integer

    def __repr__(self):
        return '{cls}({val})'.format(cls=self.__class__.__name__,
                                     val=self.value)
```

```
    def __pow__(self, other, modulo=None):
        if modulo is None:
            print('Using __pow__')
            return self.__class__(self.value ** other)
        else:
            print('Using __pow__ with modulo')
            return self.__class__(pow(self.value, other, modulo))

    def __float__(self):
        print('Using __float__')
        return float(self.value)

    def __complex__(self):
        print('Using __complex__')
        return complex(self.value, 0)
```

`__float__()` - mathcmath

```
class Integer(object):
    def __init__(self, value):
        self.value = int(value) # Cast to an integer

    def __repr__(self):
        return '{cls}({val})'.format(cls=self.__class__.__name__,
                                     val=self.value)

    def __pow__(self, other, modulo=None):
        if modulo is None:
            print('Using __pow__')
            return self.__class__(self.value ** other)
        else:
            print('Using __pow__ with modulo')
            return self.__class__(pow(self.value, other, modulo))

    def __float__(self):
        print('Using __float__')
        return float(self.value)

    def __complex__(self):
        print('Using __complex__')
        return complex(self.value, 0)
```

TypeErrorfloat

3pow

3`pow()` `pow(a, b, c)` $a^b \bmod c$

```
pow(3, 4, 17)   # 13

# equivalent unoptimized expression:
3 ** 4 % 17     # 13

# steps:
3 ** 4          # 81
81 % 17         # 13
```

- `int`
- `int >= 0`
- `int != 0`

python 3.x

3`pow`

```
pow(3, 4, 17)   # 13

# equivalent unoptimized expression:
3 ** 4 % 17     # 13

# steps:
3 ** 4          # 81
81 % 17         # 13
```

nth-root

`math.sqrt` `**` n。

。 31/3。

```
>>> x = 3
>>> y = x ** 3
>>> y
27
>>> z = y ** (1.0 / 3)
>>> z
3.0
>>> z == x
True
```

PythonPythonn。

```
x = 2 ** 100
cube = x ** 3
root = cube ** (1.0 / 3)
```

OverflowErrorlong intfloat

n。

```
x = 2 ** 100
cube = x ** 3
root = cube ** (1.0 / 3)
```

https://riptutorial.com/zh-TW/python/topic/347/

83:

Examples

py2app

py2app。

```
sudo easy_install -U py2app
```

`pip`

```
sudo easy_install -U py2app
```

python

```
sudo easy_install -U py2app
```

```
sudo easy_install -U py2app
```

.icns

```
sudo easy_install -U py2app
```

```
sudo easy_install -U py2app
```

dist。

```
sudo easy_install -U py2app
```

cx_Freeze

cx_Freeze

```
python setup.py build
sudo python setup.py install
```

python**“setup.py”**

```
python setup.py build
sudo python setup.py install
```

setup.py

```
python setup.py build
sudo python setup.py install
```

El CapitanrootSIP。

https://riptutorial.com/zh-TW/python/topic/2026/

84:

- a [startend]itemsend-1
- a [start]items
- a [end]1
- a []
- a []
-

- `lst[::-1]`
- `startend◦`

```
a[-1]    # last item in the array
a[-2:]   # last two items in the array
a[:-2]   # everything except the last two items
```

Examples

“”

```
lst = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']

lst[::2]
# Output: ['a', 'c', 'e', 'g']

lst[::3]
# Output: ['a', 'd', 'g']
```

```
lst = ['a', 'b', 'c', 'd', 'e']

lst[2:4]
# Output: ['c', 'd']

lst[2:]
# Output: ['c', 'd', 'e']

lst[:4]
# Output: ['a', 'b', 'c', 'd']
```

```
a = [1, 2, 3, 4, 5]

# steps through the list backwards (step=-1)
b = a[::-1]

# built-in list method to reverse 'a'
a.reverse()

if a = b:
    print(True)

print(b)
```

```
# Output:
# True
# [5, 4, 3, 2, 1]
```

```
def shift_list(array, s):
    """Shifts the elements of a list to the left or right.

    Args:
        array - the list to shift
        s - the amount to shift the list ('+': right-shift, '-': left-shift)

    Returns:
        shifted_array - the shifted list
    """
    # calculate actual shift amount (e.g., 11 --> 1 if length of the array is 5)
    s %= len(array)

    # reverse the shift direction to be more intuitive
    s *= -1

    # shift array with list slicing
    shifted_array = array[s:] + array[:s]

    return shifted_array

my_array = [1, 2, 3, 4, 5]

# negative numbers
shift_list(my_array, -7)
>>> [3, 4, 5, 1, 2]

# no shift on numbers equal to the size of the array
shift_list(my_array, 5)
>>> [1, 2, 3, 4, 5]

# works on positive numbers
shift_list(my_array, 3)
>>> [3, 4, 5, 1, 2]
```

https://riptutorial.com/zh-TW/python/topic/1494/--

85:

Python。 。 。

- [x1 for x in1,2,3]list comprehension[2,3,4]
- x + 1x in1,2,3generator234
- [x for x in1,2,3if x2 == 0] #list comprehension with filtergive [2]
- [x + 1 if x2 == 0 else x for x in1,2,3] #list comprehension with ternary
- [x + 1 if x2 == 0 else x for x in range-3,4if x> 0] #list comprehension with ternary and filtering
- {x for x in1,2,2,3}set comprehension{1,2,3}
- {kvkv['a'1'b'2]} #dict comprehension{'a'1'b'2}python 2.7+3.0+
- [xy[1,2]x [10,20]][11,21,12,22]
- [xy[1,2,3]xx> 2[3,4,5]y1st
- [xy[1,2,3]x[3,4,5]yx> 2]2
- [x for x in xrange10if x2 == 0]

。 。

-
-
- lambda
- 。 `[print(x) for x in range(10)]`

Examples

iterable`list` 。

```
[ <expression> for <element> in <iterable> ]
```

'if'

```
[ <expression> for <element> in <iterable> ]
```

`<condition>` true `<iterable>` `<element><element><expression>` 。 。 - 。

`list`

```
[ <expression> for <element> in <iterable> ]
```

`for(1, 2, 3, 4)x。 x * xlist 。 listsquares 。`

for

```
[ <expression> for <element> in <iterable> ]
```

```
[ <expression> for <element> in <iterable> ]
```

---

elseList comprehension。 if / elseforfor

```
[ <expression> for <element> in <iterable> ]
```

。 for…inif 。

[... for x in ... for y in ...]。 for

```
[ <expression> for <element> in <iterable> ]
```

```
[ <expression> for <element> in <iterable> ]
```

[str(x) for i in range(3) for x in foo(i)]

None 。

。 。 。

list.sort() None 。

```
[ <expression> for <element> in <iterable> ]
```

sorted()list

```
[ <expression> for <element> in <iterable> ]
```

I / O。 for。 Python 3

```
[ <expression> for <element> in <iterable> ]
```

```
[ <expression> for <element> in <iterable> ]
```

。 random.randrange()。 next() 。

```
[ <expression> for <element> in <iterable> ]
```

。

```
[ <expression> for <element> in <iterable> ]
```

。

Python 2.x 2.7

```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```


```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```


---

dictdict。

Python 2.x 2.7


```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```


。


```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```

Python 2.x 2.7


```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```


---

*hashable*dict


```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```


- `swapped = {v: k for k, v in my_dict.items()}`
- `swapped = dict((v, k) for k, v in my_dict.iteritems())`
- `swapped = dict(zip(my_dict.values(), my_dict))`
- `swapped = dict(zip(my_dict.values(), my_dict.keys()))`
- `swapped = dict(map(reversed, my_dict.items()))`


```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```


Python 2.x 2.3

itertools`izipimap` 。

。

```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```

PEP 448 。

Python 3.x 3.5

```
{x: x * x for x in (1, 2, 3, 4)}
# Out: {1: 1, 2: 4, 3: 9, 4: 16}
```

Python 3.02.7+2.0。 <2.7`dict()`。

。 ; 。

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Python 2.x 2.4

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

- list comprehension`list`generator comprehension`generator` 。
- `generator`next。

`xrange`。 。 `xrange`python 2。 python 3 `range`。 *xrange* 。

Python 2.x 2.4

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Python 3.x 3.0

`g.next()``next(g)``xrange`` range`` iterator.next()`` xrange()`Python 3。

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

Python 2.x 2.4

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

。

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

iterable。 `get_objects()`API。 。 。

```
# list comprehension
[x**2 for x in range(10)]
# Output: [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

。

Python 2.x 2.7

```
# A set containing every value in range(5):
{x for x in range(5)}
# Out: {0, 1, 2, 3, 4}

# A set of even numbers between 1 and 10:
{x for x in range(1, 11) if x % 2 == 0}
# Out: {2, 4, 6, 8, 10}

# Unique alphabetic characters in a string of text:
text = "When in the Course of human events it becomes necessary for one people..."
{ch.lower() for ch in text if ch.isalpha()}
# Out: set(['a', 'c', 'b', 'e', 'f', 'i', 'h', 'm', 'l', 'o',
#           'n', 'p', 's', 'r', 'u', 't', 'w', 'v', 'y'])
```

。 。

python 2.7+2.0。 Python 2.2Python 2.6 `set()`

Python 2.x 2.2

```
# A set containing every value in range(5):
{x for x in range(5)}
```

```
# Out: {0, 1, 2, 3, 4}

# A set of even numbers between 1 and 10:
{x for x in range(1, 11) if x % 2 == 0}
# Out: {2, 4, 6, 8, 10}

# Unique alphabetic characters in a string of text:
text = "When in the Course of human events it becomes necessary for one people..."
{ch.lower() for ch in text if ch.isalpha()}
# Out: set(['a', 'c', 'b', 'e', 'f', 'i', 'h', 'm', 'l', 'o',
#           'n', 'p', 's', 'r', 'u', 't', 'w', 'v', 'y'])
```

```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```

f(x)1000xif◦ f(x)◦ f()◦

x


```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```

```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```


v ◦ viterable


```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```

◦

```
>>> def f(x):
...     import time
```

```
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```

`f(x)` `@functools.lru_cache()` Python 3.2+ `f(x)` `x` `f`。 memoization。

```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```


```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```


。


```
>>> def f(x):
...     import time
...     time.sleep(.1)       # Simulate expensive function
...     return x**2

>>> [f(x) for x in range(1000) if f(x) > 10]
[16, 25, 36, ...]
```


L+OL ^ 2 - 。 LIL-1IL-2;Ix1LI *L ** 2/ 2。

。

`for`

```
[x + y for x, y in [(1, 2), (3, 4), (5, 6)]]
# Out: [3, 7, 11]

[x + y for x, y in zip([1, 3, 5], [2, 4, 6])]
# Out: [3, 7, 11]
```


`for`

```
[x + y for x, y in [(1, 2), (3, 4), (5, 6)]]
# Out: [3, 7, 11]

[x + y for x, y in zip([1, 3, 5], [2, 4, 6])]
# Out: [3, 7, 11]
```

```
[x + y for x, y in [(1, 2), (3, 4), (5, 6)]]
# Out: [3, 7, 11]

[x + y for x, y in zip([1, 3, 5], [2, 4, 6])]
# Out: [3, 7, 11]
```

```
# Count the numbers in `range(1000)` that are even and contain the digit `9`:
print (sum(
    1 for x in range(1000)
    if x % 2 == 0 and
    '9' in str(x)
))
# Out: 95
```

1. `range(1000)`。
2. `if`。
3. 11。
4. `1`。

1 sum。 。

。 List Comprehension`map()`。

```
# Convert a list of strings to integers.
items = ["1","2","3","4"]
[int(item) for item in items]
# Out: [1, 2, 3, 4]

# Convert a list of strings to float.
items = ["1","2","3","4"]
map(float, items)
# Out:[1.0, 2.0, 3.0, 4.0]
```

https://riptutorial.com/zh-TW/python/topic/196/

86:

110 `[i ** 2 for i in range(1,11)]irange。` for。

- [i for i in range10] #basic list comprehension
- [i for i in xrange10]python 2.x
- [i for i in range20if i2 == 0] #with filter with
- [xy[1,2,3]x[3,4,5]]y
- [i if i> 6 else 0 for i in range10] #ternary expression
- [i if i> 4 else 0 for i in range20if i2 == 0] #with filter and threenary expression
- [3,4,5][[x + y[1,2,3]x] #nested list comprehension

PEP 202Python 2.0。

Examples

`if。`

```
[<expression> for <element> in <iterable> if <condition>]
```

`<iterable><element> <iterable> ;<condition>True <expression> <element>。`

```
[<expression> for <element> in <iterable> if <condition>]
```

```
[<expression> for <element> in <iterable> if <condition>]
```

`[e for x in y if c] ecxlist(filter(lambda x: c, map(lambda x: e, y))) 。`

2。 。

`... if ... else ... <expression>。`

```
[<expression> for <element> in <iterable> if <condition>]
```

```
[<expression> for <element> in <iterable> if <condition>]
```

```
[<expression> for <element> in <iterable> if <condition>]
```

Python 2.7 `xrangerangexrange` 。

```
[<expression> for <element> in <iterable> if <condition>]
```

```
[<expression> for <element> in <iterable> if <condition>]
```

---

if。

```
[<expression> for <element> in <iterable> if <condition>]
```

```
[<expression> for <element> in <iterable> if <condition>]
```

。

List Comprehensionsfor。 forforif。 forfor。

```
[ expression for target1 in iterable1 [if condition1]
             for target2 in iterable2 [if condition2]...
             for targetN in iterableN [if conditionN] ]
```

for

```
[ expression for target1 in iterable1 [if condition1]
             for target2 in iterable2 [if condition2]...
             for targetN in iterableN [if conditionN] ]
```

for

```
[ expression for target1 in iterable1 [if condition1]
             for target2 in iterable2 [if condition2]...
             for targetN in iterableN [if conditionN] ]
```

。

。

```
[ expression for target1 in iterable1 [if condition1]
             for target2 in iterable2 [if condition2]...
             for targetN in iterableN [if conditionN] ]
```

140ns 。

---

if Sfor

```
[ expression for target1 in iterable1 [if condition1]
             for target2 in iterable2 [if condition2]...
             for targetN in iterableN [if conditionN] ]
```

for 。 2/。 。

`filtermap`。 Guido Van Rossum 2005

`filter(P, S)` `[x for x in S if P(x)]` `x==42` lambdalambda。 `map(F, S)` `map(F, S)` `[F(x) for x in S]` `map(F, S)` `[F(x) for x in S]` 。 。

“ *pythonic* ”python linters。

```
filter(lambda x: x % 2 == 0, range(10)) # even numbers < 10
map(lambda x: 2*x, range(10)) # multiply each number by two
reduce(lambda x,y: x+y, range(10)) # sum of all elements in list
```

`filtermap` ;*lambda* - 。

```
filter(lambda x: x % 2 == 0, range(10)) # even numbers < 10
map(lambda x: 2*x, range(10)) # multiply each number by two
reduce(lambda x,y: x+y, range(10)) # sum of all elements in list
```

。 ;。 MapFilter。

```
filter(lambda x: x % 2 == 0, range(10)) # even numbers < 10
map(lambda x: 2*x, range(10)) # multiply each number by two
reduce(lambda x,y: x+y, range(10)) # sum of all elements in list
```

-

- ```
  filter(lambda x: x % 2 == 0, range(10)) # even numbers < 10
  map(lambda x: 2*x, range(10)) # multiply each number by two
  reduce(lambda x,y: x+y, range(10)) # sum of all elements in list
  ```

- ```
  filter(lambda x: x % 2 == 0, range(10)) # even numbers < 10
  map(lambda x: 2*x, range(10)) # multiply each number by two
  reduce(lambda x,y: x+y, range(10)) # sum of all elements in list
  ```

FPbool

。 。

```
#List Comprehension with nested loop
[x + y for x in [1, 2, 3] for y in [3, 4, 5]]
#Out: [4, 5, 6, 5, 6, 7, 6, 7, 8]

#Nested List Comprehension
[[x + y for x in [1, 2, 3]] for y in [3, 4, 5]]
#Out: [[4, 5, 6], [5, 6, 7], [6, 7, 8]]
```

```
#List Comprehension with nested loop
[x + y for x in [1, 2, 3] for y in [3, 4, 5]]
#Out: [4, 5, 6, 5, 6, 7, 6, 7, 8]
```

```
#Nested List Comprehension
[[x + y for x in [1, 2, 3]] for y in [3, 4, 5]]
#Out: [[4, 5, 6], [5, 6, 7], [6, 7, 8]]
```

。

```
#List Comprehension with nested loop
[x + y for x in [1, 2, 3] for y in [3, 4, 5]]
#Out: [4, 5, 6, 5, 6, 7, 6, 7, 8]

#Nested List Comprehension
[[x + y for x in [1, 2, 3]] for y in [3, 4, 5]]
#Out: [[4, 5, 6], [5, 6, 7], [6, 7, 8]]
```

forfor。


```
#List Comprehension with nested loop
[x + y for x in [1, 2, 3] for y in [3, 4, 5]]
#Out: [4, 5, 6, 5, 6, 7, 6, 7, 8]

#Nested List Comprehension
[[x + y for x in [1, 2, 3]] for y in [3, 4, 5]]
#Out: [[4, 5, 6], [5, 6, 7], [6, 7, 8]]
```


zip()

```
>>> list_1 = [1, 2, 3 , 4]
>>> list_2 = ['a', 'b', 'c', 'd']
>>> list_3 = ['6', '7', '8', '9']

# Two lists
>>> [(i, j) for i, j in zip(list_1, list_2)]
[(1, 'a'), (2, 'b'), (3, 'c'), (4, 'd')]

# Three lists
>>> [(i, j, k) for i, j, k in zip(list_1, list_2, list_3)]
[(1, 'a', '6'), (2, 'b', '7'), (3, 'c', '8'), (4, 'd', '9')]

# so on ...
```

https://riptutorial.com/zh-TW/python/topic/5265/

87:

Examples

“”Iterable

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

Python 3.x 3.0

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

_° _ °

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
```

```
# Prints: 2
```

Python 3.x 3.0

*_

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

。

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

```
a, b = (1, 2)
print(a)
# Prints: 1
print(b)
# Prints: 2
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

listdict。

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

*

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

- `**`Python`dict`

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

iterable

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

```
def fun1(arg1, arg2, arg3):
    return (arg1,arg2,arg3)
```

“”

```
def fun1(*args, **kwargs):
    print(args, kwargs)
```

`*args`**`kwargs`tupledict

```
def fun1(*args, **kwargs):
    print(args, kwargs)
```

Python。

```
def fun1(*args, **kwargs):
```

```
    print(args, kwargs)
```

https://riptutorial.com/zh-TW/python/topic/4282/--

88:

Examples

IPython%% timeittimeit

```
In [1]: import string

In [2]: %%timeit s=""; long_list=list(string.ascii_letters)*50
  ....: for substring in long_list:
  ....:   s+=substring
  ....:
1000 loops, best of 3: 570 us per loop

In [3]: %%timeit long_list=list(string.ascii_letters)*50
  ....: s="".join(long_list)
  ....:
100000 loops, best of 3: 16.1 us per loop
```


```
In [1]: import string

In [2]: %%timeit s=""; long_list=list(string.ascii_letters)*50
  ....: for substring in long_list:
  ....:   s+=substring
  ....:
1000 loops, best of 3: 570 us per loop

In [3]: %%timeit long_list=list(string.ascii_letters)*50
  ....: s="".join(long_list)
  ....:
100000 loops, best of 3: 16.1 us per loop
```


timeit

```
>>> import timeit
>>> timeit.timeit('list(itertools.repeat("a", 100))', 'import itertools', number = 10000000)
10.997665435877963
>>> timeit.timeit('["a"]*100', number = 10000000)
7.118789926862576
```

timeit

```
python -m timeit "'-'.join(str(n) for n in range(100))"
10000 loops, best of 3: 29.2 usec per loop

python -m timeit "'-'.join(map(str,range(100)))"
100000 loops, best of 3: 19.4 usec per loop
```

line_profiler

@profile

```
import requests

@profile
def slow_func():
    s = requests.session()
    html=s.get("https://en.wikipedia.org/").text
    sum([pow(ord(x),3.1) for x in list(html)])

for i in range(50):
    slow_func()
```

kernprof

```
import requests

@profile
def slow_func():
    s = requests.session()
    html=s.get("https://en.wikipedia.org/").text
    sum([pow(ord(x),3.1) for x in list(html)])

for i in range(50):
    slow_func()
```

。

cProfileProfiler

PythoncProfile。 timeit。

- `ncalls`
- `tottime`
- `percall` 。 tottimencalls
- `cumtime` 。 。
- `percall` cumtime
- `filename:lineno(function)`

cProfiler

```
$ python -m cProfile main.py
```


```
$ python -m cProfile main.py
```


https://riptutorial.com/zh-TW/python/topic/3818/

89: Python

pypa`setup.py` ◦

Examples

`setup.py`◦

```
+-- package_name
|       |
|       +-- __init__.py
|
+-- setup.py
```

`__init__.pydef foo(): return 100` ◦

`setup.py`

```
+-- package_name
|       |
|       +-- __init__.py
|
+-- setup.py
```

virtualenvPython

```
+-- package_name
|       |
|       +-- __init__.py
|
+-- setup.py
```

PyPI

`setup.py` PyPI ◦

.pypirc

◦ ◦

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
```

```
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

twine 。

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

testpypi

PyPI testpypi。 。 。

testpypi。 。

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

```
# .pypirc file

[distutils]
```

```
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

。

。 testpypiPyPI `pip install`。

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

。 PyPIAPI。 。 testpypi。

PyPI

`twine`

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password
```

```
[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

PyPI。

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

。

。

。 PyPireStructuredText 。

README.rst。 PyPi。

setup.cfg

```
# .pypirc file

[distutils]
index-servers =
  pypi
  pypitest

[pypi]
repository=https://pypi.python.org/pypi
username=your_username
password=your_password

[pypitest]
repository=https://testpypi.python.org/pypi
username=your_username
password=your_password
```

MarkdownPyPi。

`LICENSE.txt`OpenSource 。

TL; DR。

`__main__.py`。

`__main__.pypackage_name`。

```
python -m package_name
```

`__main__.py`

python`package_name.__main__`;`package_name.__main__`; 'package_name'。

Python https://riptutorial.com/zh-TW/python/topic/1381/python

90:

Python。 。 Python。

Python`print()` `input()` `len()` 。 - 。

- def *function_name arg1... argN* argskw1kw2 = default...** kwargs*
- lambda *arg1... argN* argskw1kw2 = default...** kwargs expression*

arg1 ... argN	
** ARGS*	
KW1...KWN	
*** kwargs*	

5

- ```
  def f():
    print(20)
  y = f
  y()
  # Output: 20
  ```

- ```
  def f():
    print(20)
  y = f
  y()
  # Output: 20
  ```

- ```
  def f():
    print(20)
  y = f
  y()
  # Output: 20
  ```

- ```
  def f():
    print(20)
  y = f
  y()
  # Output: 20
  ```

- Closure

  ```
  def f():
    print(20)
  y = f
  ```

```
y()
# Output: 20
```

- https //www.thecodeship.com/patterns/guide-to-python-function-decorators/

Examples

`def`python。

```
def function_name(parameters):
    statement(s)
```

`function_name`function_name 。 。

`parameters`。 。

`statement(s)` - - 。 。

`Hello`

```
def function_name(parameters):
    statement(s)
```

`greet()`

```
def function_name(parameters):
    statement(s)
```

```
def function_name(parameters):
    statement(s)
```

`greet_two()`

```
def function_name(parameters):
    statement(s)
```

```
def function_name(parameters):
    statement(s)
```

```
def function_name(parameters):
    statement(s)
```

。 Python`return`return。

```
def function_name(parameters):
    statement(s)
```

Python。

returnNone

```
def function_name(parameters):
    statement(s)
```

◦ pass - ◦ ◦ ◦

return

```
def give_me_five():
    return 5

print(give_me_five())  # Print the returned value
# Out: 5
```

```
def give_me_five():
    return 5

print(give_me_five())  # Print the returned value
# Out: 5
```

```
def give_me_five():
    return 5

print(give_me_five())  # Print the returned value
# Out: 5
```

return

```
def give_me_five():
    return 5

print(give_me_five())  # Print the returned value
# Out: 5
```

return

```
def give_me_five():
    return 5

print(give_me_five())  # Print the returned value
# Out: 5
```

returnNone ◦ returnNone ◦

```
def divide(dividend, divisor):  # The names of the function and its arguments
    # The arguments are available by name in the body of the function
    print(dividend / divisor)
```

◦ ◦

```
def divide(dividend, divisor):  # The names of the function and its arguments
```

```
    # The arguments are available by name in the body of the function
    print(dividend / divisor)
```

```
def divide(dividend, divisor):  # The names of the function and its arguments
    # The arguments are available by name in the body of the function
    print(dividend / divisor)
```

= argument-name

```
def make(action='nothing'):
    return action
```

3

```
def make(action='nothing'):
    return action
```

list dict set◦ ◦ ◦

```
def func(value1, value2, optionalvalue=10):
    return '{0} {1} {2}'.format(value1, value2, optionalvalue1)
```

```
def func(value1, value2, optionalvalue=10):
    return '{0} {1} {2}'.format(value1, value2, optionalvalue1)
```

◦

```
def func(value1, value2, optionalvalue=10):
    return '{0} {1} {2}'.format(value1, value2, optionalvalue1)
```

*

```
def func(*args):
    # args will be a tuple containing all values that are passed in
    for i in args:
        print(i)

func(1, 2, 3)  # Calling it with 3 arguments
# Out: 1
#      2
#      3

list_of_arg_values = [1, 2, 3]
func(*list_of_arg_values)  # Calling it with list of values, * expands the list
# Out: 1
#      2
#      3

func()  # Calling it without arguments
```

```
# No Output
```

argsfunc(*args=[1, 2, 3])。

func(*args=[1, 2, 3])TypeError。

Iterable func(*my_stuff)。

*args args[0]

---

*

```
def func(*args):
    # args will be a tuple containing all values that are passed in
    for i in args:
        print(i)

func(1, 2, 3)  # Calling it with 3 arguments
# Out: 1
#      2
#      3

list_of_arg_values = [1, 2, 3]
func(*list_of_arg_values)  # Calling it with list of values, * expands the list
# Out: 1
#      2
#      3

func()  # Calling it without arguments
# No Output
```

func(1, 2, 3)TypeError。

kwargspython。 args['value1']value1。 KeyError。

。

/。 。

*arg。 。

。 。

**kwargs。 。

```
def func(*args):
    # args will be a tuple containing all values that are passed in
    for i in args:
        print(i)

func(1, 2, 3)  # Calling it with 3 arguments
# Out: 1
#      2
#      3
```

```
list_of_arg_values = [1, 2, 3]
func(*list_of_arg_values)  # Calling it with list of values, * expands the list
# Out: 1
#      2
#      3

func()  # Calling it without arguments
# No Output
```

- `arg1` `TypeError` 。 `func(10)` `func(arg1=10)` 。
- `kwarg1` keyword-argument `func(kwarg1=10)` 。
- `arg2kwarg2`。 `arg1` `kwarg1` 。
- `*args`。 `arg1arg2*args` `func(1, 1, 1, 1)` 。
- `**kwargs`。 `arg1` `arg2` `kwarg1kwarg2` 。 `func(kwarg3=10)` 。
- Python 3`*`。 Python 3.5`math.isclosedef math.isclose (a, b, *, rel_tol=1e-09, abs_tol=0.0)` 。

Python 2.x。 `kwargs`

```
def func(*args):
    # args will be a tuple containing all values that are passed in
    for i in args:
        print(i)

func(1, 2, 3)  # Calling it with 3 arguments
# Out: 1
#      2
#      3

list_of_arg_values = [1, 2, 3]
func(*list_of_arg_values)  # Calling it with list of values, * expands the list
# Out: 1
#      2
#      3

func()  # Calling it without arguments
# No Output
```

`argskwargs` 。

noneone `*args`**none** `**kwargs`。 `*args` `**kwargs`。 。

`***`argskwargs。

```
def func(*args):
    # args will be a tuple containing all values that are passed in
    for i in args:
        print(i)

func(1, 2, 3)  # Calling it with 3 arguments
# Out: 1
#      2
#      3
```

```
list_of_arg_values = [1, 2, 3]
func(*list_of_arg_values)  # Calling it with list of values, * expands the list
# Out: 1
#      2
#      3

func()  # Calling it without arguments
# No Output
```

。

。 __defaults__。

```
def f(a, b=42, c=[]):
    pass

print(f.__defaults__)
# Out: (42, [])
```

Argument ;。 。 。 。

```
def f(a, b=42, c=[]):
    pass

print(f.__defaults__)
# Out: (42, [])
```

IDEPyCharm。

。

None None 。

```
def f(a, b=42, c=[]):
    pass

print(f.__defaults__)
# Out: (42, [])
```

Lambda/

lambda。 。

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

```
def greeting():
```

```
    return "Hello"
```

lambda

```
def greeting():
    return "Hello"
```

lambdas。 。

greet_meHello。 lambdareturn。 :。

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

lambda

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

/。

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

lambda Ssorted filtermap。

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

map

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

lambda/。

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

`lambda`。

PEP-8 Pythonlambda

deflambda。

```
def greeting():
    return "Hello"
```

```
def greeting():
    return "Hello"
```

`f<lambda>` 。 。 lambda`def`。

- **argument** ;
- **parameter** 。

Python //。

- 。

```
def foo(x):         # here x is the parameter
    x[0] = 9        # This mutates the list labelled by both x and y
    print(x)

y = [4, 5, 6]
foo(y)              # call foo with y as argument
# Out: [9, 5, 6]    # list labelled by x has been mutated
print(y)
# Out: [9, 5, 6]    # list labelled by y has been mutated too
```

- 。

```
def foo(x):         # here x is the parameter
    x[0] = 9        # This mutates the list labelled by both x and y
    print(x)

y = [4, 5, 6]
foo(y)              # call foo with y as argument
# Out: [9, 5, 6]    # list labelled by x has been mutated
print(y)
# Out: [9, 5, 6]    # list labelled by y has been mutated too
```

Python 。

- 。 。
- 。 。

```
def foo(x):         # here x is the parameter
    x[0] = 9        # This mutates the list labelled by both x and y
    print(x)

y = [4, 5, 6]
foo(y)              # call foo with y as argument
# Out: [9, 5, 6]    # list labelled by x has been mutated
print(y)
# Out: [9, 5, 6]    # list labelled by y has been mutated too
```

Python。 `makeIncincx`。 `makeIncx`。

```
def makeInc(x):
  def inc(y):
     # x is "attached" in the definition of inc
     return y + x

  return inc

incOne = makeInc(1)
incFive = makeInc(5)

incOne(5) # returns 6
incFive(5) # returns 10
```

```
def makeInc(x):
  def inc(y):
     # x is "attached" in the definition of inc
     return y + x

  return inc

incOne = makeInc(1)
incFive = makeInc(5)

incOne(5) # returns 6
incFive(5) # returns 10
```

Python 3`nonlocal` Nonlocal Variables 。

Python 3.x 3.0

```
def makeInc(x):
  def inc(y):
     # x is "attached" in the definition of inc
     return y + x

  return inc

incOne = makeInc(1)
incFive = makeInc(5)
```

```
incOne(5) # returns 6
incFive(5) # returns 10
```

◦ `factorial(n) = n*(n-1)*(n-2)*...*3*2*1`factorial◦

```
def factorial(n):
    #n here should be an integer
    if n == 0:
        return 1
    else:
        return n*factorial(n-1)
```

```
def factorial(n):
    #n here should be an integer
    if n == 0:
        return 1
    else:
        return n*factorial(n-1)
```

◦ `return factorial(n-1)` ◦

lambdalambda

```
def factorial(n):
    #n here should be an integer
    if n == 0:
        return 1
    else:
        return n*factorial(n-1)
```

◦

Python◦ RuntimeError

```
def cursing(depth):
  try:
    cursing(depth + 1) # actually, re-cursing
  except RuntimeError as RE:
    print('I recursed {} times!'.format(depth))

cursing(0)
# Out: I recursed 1083 times!
```

`sys.setrecursionlimit(limit)` `sys.getrecursionlimit()`

```
def cursing(depth):
  try:
    cursing(depth + 1) # actually, re-cursing
  except RuntimeError as RE:
    print('I recursed {} times!'.format(depth))

cursing(0)
```

```
# Out: I recursed 1083 times!
```

Python 3.5`RecursionError` `RuntimeError`。

python。

```
def fibonacci(n):
    def step(a,b):
        return b, a+b
    a, b = 0, 1
    for i in range(n):
        a, b = step(a, b)
    return a
```

```
def fibonacci(n):
    def step(a,b):
        return b, a+b
    a, b = 0, 1
    for i in range(n):
        a, b = step(a, b)
    return a
```

argskwargs。 。

```
def unpacking(a, b, c=45, d=60, *args, **kwargs):
    print(a, b, c, d, args, kwargs)

>>> unpacking(1, 2)
1 2 45 60 () {}
>>> unpacking(1, 2, 3, 4)
1 2 3 4 () {}
>>> unpacking(1, 2, c=3, d=4)
1 2 3 4 () {}
>>> unpacking(1, 2, d=4, c=3)
1 2 3 4 () {}


>>> pair = (3,)
>>> unpacking(1, 2, *pair, d=4)
1 2 3 4 () {}
>>> unpacking(1, 2, d=4, *pair)
1 2 3 4 () {}
>>> unpacking(1, 2, *pair, c=3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'c'
>>> unpacking(1, 2, c=3, *pair)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'c'

>>> args_list = [3]
>>> unpacking(1, 2, *args_list, d=4)
1 2 3 4 () {}
>>> unpacking(1, 2, d=4, *args_list)
1 2 3 4 () {}
>>> unpacking(1, 2, c=3, *args_list)
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'c'
>>> unpacking(1, 2, *args_list, c=3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'c'


>>> pair = (3, 4)
>>> unpacking(1, 2, *pair)
1 2 3 4 () {}
>>> unpacking(1, 2, 3, 4, *pair)
1 2 3 4 (3, 4) {}
>>> unpacking(1, 2, d=4, *pair)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'd'
>>> unpacking(1, 2, *pair, d=4)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'd'



>>> args_list = [3, 4]
>>> unpacking(1, 2, *args_list)
1 2 3 4 () {}
>>> unpacking(1, 2, 3, 4, *args_list)
1 2 3 4 (3, 4) {}
>>> unpacking(1, 2, d=4, *args_list)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'd'
>>> unpacking(1, 2, *args_list, d=4)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'd'


>>> arg_dict = {'c':3, 'd':4}
>>> unpacking(1, 2, **arg_dict)
1 2 3 4 () {}
>>> arg_dict = {'d':4, 'c':3}
>>> unpacking(1, 2, **arg_dict)
1 2 3 4 () {}
>>> arg_dict = {'c':3, 'd':4, 'not_a_parameter': 75}
>>> unpacking(1, 2, **arg_dict)
1 2 3 4 () {'not_a_parameter': 75}


>>> unpacking(1, 2, *pair, **arg_dict)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'd'
>>> unpacking(1, 2, 3, 4, **arg_dict)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'd'

# Positional arguments take priority over any other form of argument passing
```

```
>>> unpacking(1, 2, **arg_dict, c=3)
1 2 3 4 () {'not_a_parameter': 75}
>>> unpacking(1, 2, 3, **arg_dict, c=3)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unpacking() got multiple values for argument 'c'
```

。

```
def f(*a, b):
    pass

f(1, 2, 3)
# TypeError: f() missing 1 required keyword-only argument: 'b'
```

Python 3。

```
def f(*a, b):
    pass

f(1, 2, 3)
# TypeError: f() missing 1 required keyword-only argument: 'b'
```

Lambda

lambda。 -

```
lambda_factorial = lambda i:1 if i==0 else i*lambda_factorial(i-1)
print(lambda_factorial(4)) # 4 * 3 * 2 * 1 = 12 * 2 = 24
```

lambda410 `i` *lambda-1 `i-1` 。 0 `return 1` 。

https://riptutorial.com/zh-TW/python/topic/228/

91:

Examples

`help`PythonPython。

```
>>> help()

Welcome to Python 3.4's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at http://docs.python.org/3.4/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".
```

_ 。

```
>>> 2 + 2
4
>>> _
4
>>> _ + 6
10
```

python。 for。 _。

```
>>> 2 + 2
4
>>> _
4
>>> _ + 6
10
```

python。 。

Python

Python`py`Windows`python`。

```
$ py
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

`python2python3` 。

PATHPython。

PYTHONSTARTUP

PythonPYTHONSTARTUP。 Python。

PYTHONSTARTUP

```
print("Welcome!")
```

Python

```
print("Welcome!")
```

Python`py` 。 `py --help`Python 3.4

```
Python Launcher

usage: py [ launcher-arguments ] [ python-arguments ] script [ script-arguments ]

Launcher arguments:

-2     : Launch the latest Python 2.x version
-3     : Launch the latest Python 3.x version
-X.Y   : Launch the specified Python version
-X.Y-32: Launch the specified 32bit Python version

The following help text is from Python:

usage: G:\Python34\python.exe [option] ... [-c cmd | -m mod | file | -] [arg] ...
Options and arguments (and corresponding environment variables):
-b     : issue warnings about str(bytes_instance), str(bytearray_instance)
         and comparing bytes/bytearray with str. (-bb: issue errors)
-B     : don't write .py[co] files on import; also PYTHONDONTWRITEBYTECODE=x
-c cmd : program passed in as string (terminates option list)
-d     : debug output from parser; also PYTHONDEBUG=x
-E     : ignore PYTHON* environment variables (such as PYTHONPATH)
-h     : print this help message and exit (also --help)
-i     : inspect interactively after running script; forces a prompt even
         if stdin does not appear to be a terminal; also PYTHONINSPECT=x
-I     : isolate Python from the user's environment (implies -E and -s)
-m mod : run library module as a script (terminates option list)
-O     : optimize generated bytecode slightly; also PYTHONOPTIMIZE=x
-OO    : remove doc-strings in addition to the -O optimizations
-q     : don't print version and copyright messages on interactive startup
-s     : don't add user site directory to sys.path; also PYTHONNOUSERSITE
-S     : don't imply 'import site' on initialization
-u     : unbuffered binary stdout and stderr, stdin always buffered;
         also PYTHONUNBUFFERED=x
         see man page for details on internal buffering relating to '-u'
-v     : verbose (trace import statements); also PYTHONVERBOSE=x
         can be supplied multiple times to increase verbosity
-V     : print the Python version number and exit (also --version)
-W arg : warning control; arg is action:message:category:module:lineno
```

```
         also PYTHONWARNINGS=arg
-x     : skip first line of source, allowing use of non-Unix forms of #!cmd
-X opt : set implementation-specific option
file   : program read from script file
-      : program read from stdin (default; interactive mode if a tty)
arg ...: arguments passed to program in sys.argv[1:]

Other environment variables:
PYTHONSTARTUP: file executed on interactive startup (no default)
PYTHONPATH   : ';'-separated list of directories prefixed to the
               default module search path.  The result is sys.path.
PYTHONHOME   : alternate <prefix> directory (or <prefix>;<exec_prefix>).
               The default module search path uses <prefix>\lib.
PYTHONCASEOK : ignore case in 'import' statements (Windows).
PYTHONIOENCODING: Encoding[:errors] used for stdin/stdout/stderr.
PYTHONFAULTHANDLER: dump the Python traceback on fatal errors.
PYTHONHASHSEED: if this variable is set to 'random', a random value is used
   to seed the hashes of str, bytes and datetime objects.  It can also be
   set to an integer in the range [0,4294967295] to get hash values with a
   predictable seed.
```

Python`help`。

`help`docstring。

```
>>> help(print)
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```

`help`docstring。

```
>>> help(print)
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```

https://riptutorial.com/zh-TW/python/topic/2473/--

92:

- abc
- ab
- x =x
- xy=xy
- x + =x。 “op =”
- del xx
- for x in something#binds x
- xx
- Ex Exceptionex

Examples

Python。 。 。

```
x = 'Hi'

def read_x():
    print(x)    # x is just referenced, therefore assumed global

read_x()        # prints Hi

def read_y():
    print(y)    # here y is just referenced, therefore assumed global

read_y()        # NameError: global name 'y' is not defined

def read_y():
    y = 'Hey'   # y appears in an assignment, therefore it's local
    print(y)    # will find the local y

read_y()        # prints Hey

def read_x_local_fail():
    if False:
        x = 'Hey'  # x appears in an assignment, therefore it's local
    print(x)    # will look for the _local_ z, which is not assigned, and will not be found

read_x_local_fail()   # UnboundLocalError: local variable 'x' referenced before assignment
```

```
x = 'Hi'

def read_x():
    print(x)    # x is just referenced, therefore assumed global

read_x()        # prints Hi

def read_y():
    print(y)    # here y is just referenced, therefore assumed global

read_y()        # NameError: global name 'y' is not defined
```

```
def read_y():
    y = 'Hey'  # y appears in an assignment, therefore it's local
    print(y)   # will find the local y

read_y()       # prints Hey

def read_x_local_fail():
    if False:
        x = 'Hey'  # x appears in an assignment, therefore it's local
    print(x)   # will look for the _local_ z, which is not assigned, and will not be found

read_x_local_fail()   # UnboundLocalError: local variable 'x' referenced before assignment
```

global

```
x = 'Hi'

def read_x():
    print(x)   # x is just referenced, therefore assumed global

read_x()       # prints Hi

def read_y():
    print(y)   # here y is just referenced, therefore assumed global

read_y()       # NameError: global name 'y' is not defined

def read_y():
    y = 'Hey'  # y appears in an assignment, therefore it's local
    print(y)   # will find the local y

read_y()       # prints Hey

def read_x_local_fail():
    if False:
        x = 'Hey'  # x appears in an assignment, therefore it's local
    print(x)   # will look for the _local_ z, which is not assigned, and will not be found

read_x_local_fail()   # UnboundLocalError: local variable 'x' referenced before assignment
```

global◦ ◦

x

1. global x x
2. nonlocal x x
3. x = 5for x in range(3)x
4. x

```
def foo():
    a = 5
    print(a) # ok

print(a) #  NameError: name 'a' is not defined
```

except

```
def foo():
    a = 5
    print(a) # ok

print(a) #  NameError: name 'a' is not defined
```

fora += 5

Python 3.x 3.0

Python 3**nonlocal**。 nonlocal。 PEP 3104。 。

```
def counter():
    num = 0
    def incrementer():
        num += 1
        return num
    return incrementer
```

UnboundLocalErrornum。

```
def counter():
    num = 0
    def incrementer():
        num += 1
        return num
    return incrementer
```

nonlocal。 counternonlocal 。 SyntaxError 。 nonlocal 。

。

```
x = 5
x += 7
for x in iterable: pass
```

- x5。 x。 “”。

。 lambdas。

```
a = 'global'

class Fred:
    a = 'class'  # class scope
    b = (a for i in range(10))  # function scope
    c = [a for i in range(10)]  # function scope
    d = a  # class scope
    e = lambda: a  # function scope
    f = lambda a=a: a  # default argument uses class scope

    @staticmethod  # or @classmethod, or regular instance method
    def g():  # function scope
        return a
```

```
print(Fred.a)  # class
print(next(Fred.b))  # global
print(Fred.c[0])  # class in Python 2, global in Python 3
print(Fred.d)  # class
print(Fred.e())  # global
print(Fred.f())  # class
print(Fred.g()) # global
```

b ceclass。

PEP 227

。 。 。

Python

;-。

```
a = 'global'

class Fred:
    a = 'class'  # class scope
    b = (a for i in range(10))  # function scope
    c = [a for i in range(10)]  # function scope
    d = a  # class scope
    e = lambda: a  # function scope
    f = lambda a=a: a  # default argument uses class scope

    @staticmethod  # or @classmethod, or regular instance method
    def g():  # function scope
        return a

print(Fred.a)  # class
print(next(Fred.b))  # global
print(Fred.c[0])  # class in Python 2, global in Python 3
print(Fred.d)  # class
print(Fred.e())  # global
print(Fred.f())  # class
print(Fred.g()) # global
```

Martijn Pieters。

del

。

del v

vdel v。

```
x = 5
print(x) # out: 5
```

```
del x
print(x) # NameError: name 'f' is not defined
```

del nonlocalglobal del vv。 v del vnonlocal vglobal v 。

。 。

del v.name

v.__delattr__(name)。

name。

```
x = 5
print(x) # out: 5
del x
print(x) # NameError: name 'f' is not defined
```

del v[item]

v.__delitem__(item)。

itemv 。

```
x = 5
print(x) # out: 5
del x
print(x) # NameError: name 'f' is not defined
```

del v[a:b]

v.__delslice__(a, b) 。

- 。

```
x = 5
print(x) # out: 5
del x
print(x) # NameError: name 'f' is not defined
```

#del 。

Python 。

。

```
foo = 1  # global
```

```
def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

◦ locals()globals()◦

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

global

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

◦

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

◦ ◦

```
foo = 1  # global

def func():
    bar = 2  # local
```

```
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

`global` vs `nonlocal` Python 3

。

`global`。

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

Python 3`nonlocal` 。

`nonlocal`Python

。

Python 3.x 3.0

```
foo = 1  # global

def func():
    bar = 2  # local
    print(foo)  # prints variable foo from global scope
    print(bar)  # prints variable bar from local scope
```

https://riptutorial.com/zh-TW/python/topic/263/

93:

Python **List**Python。 。 。 。

- [.....]
- []

`list`iterablePython。 `set` `tuple``dictionary`

`list`PythonC ++`vector<void*>`Java`ArrayList<Object>` 。 。

。 `list` 。

。

Examples

Python。

```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```

`IndexError` 。

```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```

。

```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```

```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```

`lst[start:end:step]` 。 `start``end`-1。 `start` `end` `step` 1

```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```

```
lst = [1, 2, 3, 4]
lst[0]  # 1
```

```
lst[1]  # 2
```

。

```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```


```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```


1。

slice__getitem__()。 Python。


```
lst = [1, 2, 3, 4]
lst[0]  # 1
lst[1]  # 2
```


__getitem__ 。

a

```
a = [1, 2, 3, 4, 5]
```

1. append(value) - 。

   ```
   a = [1, 2, 3, 4, 5]
   ```

   append()。 。

   ```
   a = [1, 2, 3, 4, 5]
   ```

2. extend(enumerable) - 。

   ```
   a = [1, 2, 3, 4, 5]
   ```

   +。

   ```
   a = [1, 2, 3, 4, 5]
   ```

3. index(value, [startIndex]) - 。 ValueError。 。

   ```
   a = [1, 2, 3, 4, 5]
   ```

4. insert(index, value) - indexvalue 。 index 。

```
a = [1, 2, 3, 4, 5]
```

5. `pop([index])` - `index`。 。

```
a = [1, 2, 3, 4, 5]
```

6. `remove(value)` - 。 `ValueError` 。

```
a = [1, 2, 3, 4, 5]
```

7. `reverse()` - `None` 。

```
a = [1, 2, 3, 4, 5]
```

。

8. `count(value)` - 。

```
a = [1, 2, 3, 4, 5]
```

9. `sort()` - `None` 。

```
a = [1, 2, 3, 4, 5]
```

`sort()` `reverse=True`。

```
a = [1, 2, 3, 4, 5]
```

`key`

```
a = [1, 2, 3, 4, 5]
```

dicts

```
a = [1, 2, 3, 4, 5]
```

```
a = [1, 2, 3, 4, 5]
```

attrgetteritemgetter

`attrgetteritemgetter`。 。

```
a = [1, 2, 3, 4, 5]
```

`itemgetter`。 。

```
a = [1, 2, 3, 4, 5]
```

`attrgetter`

```
a = [1, 2, 3, 4, 5]
```

10. `clear()` -

    ```
    a = [1, 2, 3, 4, 5]
    ```

11. - 。

    ```
    a = [1, 2, 3, 4, 5]
    ```

    - 。

12. - `del`

    ```
    a = [1, 2, 3, 4, 5]
    ```

13. “=”。 。 。 。

    ```
    a = [1, 2, 3, 4, 5]
    ```

    。

    ```
    a = [1, 2, 3, 4, 5]
    ```

    list

    ```
    a = [1, 2, 3, 4, 5]
    ```

    copy.copy

    ```
    a = [1, 2, 3, 4, 5]
    ```

    listold_list。

    copy.deepcopy

    ```
    a = [1, 2, 3, 4, 5]
    ```

    。

Python 3.x 3.0

`copy()` -

```
a = [1, 2, 3, 4, 5]
```

len()。

```
len(['one', 'two'])  # returns 2

len(['one', [2, 3], 'four'])  # returns 3, not 4
```

len()。

len()。

len()O(1) 。

Pythonfor

```
my_list = ['foo', 'bar', 'baz']
for item in my_list:
    print(item)

# Output: foo
# Output: bar
# Output: baz
```

```
my_list = ['foo', 'bar', 'baz']
for item in my_list:
    print(item)

# Output: foo
# Output: bar
# Output: baz
```

```
my_list = ['foo', 'bar', 'baz']
for item in my_list:
    print(item)

# Output: foo
# Output: bar
# Output: baz
```

```
my_list = ['foo', 'bar', 'baz']
for item in my_list:
    print(item)

# Output: foo
# Output: bar
# Output: baz
```

bar。

Python。 in。

```
lst = ['test', 'twest', 'tweast', 'treast']

'test' in lst
# Out: True

'toast' in lst
# Out: False
```

in。 listset set 。


```
lst = ['test', 'twest', 'tweast', 'treast']

'test' in lst
# Out: True

'toast' in lst
# Out: False
```


reversed

```
In [3]: rev = reversed(numbers)

In [4]: rev
Out[4]: [9, 8, 7, 6, 5, 4, 3, 2, 1]
```

“”。

reverse 。

-1


```
In [3]: rev = reversed(numbers)

In [4]: rev
Out[4]: [9, 8, 7, 6, 5, 4, 3, 2, 1]
```


Falselen(lst) == 0 lstnot lst

```
lst = []
if not lst:
    print("list is empty")

# Output: list is empty
```

1. list1list2

   ```
   merged = list1 + list2
   ```

2. **zip** ii

   ```
   merged = list1 + list2
   ```

```
merged = list1 + list2
```

`Noneitertools.zip_longest` Python 2`itertools.izip_longest`

```
merged = list1 + list2
```

3. ```
   merged = list1 + list2
   ```

```
merged = list1 + list2
```

`all()`iterableTrue

```
nums = [1, 1, 0, 1]
all(nums)
# False
chars = ['a', 'b', 'c', 'd']
all(chars)
# True
```

`any()`iterableTrue

```
nums = [1, 1, 0, 1]
all(nums)
# False
chars = ['a', 'b', 'c', 'd']
all(chars)
# True
```

。

```
nums = [1, 1, 0, 1]
all(nums)
# False
chars = ['a', 'b', 'c', 'd']
all(chars)
# True
```

`set` 。 `listlist()list()`

```
names = ["aixk", "duke", "edik", "tofp", "duke"]
list(set(names))
# Out: ['duke', 'tofp', 'aixk', 'edik']
```

。

`OrderedDict`

```
names = ["aixk", "duke", "edik", "tofp", "duke"]
list(set(names))
# Out: ['duke', 'tofp', 'aixk', 'edik']
```
```

```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```


```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```

```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```


for


```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```

```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```

```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```


for。


```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```

```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```

```
alist = [[[1,2],[3,4]], [[5,6,7],[8,9,10], [12, 13, 14]]]
```


。 。

```
[1, 10, 100] < [2, 10, 100]
# True, because 1 < 2
[1, 10, 100] < [1, 10, 100]
# False, because the lists are equal
[1, 10, 100] <= [1, 10, 100]
# True, because the lists are equal
[1, 10, 100] < [1, 10, 101]
# True, because 100 < 101
[1, 10, 100] < [0, 10, 100]
# False, because 0 < 1
```

。


```
[1, 10, 100] < [2, 10, 100]
# True, because 1 < 2
[1, 10, 100] < [1, 10, 100]
# False, because the lists are equal
[1, 10, 100] <= [1, 10, 100]
# True, because the lists are equal
[1, 10, 100] < [1, 10, 101]
# True, because 100 < 101
[1, 10, 100] < [0, 10, 100]
# False, because 0 < 1
```

```
None
```

```
my_list = [None] * 10
my_list = ['test'] * 10
```


```
my_list = [None] * 10
my_list = ['test'] * 10
```

```
my_list = [None] * 10
my_list = ['test'] * 10
```


https://riptutorial.com/zh-TW/python/topic/209/

94:

Python。 `unittest`。 `py.testnosetests`。 python。

Examples

。 。

```
class WrongInputException(Exception):
    pass
```

。

```
class WrongInputException(Exception):
    pass
```

`assertRaises`。 `assertRaises`

1. 。 。
2. `with`。 。 with self.assertRaisesWrongInputExceptionconvert2number“not a number”

```
class WrongInputException(Exception):
    pass
```

。 。 。 `fail`。

unittest.mock.create_autospec

`create_autospec`。 。

`custom_math.pymultiply`

```
def multiply(a, b):
    return a * b
```

`process_math.pymultiples_of`

```
def multiply(a, b):
    return a * b
```

`multiplymultiples_of`。 Pythonunittestpytestnose

```
def multiply(a, b):
    return a * b
```

test.TestCase

◦ setUp◦ tearDown◦ ◦ TestCasesuper setUptearDown◦ TestCasesetUptearDown

```
import unittest


class SomeTest(unittest.TestCase):
    def setUp(self):
        super(SomeTest, self).setUp()
        self.mock_data = [1,2,3,4,5]

    def test(self):
        self.assertEqual(len(self.mock_data), 5)

    def tearDown(self):
        super(SomeTest, self).tearDown()
        self.mock_data = []


if __name__ == '__main__':
    unittest.main()
```

python2.7 +addCleanup◦ setUptearDownsetUpaddCleanup◦

```
import unittest


class SomeTest(unittest.TestCase):
    def setUp(self):
        super(SomeTest, self).setUp()
        self.mock_data = [1,2,3,4,5]

    def test(self):
        self.assertEqual(len(self.mock_data), 5)

    def tearDown(self):
        super(SomeTest, self).tearDown()
        self.mock_data = []


if __name__ == '__main__':
    unittest.main()
```

tearDownsuper◦

unittest◦

```
def division_function(dividend, divisor):
    return dividend / divisor


class MyTestCase(unittest.TestCase):
    def test_using_context_manager(self):
        with self.assertRaises(ZeroDivisionError):
            x = division_function(1, 0)
```

◦ ◦

。

```
def division_function(dividend, divisor):
    return dividend / divisor


class MyTestCase(unittest.TestCase):
    def test_using_context_manager(self):
        with self.assertRaises(ZeroDivisionError):
            x = division_function(1, 0)
```

```
def division_function(dividend, divisor):
    return dividend / divisor


class MyTestCase(unittest.TestCase):
    def test_using_context_manager(self):
        with self.assertRaises(ZeroDivisionError):
            x = division_function(1, 0)
```

。 。

Unittests

Python`assert` Python。

`assertTrue`

```
import unittest

class SimplisticTest(unittest.TestCase):
    def test_basic(self):
        self.assertTrue(1 + 1 == 2)
```

```
import unittest

class SimplisticTest(unittest.TestCase):
    def test_basic(self):
        self.assertTrue(1 + 1 == 2)
```

。

`assertTrue`。

```
import unittest

class SimplisticTest(unittest.TestCase):
    def test_basic(self):
        self.assertTrue(1 + 1 == 2)
```

```
import unittest

class SimplisticTest(unittest.TestCase):
```

```
    def test_basic(self):
        self.assertTrue(1 + 1 == 2)
```

```
import unittest

class SimplisticTest(unittest.TestCase):
    def test_basic(self):
        self.assertTrue(1 + 1 == 2)
```

◦

◦ ◦ `1 + 1 == 2` `assertEqualassertTrue` ◦ `a is None` `assertIsNoneassertEqual` ◦

◦ `assertEqualassertNotEqual` `assertIsNoneassertIsNotNone` ◦ ◦

pytest

pytest

```
pip install pytest
```

```
pip install pytest
```

`docker_something/helpers.py`

```
pip install pytest
```

`test_docker.py`

```
pip install pytest
```

`test_docker.pytest_docker.py`

```
pip install pytest
```

`test_docker.py`pytest

```
pip install pytest
```

`test_docker.pytest_` `test_docker.py`

```
pip install pytest
```

```
pip install pytest
```

`teststests`

```
pip install pytest
```

https://riptutorial.com/zh-TW/python/topic/631/

95: GIL

GIL

1992PythonGILCPython。 python。 GILPythonPython。 GIL。

GIL*python*。

。

GILPython。 pythonGIL。 PythonGIL。

GIL CPythonPyPyCython`nogil`GIL

GIL JythonIronPython

GIL

GIL。 GIL。 CPython3.2python。 。 3.2GIL。 。 dabeaz.com。

I / OCPythonGIL。 numpyGIL。

GIL

GILGIL。 。

- - GIL。 *CPythongarbargeGIL*。 ;GILpython.org wikiGIL。
- GIL -
-

GIL

GILpythonpython。 python。 GIL。

---

https://wiki.python.org/moin/GlobalInterpreterLock -

http://programmers.stackexchange.com/questions/186889/why-was-python-written-with-the-gil -

http://www.dabeaz.com/python/UnderstandingGIL.pdf - GIL

http://www.dabeaz.com/GIL/gilvis/index.html - GIL

http://jeffknupp.com/blog/2012/03/31/pythons-hardest-problem/ - GIL

https://jeffknupp.com/blog/2013/06/30/pythons-hardest-problem-revisited/ - GIL

Examples

Multiprocessing.Pool

Python“。 。 ”Pool。

David BeazleyGIL multiprocessing.Pool

David BeazleyGIL

```
from threading import Thread
import time
def countdown(n):
    while n > 0:
        n -= 1

COUNT = 10000000

t1 = Thread(target=countdown,args=(COUNT/2,))
t2 = Thread(target=countdown,args=(COUNT/2,))
start = time.time()
t1.start();t2.start()
t1.join();t2.join()
end = time.time()
print end-start
```

multiprocessing.Pool

```
from threading import Thread
import time
def countdown(n):
    while n > 0:
        n -= 1

COUNT = 10000000

t1 = Thread(target=countdown,args=(COUNT/2,))
t2 = Thread(target=countdown,args=(COUNT/2,))
start = time.time()
t1.start();t2.start()
t1.join();t2.join()
end = time.time()
print end-start
```

。 GIL。 multiprocessing.Pool。 。 。 。

`with`。 。

Cython nogil

Cythonpython。 GIL。

David BeazleyGIL nogil

David BeazleyGIL

```
from threading import Thread
import time
def countdown(n):
    while n > 0:
        n -= 1

COUNT = 10000000

t1 = Thread(target=countdown,args=(COUNT/2,))
t2 = Thread(target=countdown,args=(COUNT/2,))
start = time.time()
t1.start();t2.start()
t1.join();t2.join()
end = time.time()
print end-start
```

nogilCYTHON

```
from threading import Thread
import time
def countdown(n):
    while n > 0:
        n -= 1

COUNT = 10000000

t1 = Thread(target=countdown,args=(COUNT/2,))
t2 = Thread(target=countdown,args=(COUNT/2,))
start = time.time()
t1.start();t2.start()
t1.join();t2.join()
end = time.time()
print end-start
```

cython。 python

PythonGILPython。 Cython。

GIL https://riptutorial.com/zh-TW/python/topic/4061/-gil-

96:

python

Examples

PyDotPlus

PyDotPluspydotGraphvizDotPython。

```
pip install pydotplus
```


```
pip install pydotplus
```


DOT

- DOT。 Dot。 demo.dot

  digraph demo1 {a - > b - > c; c - > a; }

```
pip install pydotplus
```

svg

PyGraphviz

http://pypi.python.org/pypi/pygraphvizPythonPyGraphviz

```
pip install pygraphviz
```

Python。

github.com

```
pip install git://github.com/pygraphviz/pygraphviz.git#egg=pygraphviz
```

http://pypi.python.org/pypi/pygraphvizPythonPyGraphviz

```
easy_install pygraphviz
```

Python。

DOT

- DOT。 Dot。 demo.dot

  digraph demo1 {a - > b - > c; c - > a; }

- 。

```
import pygraphviz as pgv
G = pgv.AGraph("demo.dot")
G.draw('test', format='svg', prog='dot')
```

svg

https://riptutorial.com/zh-TW/python/topic/9483/

97: windowsvirtualenvwrapper

Examples

windowsvirtualenvwrapper

ABC.ABpython 3。 Cpython 2.7。

- python

1pip `python -m pip install -U pip`

2“virtualenvwrapper-win”windows power shell

`pip install virtualenvwrapper-win`

3virtualenv `mkvirtualenv python_3.5`

4

```
workon < environment name>
```

virtualenvwrapper

```
workon < environment name>
```

windowsvirtualenvwrapper https://riptutorial.com/zh-TW/python/topic/9984/windows virtualenvwrapper

98:

Python“return”。

Examples

var1

```
def func(params):
    for value in params:
        print ('Got value {}'.format(value))

        if value == 1:
            # Returns from function as soon as value is 1
            print (">>>> Got 1")
            return

        print ("Still looping")

    return "Couldn't find 1"

func([5, 3, 1, 2, 8, 9])
```

```
def func(params):
    for value in params:
        print ('Got value {}'.format(value))

        if value == 1:
            # Returns from function as soon as value is 1
            print (">>>> Got 1")
            return

        print ("Still looping")

    return "Couldn't find 1"

func([5, 3, 1, 2, 8, 9])
```

https://riptutorial.com/zh-TW/python/topic/10883/

99:

- mapfunctioniterable [* additional_iterables]
- future_builtins.mapfunctioniterable [* additional_iterables]
- itertools.imapfunctioniterable [* additional_iterables]

	iterable
* additional_iterables	

`map`comprehensions

```
list(map(abs, [-1,-2,-3]))     # [1, 2, 3]
[abs(i) for i in [-1,-2,-3]]   # [1, 2, 3]
```

iterables`zip`

```
list(map(abs, [-1,-2,-3]))     # [1, 2, 3]
[abs(i) for i in [-1,-2,-3]]   # [1, 2, 3]
```

`map`。

Examples

mapitertools.imapfuture_builtins.map

mapPython。 `map()`iterable

```
names = ['Fred', 'Wilma', 'Barney']
```

Python 3.x 3.0

```
names = ['Fred', 'Wilma', 'Barney']
```

`future_builtins`Python 3`map`

Python 2.x 2.6

```
names = ['Fred', 'Wilma', 'Barney']
```

Python 2`itertools imap`

Python 2.x 2.3

```
names = ['Fred', 'Wilma', 'Barney']
```

`list`Python 23

```
names = ['Fred', 'Wilma', 'Barney']
```

`map()`

```
names = ['Fred', 'Wilma', 'Barney']
```

```
list(map(abs, (1, -1, 2, -2, 3, -3))) # the call to `list` is unnecessary in 2.x
# Out: [1, 1, 2, 2, 3, 3]
```


```
list(map(abs, (1, -1, 2, -2, 3, -3))) # the call to `list` is unnecessary in 2.x
# Out: [1, 1, 2, 2, 3, 3]
```

```
list(map(abs, (1, -1, 2, -2, 3, -3))) # the call to `list` is unnecessary in 2.x
# Out: [1, 1, 2, 2, 3, 3]
```

```
list(map(abs, (1, -1, 2, -2, 3, -3))) # the call to `list` is unnecessary in 2.x
# Out: [1, 1, 2, 2, 3, 3]
```


`functools.partialmaplambda`。

`i`

```
def average(*args):
    return float(sum(args)) / len(args)  # cast to float - only mandatory for python 2.x

measurement1 = [100, 111, 99, 97]
measurement2 = [102, 117, 91, 102]
measurement3 = [104, 102, 95, 101]

list(map(average, measurement1, measurement2, measurement3))
# Out: [102.0, 110.0, 95.0, 100.0]
```

pythoniterable`map`

-
```
def average(*args):
    return float(sum(args)) / len(args)  # cast to float - only mandatory for python 2.x

measurement1 = [100, 111, 99, 97]
measurement2 = [102, 117, 91, 102]
measurement3 = [104, 102, 95, 101]

list(map(average, measurement1, measurement2, measurement3))
# Out: [102.0, 110.0, 95.0, 100.0]
```


TypeErrormedian_of_three1'c'

```
def average(*args):
    return float(sum(args)) / len(args)  # cast to float - only mandatory for python 2.x

measurement1 = [100, 111, 99, 97]
measurement2 = [102, 117, 91, 102]
measurement3 = [104, 102, 95, 101]

list(map(average, measurement1, measurement2, measurement3))
# Out: [102.0, 110.0, 95.0, 100.0]
```

TypeErrormedian_of_three34

Python 2.x 2.0.1

- `map` iterableiterables`None`

```
def average(*args):
    return float(sum(args)) / len(args)  # cast to float - only mandatory for python 2.x

measurement1 = [100, 111, 99, 97]
measurement2 = [102, 117, 91, 102]
measurement3 = [104, 102, 95, 101]

list(map(average, measurement1, measurement2, measurement3))
# Out: [102.0, 110.0, 95.0, 100.0]
```

TypeError - 'int''NoneType'

- `itertools.imapfuture_builtins.map` iterable

```
def average(*args):
    return float(sum(args)) / len(args)  # cast to float - only mandatory for python 2.x

measurement1 = [100, 111, 99, 97]
measurement2 = [102, 117, 91, 102]
measurement3 = [104, 102, 95, 101]

list(map(average, measurement1, measurement2, measurement3))
# Out: [102.0, 110.0, 95.0, 100.0]
```

Python 3.x 3.0.0

- ```
  def average(*args):
      return float(sum(args)) / len(args)  # cast to float - only mandatory for python 2.x

  measurement1 = [100, 111, 99, 97]
  measurement2 = [102, 117, 91, 102]
  measurement3 = [104, 102, 95, 101]

  list(map(average, measurement1, measurement2, measurement3))
  # Out: [102.0, 110.0, 95.0, 100.0]
  ```

Map“None”python 2.x
  ```

```
from itertools import imap
from future_builtins import map as fmap # Different name to highlight differences

image = [[1, 2, 3],
         [4, 5, 6],
         [7, 8, 9]]

list(map(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
list(fmap(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
list(imap(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]

image2 = [[1, 2, 3],
          [4, 5],
          [7, 8, 9]]
list(map(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8), (3, None, 9)]  # Fill missing values with None
list(fmap(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8)]                # ignore columns with missing values
list(imap(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8)]                # dito
```

Python 3.x 3.0.0

```
from itertools import imap
from future_builtins import map as fmap # Different name to highlight differences

image = [[1, 2, 3],
         [4, 5, 6],
         [7, 8, 9]]

list(map(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
list(fmap(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
list(imap(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]

image2 = [[1, 2, 3],
          [4, 5],
          [7, 8, 9]]
list(map(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8), (3, None, 9)]  # Fill missing values with None
list(fmap(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8)]                # ignore columns with missing values
list(imap(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8)]                # dito
```

TypeError'NoneType'

```
from itertools import imap
from future_builtins import map as fmap # Different name to highlight differences

image = [[1, 2, 3],
         [4, 5, 6],
         [7, 8, 9]]
```

```
list(map(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
list(fmap(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]
list(imap(None, *image))
# Out: [(1, 4, 7), (2, 5, 8), (3, 6, 9)]

image2 = [[1, 2, 3],
          [4, 5],
          [7, 8, 9]]
list(map(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8), (3, None, 9)]  # Fill missing values with None
list(fmap(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8)]                # ignore columns with missing values
list(imap(None, *image2))
# Out: [(1, 4, 7), (2, 5, 8)]                # dito
```

map'import'。print5importprintimport pprint。pprint

iterable。。

1

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```


```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```


2


```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```



```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```


。。


```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```


3

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```

4

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```

5

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```

```
insects = ['fly', 'ant', 'beetle', 'cankerworm']
f = lambda x: x + ' is an insect'
print(list(map(f, insects))) # the function defined by f is executed on each item of the
iterable insects
```

https://riptutorial.com/zh-TW/python/topic/333/

100:

Python3.5。 `a = myobject` myobject。 `0`

Python`del`。 。 `del`refcountrefcount 。

Pythona = objectcpython。 refcount0GC。

1960John McCarthyLisp。 Python**Generational Garbage Collection**。

Python。 Python。 0“1”0。 。 Python30;GC1GC2 。 “2”GCedGCGC。 。 “”。 “”60。

。 GC25。 GC。 “”;GCGCMartinvonLöwis2008。 “”“”。

GC。

```
/* Break reference cycles by clearing the containers involved.  This is
 * tricky business as the lists can be changing and we don't know which
 * objects may be freed.  It is possible I screwed something up here.
 */
static void
delete_garbage(PyGC_Head *collectable, PyGC_Head *old)
```

;。 GC0.。 Python`__del__`Python`__del__` 。 `__del__`。`__del__`。

```
/* Break reference cycles by clearing the containers involved.  This is
 * tricky business as the lists can be changing and we don't know which
 * objects may be freed.  It is possible I screwed something up here.
 */
static void
delete_garbage(PyGC_Head *collectable, PyGC_Head *old)
```

ABA;A`__del__`A。 B`__del__` A。 `__del__`。`__del__`GCedGC。 `__del__`; PythonGCing`__del__`。

CPythonGC`__del__`

```
/* Break reference cycles by clearing the containers involved.  This is
 * tricky business as the lists can be changing and we don't know which
 * objects may be freed.  It is possible I screwed something up here.
 */
static void
delete_garbage(PyGC_Head *collectable, PyGC_Head *old)
```

Examples

Python。

。 。

◦ ◦

```
>>> import gc; gc.disable()  # disable garbage collector
>>> class Track:
        def __init__(self):
            print("Initialized")
        def __del__(self):
            print("Destructed")
>>> def foo():
        Track()
        # destructed immediately since no longer has any references
        print("---")
        t = Track()
        # variable is referenced, so it's not destructed yet
        print("---")
        # variable is destructed when function exits
>>> foo()
Initialized
Destructed
---
Initialized
---
Destructed
```

```
>>> import gc; gc.disable()  # disable garbage collector
>>> class Track:
        def __init__(self):
            print("Initialized")
        def __del__(self):
            print("Destructed")
>>> def foo():
        Track()
        # destructed immediately since no longer has any references
        print("---")
        t = Track()
        # variable is referenced, so it's not destructed yet
        print("---")
        # variable is destructed when function exits
>>> foo()
Initialized
Destructed
---
Initialized
---
Destructed
```

◦ ABBAAB.AB1◦

```
>>> import gc; gc.disable()  # disable garbage collector
>>> class Track:
        def __init__(self):
            print("Initialized")
        def __del__(self):
            print("Destructed")
>>> A = Track()
Initialized
>>> B = Track()
Initialized
```

```
>>> A.other = B
>>> B.other = A
>>> del A; del B  # objects are not destructed due to reference cycle
>>> gc.collect()  # trigger collection
Destructed
Destructed
4
```

- ABC.....AZAZ

```
>>> import gc; gc.disable()  # disable garbage collector
>>> class Track:
        def __init__(self):
            print("Initialized")
        def __del__(self):
            print("Destructed")
>>> A = Track()
Initialized
>>> B = Track()
Initialized
>>> A.other = B
>>> B.other = A
>>> del A; del B  # objects are not destructed due to reference cycle
>>> gc.collect()  # trigger collection
Destructed
Destructed
4
```

del

`del vdel v[item]del[i:j]del v[item]del v.name` ◦ ◦

```
>>> import gc
>>> gc.disable()  # disable garbage collector
>>> class Track:
        def __init__(self):
            print("Initialized")
        def __del__(self):
            print("Destructed")
>>> def bar():
    return Track()
>>> t = bar()
Initialized
>>> another_t = t  # assign another reference
>>> print("...")
...
>>> del t          # not destructed yet - another_t still refers to it
>>> del another_t  # final reference gone, object is destructed
Destructed
```

- ;-5256Python

```
>>> import sys
>>> sys.getrefcount(1)
797
>>> a = 1
```

```
>>> b = 1
>>> sys.getrefcount(1)
799
```

ab1。 Python

```
>>> import sys
>>> sys.getrefcount(1)
797
>>> a = 1
>>> b = 1
>>> sys.getrefcount(1)
799
```

999999999ab。

```
>>> import sys
>>> a = object()
>>> sys.getrefcount(a)
2
>>> b = a
>>> sys.getrefcount(a)
3
>>> del b
>>> sys.getrefcount(a)
2
```

Python 230。

ctypes。

Python。 。

Python 3.x 3.0

```
import ctypes
deallocated = 12345
ctypes.pythonapi._Py_Dealloc(ctypes.py_object(deallocated))
```

Python 2.x 2.3

```
import ctypes
deallocated = 12345
ctypes.pythonapi._Py_Dealloc(ctypes.py_object(deallocated))
```

Python - 。 ...

None - Fatal Python error: deallocating NoneFatal Python error: deallocating None 。

。 。

。 Python。 - **generation0** 。 - 。

```
import gc
gc.set_threshold(1000, 100, 10) # Values are just for demonstration purpose
```

generation0。 1000。

。 。 **generation0**100**generation1generation1**。 **generation1**10。

generation0_threshold。 。 。

```
import gc
gc.set_threshold(1000, 100, 10) # Values are just for demonstration purpose
```

。 。 。

。 。 。

。 Web。 Web。

。

。

f。

```
>>> f = open("test.txt")
>>> del f
```

f.close() 。 with

```
>>> f = open("test.txt")
>>> del f
```

with。 。 while。

https://riptutorial.com/zh-TW/python/topic/2532/

101:

Examples

inputraw_input

Python 2.x 2.3

`raw_input`。

```
foo = raw_input("Put a message here that asks the user for input")
```

`foo`。

Python 3.x 3.0

`input`。

```
foo = raw_input("Put a message here that asks the user for input")
```

`foo`。

Python 3.x 3.0

Python 3

```
print("This string will be displayed in the output")
# This string will be displayed in the output

print("You can print \n escape characters too.")
# You can print escape characters too.
```

Python 2.x 2.3

Python 2print。

```
print("This string will be displayed in the output")
# This string will be displayed in the output

print("You can print \n escape characters too.")
# You can print escape characters too.
```

Python 2`from __future__ import print_function`Python 3`print()`。 Python 2.6。

```
def input_number(msg, err_msg=None):
    while True:
        try:
            return float(raw_input(msg))
```

```
        except ValueError:
            if err_msg is not None:
                print(err_msg)


def input_number(msg, err_msg=None):
    while True:
        try:
            return float(input(msg))
        except ValueError:
            if err_msg is not None:
                print(err_msg)
```

```
def input_number(msg, err_msg=None):
    while True:
        try:
            return float(raw_input(msg))
        except ValueError:
            if err_msg is not None:
                print(err_msg)


def input_number(msg, err_msg=None):
    while True:
        try:
            return float(input(msg))
        except ValueError:
            if err_msg is not None:
                print(err_msg)
```

“”

```
def input_number(msg, err_msg=None):
    while True:
        try:
            return float(raw_input(msg))
        except ValueError:
            if err_msg is not None:
                print(err_msg)


def input_number(msg, err_msg=None):
    while True:
        try:
            return float(input(msg))
        except ValueError:
            if err_msg is not None:
                print(err_msg)
```

Python 2.x 2.3

Python 2.x`printprint`。 。

```
print "Hello,",
print "World!"
# Hello, World!
```

Python 3.x 3.0

Python 3.x `printend`。

```
print "Hello,",
print "World!"
# Hello, World!
```

```
print "Hello,",
print "World!"
# Hello, World!
```

`sys.stdout.write`

```
print "Hello,",
print "World!"
# Hello, World!
```

stdin

Pythonunix。 `stdin`

```
import sys

for line in sys.stdin:
    print(line)
```

`sys.stdin`。 for。

python

```
import sys

for line in sys.stdin:
    print(line)
```

`cat myfilestdout`unix。

fileinput

```
import sys

for line in sys.stdin:
    print(line)
```

。 open 。 `with <command> as <name>`“”open

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

。

◦ ◦ r ◦ rb ◦ a ◦ w◦ r+◦ open()◦ r ◦

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

◦ ◦

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

◦ for line in fileobjstrip()◦

fileobjfileobj ◦ 0 ◦ tell

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

◦ read()◦ read()◦ rbr

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

```
with open('somefile.txt', 'r') as fileobj:
    # write code here using fileobj
```

https://riptutorial.com/zh-TW/python/topic/266/

102:

LIFO。 。 - 。 。

- stack = []
- stack.appendobject
- stack.pop - > object
- list [-1] - > object

  push pop 。

LIFO 。

Python`append()`push `pop()`pop。 O1。

Python`deque`。 `deque` s。

Examples

ListStack

`list`。 python `listStack` 。

```
#define a stack class
class Stack:
    def __init__(self):
        self.items = []

    #method to check the stack is empty or not
    def isEmpty(self):
        return self.items == []

    #method for pushing an item
    def push(self, item):
        self.items.append(item)

    #method for popping an item
    def pop(self):
        return self.items.pop()

    #check what item is on top of the stack without removing it
    def peek(self):
        return self.items[-1]

    #method to get the size
    def size(self):
        return len(self.items)

    #to view the entire stack
    def fullStack(self):
        return self.items
```

```
#define a stack class
class Stack:
    def __init__(self):
        self.items = []

    #method to check the stack is empty or not
    def isEmpty(self):
        return self.items == []

    #method for pushing an item
    def push(self, item):
        self.items.append(item)

    #method for popping an item
    def pop(self):
        return self.items.pop()

    #check what item is on top of the stack without removing it
    def peek(self):
        return self.items[-1]

    #method to get the size
    def size(self):
        return len(self.items)

    #to view the entire stack
    def fullStack(self):
        return self.items
```


```
#define a stack class
class Stack:
    def __init__(self):
        self.items = []

    #method to check the stack is empty or not
    def isEmpty(self):
        return self.items == []

    #method for pushing an item
    def push(self, item):
        self.items.append(item)

    #method for popping an item
    def pop(self):
        return self.items.pop()

    #check what item is on top of the stack without removing it
    def peek(self):
        return self.items[-1]

    #method to get the size
    def size(self):
        return len(self.items)

    #to view the entire stack
    def fullStack(self):
        return self.items
```


- -

```
([])◦  ()<>))◦  ([)]◦
```

```
def checkParenth(str):
    stack = Stack()
    pushChars, popChars = "<({[", ">)}]"
    for c in str:
        if c in pushChars:
            stack.push(c)
        elif c in popChars:
            if stack.isEmpty():
                return False
            else:
                stackTop = stack.pop()
                # Checks to see whether the opening bracket matches the closing one
                balancingBracket = pushChars[popChars.index(c)]
                if stackTop != balancingBracket:
                    return False
        else:
            return False

    return not stack.isEmpty()
```

https://riptutorial.com/zh-TW/python/topic/3807/

103:

Examples

worker。

- `countUp()`1。
- `countDown()`1。

```
import multiprocessing
import time
from random import randint

def countUp():
    i = 0
    while i <= 3:
        print('Up:\t{}'.format(i))
        time.sleep(randint(1, 3)) # sleep 1, 2 or 3 seconds
        i += 1

def countDown():
    i = 3
    while i >= 0:
        print('Down:\t{}'.format(i))
        time.sleep(randint(1, 3)) # sleep 1, 2 or 3 seconds
        i -= 1

if __name__ == '__main__':
    # Initiate the workers.
    workerUp = multiprocessing.Process(target=countUp)
    workerDown = multiprocessing.Process(target=countDown)

    # Start the workers.
    workerUp.start()
    workerDown.start()

    # Join the workers. This will block in the main (parent) process
    # until the workers are complete.
    workerUp.join()
    workerDown.join()
```

```
import multiprocessing
import time
from random import randint

def countUp():
    i = 0
    while i <= 3:
        print('Up:\t{}'.format(i))
        time.sleep(randint(1, 3)) # sleep 1, 2 or 3 seconds
        i += 1

def countDown():
    i = 3
    while i >= 0:
```

```
        print('Down:\t{}'.format(i))
        time.sleep(randint(1, 3)) # sleep 1, 2 or 3 seconds
        i -= 1

if __name__ == '__main__':
    # Initiate the workers.
    workerUp = multiprocessing.Process(target=countUp)
    workerDown = multiprocessing.Process(target=countDown)

    # Start the workers.
    workerUp.start()
    workerDown.start()

    # Join the workers. This will block in the main (parent) process
    # until the workers are complete.
    workerUp.join()
    workerDown.join()
```

```
from multiprocessing import Pool

def cube(x):
    return x ** 3

if __name__ == "__main__":
    pool = Pool(5)
    result = pool.map(cube, [0, 1, 2, 3])
```

`PoolWorkers` ◦

`Pool(5)` 5Pool `pool.map` map◦

`map_async` `apply` `apply_async`◦

https://riptutorial.com/zh-TW/python/topic/3601/

104:

Examples

。 。 。 XYYX。

```
class Shape:
    """
    This is a parent class that is intended to be inherited by other classes
    """

    def calculate_area(self):
        """
        This method is intended to be overridden in subclasses.
        If a subclass doesn't implement it but it is called, NotImplemented will be raised.

        """
        raise NotImplemented

class Square(Shape):
    """
    This is a subclass of the Shape class, and represents a square
    """
    side_length = 2     # in this example, the sides are 2 units long

    def calculate_area(self):
        """
        This method overrides Shape.calculate_area(). When an object of type
        Square has its calculate_area() method called, this is the method that
        will be called, rather than the parent class' version.

        It performs the calculation necessary for this shape, a square, and
        returns the result.
        """
        return self.side_length * 2

class Triangle(Shape):
    """
    This is also a subclass of the Shape class, and it represents a triangle
    """
    base_length = 4
    height = 3

    def calculate_area(self):
        """
        This method also overrides Shape.calculate_area() and performs the area
        calculation for a triangle, returning the result.
        """

        return 0.5 * self.base_length * self.height

def get_area(input_obj):
    """
    This function accepts an input object, and will call that object's
    calculate_area() method. Note that the object type is not specified. It
    could be a Square, Triangle, or Shape object.
    """
```

```
    print(input_obj.calculate_area())

# Create one object of each class
shape_obj = Shape()
square_obj = Square()
triangle_obj = Triangle()

# Now pass each object, one at a time, to the get_area() function and see the
# result.
get_area(shape_obj)
get_area(square_obj)
get_area(triangle_obj)
```

4
6

◦ get_area()◦

```
class Shape:
    """
    This is a parent class that is intended to be inherited by other classes
    """

    def calculate_area(self):
        """
        This method is intended to be overridden in subclasses.
        If a subclass doesn't implement it but it is called, NotImplemented will be raised.

        """
        raise NotImplemented

class Square(Shape):
    """
    This is a subclass of the Shape class, and represents a square
    """
    side_length = 2     # in this example, the sides are 2 units long

    def calculate_area(self):
        """
        This method overrides Shape.calculate_area(). When an object of type
        Square has its calculate_area() method called, this is the method that
        will be called, rather than the parent class' version.

        It performs the calculation necessary for this shape, a square, and
        returns the result.
        """
        return self.side_length * 2

class Triangle(Shape):
    """
    This is also a subclass of the Shape class, and it represents a triangle
    """
    base_length = 4
    height = 3

    def calculate_area(self):
```

```
        """
        This method also overrides Shape.calculate_area() and performs the area
        calculation for a triangle, returning the result.
        """

        return 0.5 * self.base_length * self.height

def get_area(input_obj):
    """
    This function accepts an input object, and will call that object's
    calculate_area() method. Note that the object type is not specified. It
    could be a Square, Triangle, or Shape object.
    """

    print(input_obj.calculate_area())

# Create one object of each class
shape_obj = Shape()
square_obj = Square()
triangle_obj = Triangle()

# Now pass each object, one at a time, to the get_area() function and see the
# result.
get_area(shape_obj)
get_area(square_obj)
get_area(triangle_obj)
```

4
6

“”Python 3。 Python 2.x3.xPython 2.xtypeinput_obj。 “”object。

Python。 Python。

```
class Duck:
    def quack(self):
        print("Quaaaaaack!")
    def feathers(self):
        print("The duck has white and gray feathers.")

class Person:
    def quack(self):
        print("The person imitates a duck.")
    def feathers(self):
        print("The person takes a feather from the ground and shows it.")
    def name(self):
        print("John Smith")

def in_the_forest(obj):
    obj.quack()
    obj.feathers()

donald = Duck()
john = Person()
in_the_forest(donald)
in_the_forest(john)
```

Quaaaaaack

-
-
-

https://riptutorial.com/zh-TW/python/topic/5100/

105:

Examples

。

```
lst=[[1,2,3],[4,5,6],[7,8,9]]
```

lst。 [1,2,3] [4,5,6] [7,8,9] 。

```
lst=[[1,2,3],[4,5,6],[7,8,9]]
```

```
lst=[[1,2,3],[4,5,6],[7,8,9]]
```

[]。 0 0 [1,2,3] 。 []。 01 0 1 1 2

```
lst=[[1,2,3],[4,5,6],[7,8,9]]
```

[[10,11,12],[4,5,6],[7,8,9]] 。 。

```
lst=[[1,2,3],[4,5,6],[7,8,9]]
```

[[10,11,12],[4,5,15],[7,8,9]] 。 。 12615。

...

。

```
[[[111,112,113],[121,122,123],[131,132,133]],[[211,212,213],[221,222,223],[231,232,233]],[[311,312,313
```

。

```
[[[111,112,113],[121,122,123],[131,132,133]],[[211,212,213],[221,222,223],[231,232,233]],[[311,312,313
```

。

2D

```
[[[111,112,113],[121,122,123],[131,132,133]],[[211,212,213],[221,222,223],[231,232,233]],[[311,312,313
```

```
[[[111,112,113],[121,122,123],[131,132,133]],[[211,212,213],[221,222,223],[231,232,233]],[[311,312,313
```

https://riptutorial.com/zh-TW/python/topic/8186/

106:

Python。 。

Examples

threadingthreading.Thread。 threading.Thread

```
import threading

def foo():
  print "Hello threading!"

my_thread = threading.Thread(target=foo)
```

target。 Threadstart。

```
import threading

def foo():
  print "Hello threading!"

my_thread = threading.Thread(target=foo)
```

my_threadstartRuntimeError 。 start()daemon=True kwargmy_thread.daemonTrue Thread。

Thread.join()。

。

```
import threading

def foo():
  print "Hello threading!"

my_thread = threading.Thread(target=foo)
```

join()。

threading.ThreadThread。 run。

```
import threading

def foo():
  print "Hello threading!"

my_thread = threading.Thread(target=foo)
```

。

Queuequeue。

```
from queue import Queue
from threading import Thread

# create a data producer
def producer(output_queue):
    while True:
        data = data_computation()

        output_queue.put(data)

# create a consumer
def consumer(input_queue):
    while True:
        # retrieve data (blocking)
        data = input_queue.get()

        # do something with the data

        # indicate data has been consumed
        input_queue.task_done()
```


```
from queue import Queue
from threading import Thread

# create a data producer
def producer(output_queue):
    while True:
        data = data_computation()

        output_queue.put(data)

# create a consumer
def consumer(input_queue):
    while True:
        # retrieve data (blocking)
        data = input_queue.get()

        # do something with the data

        # indicate data has been consumed
        input_queue.task_done()
```


threadingqueue

```
from socket import socket, AF_INET, SOCK_STREAM
from threading import Thread
from queue import Queue

def echo_server(addr, nworkers):
    print('Echo server running at', addr)
    # 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)
```

concurrent.futures.Threadpoolexecutor

```
from socket import socket, AF_INET, SOCK_STREAM
from threading import Thread
from queue import Queue

def echo_server(addr, nworkers):
    print('Echo server running at', addr)
    # 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)
```

Python Cookbook3David BeazleyBrian K. JonesO'Reilly。2013 David BeazleyBrian Jones978-1-449-34037-7。

。

。 “”。

```
#!/usr/bin/env python2

import threading
import Queue
import time
import sys
import subprocess
from backports.shutil_get_terminal_size import get_terminal_size

printq = Queue.Queue()
interrupt = False
lines = []

def main():
```

```
    ptt = threading.Thread(target=printer) # Turn the printer on
    ptt.daemon = True
    ptt.start()

    # Stupid example of stuff to print
    for i in xrange(1,100):
        printq.put(' '.join([str(x) for x in range(1,i)]))           # The actual way to send
stuff to the printer
        time.sleep(.5)

def split_line(line, cols):
    if len(line) > cols:
        new_line = ''
        ww = line.split()
        i = 0
        while len(new_line) <= (cols - len(ww[i]) - 1):
            new_line += ww[i] + ' '
            i += 1
            print len(new_line)
        if new_line == '':
            return (line, '')

        return (new_line, ' '.join(ww[i:]))
    else:
        return (line, '')


def printer():

    while True:
        cols, rows = get_terminal_size() # Get the terminal dimensions
        msg = '#' + '-' * (cols - 2) + '#\n' # Create the
        try:
            new_line = str(printq.get_nowait())
            if new_line != '!@#EXIT#@!': # A nice way to turn the printer
                                         # thread out gracefully
                lines.append(new_line)
                printq.task_done()
            else:
                printq.task_done()
                sys.exit()
        except Queue.Empty:
            pass

        # Build the new message to show and split too long lines
        for line in lines:
            res = line          # The following is to split lines which are
                                # longer than cols.
            while len(res) !=0:
                toprint, res = split_line(res, cols)
                msg += '\n' + toprint

        # Clear the shell and print the new output
        subprocess.check_call('clear') # Keep the shell clean
        sys.stdout.write(msg)
        sys.stdout.flush()
        time.sleep(.5)
```

while

```
import threading
import time

class StoppableThread(threading.Thread):
    """Thread class with a stop() method. The thread itself has to check
    regularly for the stopped() condition."""

    def __init__(self):
        super(StoppableThread, self).__init__()
        self._stop_event = threading.Event()

    def stop(self):
        self._stop_event.set()

    def join(self, *args, **kwargs):
        self.stop()
        super(StoppableThread,self).join(*args, **kwargs)

    def run()
        while not self._stop_event.is_set():
            print("Still running!")
            time.sleep(2)
        print("stopped!"
```

。

https://riptutorial.com/zh-TW/python/topic/544/

107:

。 Python。

socket.AF_UNIX	UNIX
socket.AF_INET	IPv4
socket.AF_INET6	IPv6
socket.SOCK_STREAM	TCP
socket.SOCK_DGRAM	UDP

Examples

UDP

UDP。 。 。 UDP。

UDPlocalhost6667

“”UDP 。

```
from socket import socket, AF_INET, SOCK_DGRAM
s = socket(AF_INET, SOCK_DGRAM)
msg = ("Hello you there!").encode('utf-8')  # socket.sendto() takes bytes as input, hence we
must encode the string first.
s.sendto(msg, ('localhost', 6667))
```

UDP

UDP。 。 `socket.recvfrom` `msg` [] `addr` []

`socket`UDP

```
from socket import socket, AF_INET, SOCK_DGRAM
sock = socket(AF_INET, SOCK_DGRAM)
sock.bind(('localhost', 6667))

while True:
    msg, addr = sock.recvfrom(8192)  # This is the amount of bytes to read at maximum
    print("Got message from %s: %s" % (addr, msg))
```

`socketserver.UDPServer`

```
from socket import socket, AF_INET, SOCK_DGRAM
```

```
sock = socket(AF_INET, SOCK_DGRAM)
sock.bind(('localhost', 6667))

while True:
    msg, addr = sock.recvfrom(8192)  # This is the amount of bytes to read at maximum
    print("Got message from %s: %s" % (addr, msg))
```

sockets。 。

TCP

。 。

b'Hello'localhost6667TCP

```
from socket import socket, AF_INET, SOCK_STREAM
s = socket(AF_INET, SOCK_STREAM)
s.connect(('localhost', 6667))  # The address of the TCP server listening
s.send(b'Hello')
s.close()
```

“”。 。 send。

。

TCP

TCP5000127.0.0.1。 。

-c。 JSON。 -n。 。

client_list.py

```
import argparse
import json
import socket
import threading

def handle_client(client_list, conn, address):
    name = conn.recv(1024)
    entry = dict(zip(['name', 'address', 'port'], [name, address[0], address[1]]))
    client_list[name] = entry
    conn.sendall(json.dumps(client_list))
    conn.shutdown(socket.SHUT_RDWR)
    conn.close()

def server(client_list):
    print "Starting server..."
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    s.bind(('127.0.0.1', 5000))
    s.listen(5)
    while True:
        (conn, address) = s.accept()
```

```
        t = threading.Thread(target=handle_client, args=(client_list, conn, address))
        t.daemon = True
        t.start()

def client(name):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('127.0.0.1', 5000))
    s.send(name)
    data = s.recv(1024)
    result = json.loads(data)
    print json.dumps(result, indent=4)

def parse_arguments():
    parser = argparse.ArgumentParser()
    parser.add_argument('-c', dest='client', action='store_true')
    parser.add_argument('-n', dest='name', type=str, default='name')
    result = parser.parse_args()
    return result

def main():
    client_list = dict()
    args = parse_arguments()
    if args.client:
        client(args.name)
    else:
        try:
            server(client_list)
        except KeyboardInterrupt:
            print "Keyboard interrupt"

if __name__ == '__main__':
    main()
```

```
import argparse
import json
import socket
import threading

def handle_client(client_list, conn, address):
    name = conn.recv(1024)
    entry = dict(zip(['name', 'address', 'port'], [name, address[0], address[1]]))
    client_list[name] = entry
    conn.sendall(json.dumps(client_list))
    conn.shutdown(socket.SHUT_RDWR)
    conn.close()

def server(client_list):
    print "Starting server..."
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    s.bind(('127.0.0.1', 5000))
    s.listen(5)
    while True:
        (conn, address) = s.accept()
        t = threading.Thread(target=handle_client, args=(client_list, conn, address))
        t.daemon = True
        t.start()

def client(name):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
    s.connect(('127.0.0.1', 5000))
    s.send(name)
    data = s.recv(1024)
    result = json.loads(data)
    print json.dumps(result, indent=4)

def parse_arguments():
    parser = argparse.ArgumentParser()
    parser.add_argument('-c', dest='client', action='store_true')
    parser.add_argument('-n', dest='name', type=str, default='name')
    result = parser.parse_args()
    return result

def main():
    client_list = dict()
    args = parse_arguments()
    if args.client:
        client(args.name)
    else:
        try:
            server(client_list)
        except KeyboardInterrupt:
            print "Keyboard interrupt"

if __name__ == '__main__':
    main()
```

```
import argparse
import json
import socket
import threading

def handle_client(client_list, conn, address):
    name = conn.recv(1024)
    entry = dict(zip(['name', 'address', 'port'], [name, address[0], address[1]]))
    client_list[name] = entry
    conn.sendall(json.dumps(client_list))
    conn.shutdown(socket.SHUT_RDWR)
    conn.close()

def server(client_list):
    print "Starting server..."
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    s.bind(('127.0.0.1', 5000))
    s.listen(5)
    while True:
        (conn, address) = s.accept()
        t = threading.Thread(target=handle_client, args=(client_list, conn, address))
        t.daemon = True
        t.start()

def client(name):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('127.0.0.1', 5000))
    s.send(name)
    data = s.recv(1024)
    result = json.loads(data)
    print json.dumps(result, indent=4)
```

```
def parse_arguments():
    parser = argparse.ArgumentParser()
    parser.add_argument('-c', dest='client', action='store_true')
    parser.add_argument('-n', dest='name', type=str, default='name')
    result = parser.parse_args()
    return result

def main():
    client_list = dict()
    args = parse_arguments()
    if args.client:
        client(args.name)
    else:
        try:
            server(client_list)
        except KeyboardInterrupt:
            print "Keyboard interrupt"

if __name__ == '__main__':
    main()
```

1024。JSON。

```
import argparse
import json
import socket
import threading

def handle_client(client_list, conn, address):
    name = conn.recv(1024)
    entry = dict(zip(['name', 'address', 'port'], [name, address[0], address[1]]))
    client_list[name] = entry
    conn.sendall(json.dumps(client_list))
    conn.shutdown(socket.SHUT_RDWR)
    conn.close()

def server(client_list):
    print "Starting server..."
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    s.bind(('127.0.0.1', 5000))
    s.listen(5)
    while True:
        (conn, address) = s.accept()
        t = threading.Thread(target=handle_client, args=(client_list, conn, address))
        t.daemon = True
        t.start()

def client(name):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('127.0.0.1', 5000))
    s.send(name)
    data = s.recv(1024)
    result = json.loads(data)
    print json.dumps(result, indent=4)

def parse_arguments():
    parser = argparse.ArgumentParser()
    parser.add_argument('-c', dest='client', action='store_true')
    parser.add_argument('-n', dest='name', type=str, default='name')
```

```
    result = parser.parse_args()
    return result

def main():
    client_list = dict()
    args = parse_arguments()
    if args.client:
        client(args.name)
    else:
        try:
            server(client_list)
        except KeyboardInterrupt:
            print "Keyboard interrupt"

if __name__ == '__main__':
    main()
```

Linux

```
sudo ethtool -K eth1 tx off
```

SOCK_RAW

```
sudo ethtool -K eth1 tx off
```

https://riptutorial.com/zh-TW/python/topic/1530/

108:

- subprocess.callargs*stdin = Nonestdout = Nonestderr = Noneshell = Falsetimeout = None
- subprocess.Popenargsbufsize = -1executable = Nonestdin = Nonestdout = Nonestderr = Nonepreexec_fn = Noneclose_fds = Trueshell = Falsecwd = Noneenv = None universal_newlines = False startupinfo = Nonecreationflags = 0restore_signals = True start_new_session = Falsepass_fds =

`args`	- `'ls'` `['ls', '-la']`
`shell`	shell POSIX`/bin/sh`shell。
`cwd`	。

Examples

`subprocess.call`。 。 。 。

```
subprocess.call([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

shell`shell=True`。

```
subprocess.call([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

`exit status` 。 `shell=True` 。

`subprocess.callsubprocess.check_output` 。 。

Popen

`subprocess.Popensubprocess.call`。

```
process = subprocess.Popen([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

`Popencall`; `Popencall`。

```
process = subprocess.Popen([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

```
process = subprocess.Popen([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

stdinstdout。 。

```
process = subprocess.Popen([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

communicate()stdinstdout 。

```
process = subprocess.Popen([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

EOL。

```
process = subprocess.Popen([r'C:\path\to\app.exe', 'arg1', '--flag', 'arg'])
```

read1read1。 。 0read EOF 。

process.poll()None 。 。

stderr 。

shell_mode=True 。

。 shlex 。

```
import shlex
cmd_to_subprocess = shlex.split(command_used_in_the_shell)
```

```
import shlex
cmd_to_subprocess = shlex.split(command_used_in_the_shell)
```

https://riptutorial.com/zh-TW/python/topic/1393/

109:

- mydict = {}
- mydict [k] =
- value = mydict [k]
- value = mydict.getk
- value = mydict.getk“default_value”

-
- hash;TypeError
- 。

Examples

```
dictionary = {"Hello": 1234, "World": 5678}
print(dictionary["Hello"])
```

1234 。

"Hello" 。 dict。

dict1234 。 "Hello" dict 。

KeyError。 KeyErrordictionary.get。 None 。 None 。

```
dictionary = {"Hello": 1234, "World": 5678}
print(dictionary["Hello"])
```

wNone x"nuh-uh" 。

dict

dict()。

```
dict(a=1, b=2, c=3)                    # {'a': 1, 'b': 2, 'c': 3}
dict([('d', 4), ('e', 5), ('f', 6)])   # {'d': 4, 'e': 5, 'f': 6}
dict([('a', 1)], b=2, c=3)             # {'a': 1, 'b': 2, 'c': 3}
dict({'a' : 1, 'b' : 2}, c=3)          # {'a': 1, 'b': 2, 'c': 3}
```

KeyError

。 KeyError

```
mydict = {}
mydict['not there']
```

```
mydict = {}
mydict['not there']
```

dict.get。

```
mydict = {}
mydict['not there']
```

mydict[key] default_value。 mydictkey。 mydict.setdefault(key, default_value)。

```
mydict = {}
mydict['not there']
```

```
mydict = {}
mydict['not there']
```

in。

```
mydict = {}
mydict['not there']
```

。

dictcollections.defaultdictdefault_factorynew_keydict。

for。

```
mydict = {
    'a': '1',
    'b': '2'
}
```

keys()

```
mydict = {
    'a': '1',
    'b': '2'
}
```

values()

```
mydict = {
    'a': '1',
    'b': '2'
}
```

items()

```
mydict = {
    'a': '1',
    'b': '2'
}
```

dict keys() values()items()。 sort() sorted()OrderedDict 。

Python 2/3 Python 3Python 2 iterkeys() itervalues()iteritems()。 。 PEP 3106 。

Python 。 。 。 。

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

Python 3.x 3.5

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

dict

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

dict()

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

listdictionary

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

```
d = {}                          # empty dict
d = {'key': 'value'}            # dict with initial values
```

defaultdict

```
from collections import defaultdict

d = defaultdict(int)
d['key']                          # 0
d['key'] = 5
```

```
d['key']                        # 5

d = defaultdict(lambda: 'empty')
d['key']                        # 'empty'
d['key'] = 'full'
d['key']                        # 'full'
```

[*]`dict using dict.setdefault()`

```
from collections import defaultdict

d = defaultdict(int)
d['key']                        # 0
d['key'] = 5
d['key']                        # 5

d = defaultdict(lambda: 'empty')
d['key']                        # 'empty'
d['key'] = 'full'
d['key']                        # 'full'
```

`dict.setdefault()[]` - 。

[*] *Python Cookbook3David BeazleyBrian K. JonesO'Reilly。2013 David BeazleyBrian Jones978-1-449-34037-7。*

。

`collectionsOrderedDict` 。 。

```
from collections import OrderedDict

d = OrderedDict()
d['first'] = 1
d['second'] = 2
d['third'] = 3
d['last'] = 4

# Outputs "first 1", "second 2", "third 3", "last 4"
for key in d:
    print(key, d[key])
```

**

`**`。

```
>>>
>>> def parrot(voltage, state, action):
...     print("This parrot wouldn't", action, end=' ')
...     print("if you put", voltage, "volts through it.", end=' ')
...     print("E's", state, "!")
...
>>> d = {"voltage": "four million", "state": "bleedin' demised", "action": "VOOM"}
>>> parrot(**d)
```

```
This parrot wouldn't VOOM if you put four million volts through it. E's bleedin' demised !
```

Python 3.5`dict`。

```
>>>
>>> def parrot(voltage, state, action):
...     print("This parrot wouldn't", action, end=' ')
...     print("if you put", voltage, "volts through it.", end=' ')
...     print("E's", state, "!")
...
>>> d = {"voltage": "four million", "state": "bleedin' demised", "action": "VOOM"}
>>> parrot(**d)

This parrot wouldn't VOOM if you put four million volts through it. E's bleedin' demised !
```

lattermost“Clifford”“Nemo”。

```
>>> fish = {'name': "Nemo", 'hands': "fins", 'special': "gills"}
>>> dog = {'name': "Clifford", 'hands': "paws", 'color': "red"}
```

Python 3.5+

```
>>> fish = {'name': "Nemo", 'hands': "fins", 'special': "gills"}
>>> dog = {'name': "Clifford", 'hands': "paws", 'color': "red"}
```

lattermost“Clifford”“Nemo”。

Python 3.3+

```
>>> fish = {'name': "Nemo", 'hands': "fins", 'special': "gills"}
>>> dog = {'name': "Clifford", 'hands': "paws", 'color': "red"}
```

“Clifford”“Nemo”。

Python 2.x3.x

```
>>> fish = {'name': "Nemo", 'hands': "fins", 'special': "gills"}
>>> dog = {'name': "Clifford", 'hands': "paws", 'color': "red"}
```

lattermost`**`“Clifford”“Nemo”。

```
>>> fish = {'name': "Nemo", 'hands': "fins", 'special': "gills"}
```

```
>>> dog = {'name': "Clifford", 'hands': "paws", 'color': "red"}
```

`dict.update`dict。

。

```
role = {"By day": "A typical programmer",
        "By night": "Still a typical programmer", }
```

PEP 8。

```
options = {
    "x": ["a", "b"],
    "y": [10, 20, 30]
}
```

。 `"x"="a""y"=10` `"x"="a"` `"y"=10` 。

。

```
options = {
    "x": ["a", "b"],
    "y": [10, 20, 30]
}
```

`combinations`

```
options = {
    "x": ["a", "b"],
    "y": [10, 20, 30]
}
```

`for` 。

```
d = {'a': 1, 'b': 2, 'c':3}
for key in d:
    print(key, d[key])
# c 3
# b 2
# a 1
```

```
d = {'a': 1, 'b': 2, 'c':3}
for key in d:
    print(key, d[key])
# c 3
# b 2
# a 1
```

Python 3.x 3.0

`items()`

```
d = {'a': 1, 'b': 2, 'c':3}
for key in d:
    print(key, d[key])
# c 3
# b 2
# a 1
```

values()

```
d = {'a': 1, 'b': 2, 'c':3}
for key in d:
    print(key, d[key])
# c 3
# b 2
# a 1
```

Python 2.x 2.2

keys() values()items()iterkeys() itervalues()iteritems()。

-
- hash;TypeError
- 。

```
# Creating and populating it with values
stock = {'eggs': 5, 'milk': 2}

# Or creating an empty dictionary
dictionary = {}

# And populating it after
dictionary['eggs'] = 5
dictionary['milk'] = 2

# Values can also be lists
mydict = {'a': [1, 2, 3], 'b': ['one', 'two', 'three']}

# Use list.append() method to add new elements to the values list
mydict['a'].append(4)   # => {'a': [1, 2, 3, 4], 'b': ['one', 'two', 'three']}
mydict['b'].append('four')  # => {'a': [1, 2, 3, 4], 'b': ['one', 'two', 'three', 'four']}

# We can also create a dictionary using a list of two-items tuples
iterable = [('eggs', 5), ('milk', 2)]
dictionary = dict(iterables)

# Or using keyword argument:
dictionary = dict(eggs=5, milk=2)

# Another way will be to use the dict.fromkeys:
dictionary = dict.fromkeys((milk, eggs))  # => {'milk': None, 'eggs': None}
dictionary = dict.fromkeys((milk, eggs), (2, 5))  # => {'milk': 2, 'eggs': 5}
```

。

```
car = {}
car["wheels"] = 4
```

```
car["color"] = "Red"
car["model"] = "Corvette"
```

。

```
car = {}
car["wheels"] = 4
car["color"] = "Red"
car["model"] = "Corvette"
```

JSON

```
car = {}
car["wheels"] = 4
car["color"] = "Red"
car["model"] = "Corvette"
```

```
car = {}
car["wheels"] = 4
car["color"] = "Red"
car["model"] = "Corvette"
```

https://riptutorial.com/zh-TW/python/topic/396/

110:

- str.capitalize - > str
- str.casefold - > str [Python> 3.3]
- str.centerwidth [fillchar] - > str
- str.countsub [start [end]] - > int
- str.decodeencoding =“utf-8”[errors] - > unicode [Python 2.x]
- str.encodeencoding =“utf-8”errors =“strict” - > bytes
- str.endswithsuffix [start [end]] - > bool
- str.expandtabstabsize = 8 - > str
- str.findsub [start [end]] - > int
- str.format* args** kwargs - > str
- str.format_mapmapping - > str
- str.indexsub [start [end]] - > int
- str.isalnum - > bool
- str.isalpha - > bool
- str.isdecimal - > bool
- str.isdigit - > bool
- str.isidentifier - > bool
- str.islower - > bool
- str.isnumeric - > bool
- str.isprintable - > bool
- str.isspace - > bool
- str.istitle - > bool
- str.isupper - > bool
- str.joiniterable - > str
- str.ljustwidth [fillchar] - > str
- str.lower - > str
- str.lstrip[chars] - > str
- static str.maketransx [y [z]]
- str.partitionsep - >headseptail
- str.replaceoldnew [count] - > str
- str.rfindsub [start [end]] - > int
- str.rindexsub [start [end]] - > int
- str.rjustwidth [fillchar] - > str
- str.rpartitionsep - >headseptail
- str.rsplitsep = Nonemaxsplit = -1 - >
- str.rstrip[chars] - > str
- str.splitsep = Nonemaxsplit = -1 - >
- str.splitlines[keepends] - >
- str.startswith[start [end]] - > book
- str.strip[chars] - > str
- str.swapcase - > str
- str.title - > str
- str.translatetable - > str

- str.upper - > str
- str.zfillwidth - > str

◦ str str◦

Examples

Python◦

- str.casefold
- str.upper
- str.lower
- str.capitalize
- str.title
- str.swapcase

unicodePython 3 11◦ ◦

Python 3.x 3.3

str.casefold()

str.casefold◦ str.lower◦

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

CaseFolding.txtUnicode Consortium◦

---

str.upper()

str.upper

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

---

str.lower()

str.lower;

```
"XßΣ".casefold()
# 'xssσ'
```

```
"XßΣ".lower()
# 'xßς'
```

---

str.capitalize()

str.capitalize

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

---

str.title()

str.title

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

---

str.swapcase()

str.swapcase

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

---

str

strstr.upper

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

map。

```
"XßΣ".casefold()
# 'xssσ'

"XßΣ".lower()
# 'xßς'
```

str.split(sep=None, maxsplit=-1)

str.split。 sep。

sep None 。

```
>>> "This is a sentence.".split()
['This', 'is', 'a', 'sentence.']

>>> " This is    a sentence.  ".split()
['This', 'is', 'a', 'sentence.']

>>> "            ".split()
[]
```

sep。 。 。

```
>>> "This is a sentence.".split()
['This', 'is', 'a', 'sentence.']

>>> " This is    a sentence.  ".split()
['This', 'is', 'a', 'sentence.']

>>> "            ".split()
[]
```

maxsplit。 -1

```
>>> "This is a sentence.".split()
['This', 'is', 'a', 'sentence.']

>>> " This is    a sentence.  ".split()
['This', 'is', 'a', 'sentence.']

>>> "            ".split()
[]
```

---

str.rsplit(sep=None, maxsplit=-1)

maxsplit str.rsplit “right split”str.split “left split”。

```
>>> "This is a sentence.".split()
['This', 'is', 'a', 'sentence.']

>>> " This is    a sentence.  ".split()
['This', 'is', 'a', 'sentence.']

>>> "            ".split()
[]
```

Python。 。

Pythonstr。 re.sub 。

---

str.replace(old, new[, count])

str.replaceoldnewoldnew。 count

'foo''spam' old = 'foo'new = 'spam'str.replace

```
>>> "Make sure to foo your sentence.".replace('foo', 'spam')
"Make sure to spam your sentence."
```

oldnew

```
>>> "Make sure to foo your sentence.".replace('foo', 'spam')
"Make sure to spam your sentence."
```

count 。 count

```
>>> "Make sure to foo your sentence.".replace('foo', 'spam')
"Make sure to spam your sentence."
```

str.formatf-strings

Python2.6`str.format`3.6f-strings。

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

PythonC。 `str.format`

```
i = 10
f = 1.5
```

```
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

str.format。

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

python pyformat.info。

{}

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

```
i = 10
f = 1.5
s = "foo"
l = ['a', 1, 2]
d = {'a': 1, 2: 'foo'}
```

。 str.format()PEP 3101PEP 498 f。

str.countstr.count 。

---

str.count(sub[, start[, end]])

str.countintsub。 startend。 start = 0end = len(str)

```
>>> s = "She sells seashells by the seashore."
>>> s.count("sh")
2
>>> s.count("se")
3
>>> s.count("sea")
2
>>> s.count("seashells")
1
```

start

endstart13

```
>>> s = "She sells seashells by the seashore."
>>> s.count("sh")
2
>>> s.count("se")
3
>>> s.count("sea")
2
>>> s.count("seashells")
1
```


```
>>> s = "She sells seashells by the seashore."
>>> s.count("sh")
2
>>> s.count("se")
3
>>> s.count("sea")
2
>>> s.count("seashells")
1
```


Python`str.startswith()str.endswith()` ◦

`str.startswith(prefix[, start[, end]])`

`str.startswithprefix`◦

```
>>> s = "This is a test string"
>>> s.startswith("T")
True
>>> s.startswith("Thi")
True
>>> s.startswith("thi")
False
```

`startend`◦ 22


```
>>> s = "This is a test string"
>>> s.startswith("T")
True
>>> s.startswith("Thi")
True
>>> s.startswith("thi")
False
```


`s[2] == 'i's[3] == 's'True` ◦

`tuple`


```
>>> s = "This is a test string"
>>> s.startswith("T")
True
>>> s.startswith("Thi")
```

```
True
>>> s.startswith("thi")
False
```

---

str.endswith(prefix[, start[, end]])

str.endswithstr.startswith◦

```
>>> s = "This is a test string"
>>> s.startswith("T")
True
>>> s.startswith("Thi")
True
>>> s.startswith("thi")
False
```

startswith

```
>>> s = "This is a test string"
>>> s.startswith("T")
True
>>> s.startswith("Thi")
True
>>> s.startswith("thi")
False
```

tuple

```
>>> s = "This is a test string"
>>> s.startswith("T")
True
>>> s.startswith("Thi")
True
>>> s.startswith("thi")
False
```

Python str◦ str.isalpha str.isdigit str.isalnum str.isspace ◦ str.isupper str.islower str.istitlestr.istitle ◦

---

str.isalpha

str.isalphaTrue

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

"".isalpha()False

。

---

str.isupper str.islower str.istitle

。

str.isupperTrueFalse 。

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

str.islowerTrueFalse 。

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

str.istitleTrue ;。

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

---

str.isdecimal str.isdigit str.isnumeric

str.isdecimal。

str.isdigit。

str.isnumeric0-9。

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
```

```
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

BytestringsPython 3`bytes` Python 2`str` `isdigit` ASCII。

`str.isalpha` `False` 。

`str.isalnum`

`str.isalphastr.isnumeric` `True`

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

`str.isspace`

`True` 。

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

“”

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
```

```
>>> "HelloWorld".isalpha()
True
```

strip

```
>>> "Hello World".isalpha()  # contains a space
False
>>> "Hello2World".isalpha()  # contains a number
False
>>> "HelloWorld!".isalpha()  # contains punctuation
False
>>> "HelloWorld".isalpha()
True
```

str.translate

Pythonstrtranslate。

```
str.translate(table[, deletechars])
```

table	。
deletechars	。

maketrans str.maketransPython 3string.maketransPython 2。

```
str.translate(table[, deletechars])
```

translate。

---

tableNone 。

```
str.translate(table[, deletechars])
```

/

str.strip str.rstripstr.lstrip 。

---

str.strip([chars])

str.stripcharschars ;chars“ None 。

```
>>> "    a line with leading and trailing space     ".strip()
'a line with leading and trailing space'
```

chars

。

```
>>> "    a line with leading and trailing space     ".strip()
'a line with leading and trailing space'
```

---

str.rstrip([chars])str.lstrip([chars])

str.strip()。 str.rstrip()str.lstrip()。

str.rstrip

```
>>> "    a line with leading and trailing space     ".strip()
'a line with leading and trailing space'
```

str.lstrip

```
>>> "    a line with leading and trailing space     ".strip()
'a line with leading and trailing space'
```

。 unicodePython 3。 Python 2Python 3 - unicode。

unicode。 text.lower() != text.upper().lower()"ß"

```
>>> "ß".lower()
'ß'

>>> "ß".upper().lower()
'ss'
```

"Buße""BUSSE""Buße"。 "BUSSE""BU E" - 。 casefold

Python 3.x 3.3

```
>>> "ß".lower()
'ß'

>>> "ß".upper().lower()
'ss'
```

lower 。 casefold.upper().lower()。

。 "ê" == "ê " -

```
>>> "ß".lower()
'ß'

>>> "ß".upper().lower()
'ss'
```

```
>>> "ß".lower()
```

```
'ß'

>>> "ß".upper().lower()
'ss'
```

`unicodedata.normalize` ◦ **NFKD**◦

```
>>> "ß".lower()
'ß'

>>> "ß".upper().lower()
'ss'
```


```
>>> "ß".lower()
'ß'

>>> "ß".upper().lower()
'ss'
```


`join()`◦ ◦

```
>>> " ".join(["once","upon","a","time"])
"once upon a time"
```

◦


```
>>> " ".join(["once","upon","a","time"])
"once upon a time"
```


Python`string`◦ `string`

```
>>> import string
```

`string.ascii_letters`

`ascii_lowercaseascii_uppercase`

```
>>> import string
```

`string.ascii_lowercase`

ASCII

```
>>> import string
```

`string.ascii_uppercase`

ASCII

```
>>> import string
```

---

string.digits

```
>>> import string
```

---

string.hexdigits

```
>>> import string
```

---

string.octaldigits

```
>>> import string
```

---

string.punctuation

C

```
>>> import string
```

---

string.whitespace

ASCII

```
>>> import string
```

`print(string.whitespace)str`。

---

string.printable

; `string.digits` `string.ascii_letters` `string.punctuationstring.whitespace` 。

```
>>> import string
```

reverse `reversed()reversed()` 。

```
>>> reversed('hello')
<reversed object at 0x0000000000000000>
>>> [char for char in reversed('hello')]
['o', 'l', 'l', 'e', 'h']
```

`reversed()''.join()` 。

```
>>> reversed('hello')
<reversed object at 0x0000000000000000>
>>> [char for char in reversed('hello')]
['o', 'l', 'l', 'e', 'h']
```

`reversed()`Python`-1`。

```
>>> reversed('hello')
<reversed object at 0x0000000000000000>
>>> [char for char in reversed('hello')]
['o', 'l', 'l', 'e', 'h']
```

Python。

`str.ljuststr.rjust`

```
interstates_lengths = {
    5: (1381, 2222),
    19: (63, 102),
    40: (2555, 4112),
    93: (189,305),
}
for road, length in interstates_lengths.items():
    miles,kms = length
    print('{} -> {} mi. ({} km.)'.format(str(road).rjust(4), str(miles).ljust(4),
str(kms).ljust(4)))
```

```
interstates_lengths = {
    5: (1381, 2222),
    19: (63, 102),
    40: (2555, 4112),
    93: (189,305),
}
for road, length in interstates_lengths.items():
    miles,kms = length
    print('{} -> {} mi. ({} km.)'.format(str(road).rjust(4), str(miles).ljust(4),
str(kms).ljust(4)))
```

`ljustrjust`。 `widthfillchar`。 `width`。 `width` alread。 `fillchar' '`。

`ljustfillcharwidth`。 `rjustrjust`。 `lr`*`fillchar`*。

strunicode

。 unicode。

Python 2strUnicode。 `'' ""`ASCIIASCII。 Unicode`u'' u""`。

Python 2.x 2.3

```
# You get "© abc" encoded in UTF-8 from a file, network, or other data source
```

```
s = '\xc2\xa9 abc'  # s is a byte array, not a string of characters
                    # Doesn't know the original was UTF-8
                    # Default form of string literals in Python 2
s[0]                # '\xc2' - meaningless byte (without context such as an encoding)
type(s)             # str - even though it's not a useful one w/o having a known encoding

u = s.decode('utf-8')  # u'\xa9 abc'
                       # Now we have a Unicode string, which can be read as UTF-8 and printed
properly
                       # In Python 2, Unicode string literals need a leading u
                       # str.decode converts a string which may contain escaped bytes to a
Unicode string
u[0]                # u'\xa9' - Unicode Character 'COPYRIGHT SIGN' (U+00A9) '©'
type(u)             # unicode

u.encode('utf-8')   # '\xc2\xa9 abc'
                    # unicode.encode produces a string with escaped bytes for non-ASCII
characters
```

Python 3“”Unicode。 Unicodebytestring`b''` `b""`。 `True``isinstance(some_val, byte)` `some_val`。

Python 3.x 3.0

```
# You get "© abc" encoded in UTF-8 from a file, network, or other data source

s = '\xc2\xa9 abc'  # s is a byte array, not a string of characters
                    # Doesn't know the original was UTF-8
                    # Default form of string literals in Python 2
s[0]                # '\xc2' - meaningless byte (without context such as an encoding)
type(s)             # str - even though it's not a useful one w/o having a known encoding

u = s.decode('utf-8')  # u'\xa9 abc'
                       # Now we have a Unicode string, which can be read as UTF-8 and printed
properly
                       # In Python 2, Unicode string literals need a leading u
                       # str.decode converts a string which may contain escaped bytes to a
Unicode string
u[0]                # u'\xa9' - Unicode Character 'COPYRIGHT SIGN' (U+00A9) '©'
type(u)             # unicode

u.encode('utf-8')   # '\xc2\xa9 abc'
                    # unicode.encode produces a string with escaped bytes for non-ASCII
characters
```

Python。 `in`

```
>>> "foo" in "foo.baz.bar"
True
```

`True`

```
>>> "foo" in "foo.baz.bar"
True
```

https://riptutorial.com/zh-TW/python/topic/278/

111:

。 Python。

- “{}”。 format42==>“42”
- “{0}”。 format42==>“42”
- “{0.2f}”。 format42==>“42.00”
- “{0.0f}”。 42.1234==>“42”
- “{answer}”。 formatno_answer = 41answer = 42==>“42”
- “{answer.2f}”。 no_answer = 41answer = 42==>“42.00”
- “{[key]}”。 format{'key''value'}==>“value”
- “{[1]}”。 format['zero''one''two']==>“one”
- “{answer} = {answer}”。 formatanswer = 42==>“42 = 42”
- ''。 join['stack''overflow']==>“”

- PyFormat.info/。

Examples

```
foo = 1
bar = 'bar'
baz = 3.14
```

`str.format`。

```
foo = 1
bar = 'bar'
baz = 3.14
```

。 `str.format`0。

```
foo = 1
bar = 'bar'
baz = 3.14
```

```
foo = 1
bar = 'bar'
baz = 3.14
```

`str.format`

```
foo = 1
bar = 'bar'
baz = 3.14
```

```
foo = 1
```

```
bar = 'bar'
baz = 3.14
```

listtuple

```
foo = 1
bar = 'bar'
baz = 3.14
```

str.format Python% PEP 3101 - ◦ str.format%◦

◦ : ◦ ◦ :~^20 ^20

```
foo = 1
bar = 'bar'
baz = 3.14
```

format%

```
foo = 1
bar = 'bar'
baz = 3.14
```

format

```
foo = 1
bar = 'bar'
baz = 3.14
```

Python 2.x 2.6

format()◦ :[fill_char][align_operator][width]align_operator

- <width◦
- >width◦
- ^width◦
- =◦

fill_char ◦

```
'{:~<9s}, World'.format('Hello')
# 'Hello~~~~, World'

'{:~>9s}, World'.format('Hello')
# '~~~~Hello, World'

'{:~^9s}'.format('Hello')
# '~~Hello~~'

'{:0=6d}'.format(-123)
# '-00123'
```

`ljust()` `rjust()` `center()` `zfill()` `ljust()`2.5。

f-string

PEP 498 Python3.6`f.format`。

```
>>> foo = 'bar'
>>> f'Foo is {foo}'
'Foo is bar'
```

。

```
>>> foo = 'bar'
>>> f'Foo is {foo}'
'Foo is bar'
```

`f''b''bytes u''unicode`python2。 。

```
>>> foo = 'bar'
>>> f'Foo is {foo}'
'Foo is bar'
```

f-string。

```
>>> foo = 'bar'
>>> f'Foo is {foo}'
'Foo is bar'
```

datetime

`__format__`。 Python`datetimestr.formatstr.format strftime`

```
>>> from datetime import datetime
>>> 'North America: {dt:%m/%d/%Y}.  ISO: {dt:%Y-%m-%d}.'.format(dt=datetime.now())
'North America: 07/21/2016.  ISO: 2016-07-21.'
```

。

GetitemGetattr

`__getitem__`

```
person = {'first': 'Arthur', 'last': 'Dent'}
'{p[first]} {p[last]}'.format(p=person)
# 'Arthur Dent'
```

`getattr()`

```
person = {'first': 'Arthur', 'last': 'Dent'}
```

```
'{p[first]} {p[last]}'.format(p=person)
# 'Arthur Dent'
```

```
>>> '{0:.0f}'.format(42.12345)
'42'

>>> '{0:.1f}'.format(42.12345)
'42.1'

>>> '{0:.3f}'.format(42.12345)
'42.123'

>>> '{0:.5f}'.format(42.12345)
'42.12345'

>>> '{0:.7f}'.format(42.12345)
'42.1234500'
```


```
>>> '{0:.0f}'.format(42.12345)
'42'

>>> '{0:.1f}'.format(42.12345)
'42.1'

>>> '{0:.3f}'.format(42.12345)
'42.123'

>>> '{0:.5f}'.format(42.12345)
'42.12345'

>>> '{0:.7f}'.format(42.12345)
'42.1234500'
```

```
>>> '{0:.0f}'.format(42.12345)
'42'

>>> '{0:.1f}'.format(42.12345)
'42.1'

>>> '{0:.3f}'.format(42.12345)
'42.123'

>>> '{0:.5f}'.format(42.12345)
'42.12345'

>>> '{0:.7f}'.format(42.12345)
'42.1234500'
```


`{0}{name}`。 *1*


```
>>> '{0:.0f}'.format(42.12345)
'42'

>>> '{0:.1f}'.format(42.12345)
'42.1'

>>> '{0:.3f}'.format(42.12345)
```

```
'42.123'

>>> '{0:.5f}'.format(42.12345)
'42.12345'

>>> '{0:.7f}'.format(42.12345)
'42.1234500'
```

`.format()`

```
>>> '{:c}'.format(65)    # Unicode character
'A'

>>> '{:d}'.format(0x0a)  # base 10
'10'

>>> '{:n}'.format(0x0a)  # base 10 using current locale for separators
'10'
```


```
>>> '{:c}'.format(65)    # Unicode character
'A'

>>> '{:d}'.format(0x0a)  # base 10
'10'

>>> '{:n}'.format(0x0a)  # base 10 using current locale for separators
'10'
```


RGB


```
>>> '{:c}'.format(65)    # Unicode character
'A'

>>> '{:d}'.format(0x0a)  # base 10
'10'

>>> '{:n}'.format(0x0a)  # base 10 using current locale for separators
'10'
```

```
>>> '{:c}'.format(65)    # Unicode character
'A'

>>> '{:d}'.format(0x0a)  # base 10
'10'

>>> '{:n}'.format(0x0a)  # base 10 using current locale for separators
'10'
```


`str.formatformat。 。`

`format`Python `__format____format__。 __format__format。`

`__str____format__。` 1

```
object.__format__(self, format_spec)
```

```
object.__format__(self, format_spec)
```

`__format__`format**Python2**`__str__`、`__repr__`repr s。 **Python3** `format`、`__format__`。

```
>>> '{:.>10}'.format('foo')
'.......foo'
```

format{}{}

```
>>> '{:.>10}'.format('foo')
'.......foo'
```

`'{:{}{}{}}''{:*^15}'` ”*15“`'foo'` 。

```
>>> '{:.>10}'.format('foo')
'.......foo'
```

3。

`{}`。

```
s = """

pad
{{:3}}             :{a:3}:

truncate
{{:.3}}            :{e:.3}:

combined
{{:>3.3}}          :{a:>3.3}:
{{:3.3}}           :{a:3.3}:
{{:3.3}}           :{c:3.3}:
{{:3.3}}           :{e:3.3}:
"""

print (s.format(a="1"*1, c="3"*3, e="5"*5))
```

```
s = """

pad
{{:3}}             :{a:3}:

truncate
{{:.3}}            :{e:.3}:

combined
{{:>3.3}}          :{a:>3.3}:
{{:3.3}}           :{a:3.3}:
{{:3.3}}           :{c:3.3}:
{{:3.3}}           :{e:3.3}:
"""

print (s.format(a="1"*1, c="3"*3, e="5"*5))
```

format。

Python 2.x

```
>>> data = {'first': 'Hodor', 'last': 'Hodor!'}
>>> '{first} {last}'.format(**data)
'Hodor Hodor!'
```

Python 3.2+

```
>>> data = {'first': 'Hodor', 'last': 'Hodor!'}
>>> '{first} {last}'.format(**data)
'Hodor Hodor!'
```

str.format_mapstr.format_map。 datadict 。

```
>>> data = {'first': 'Hodor', 'last': 'Hodor!'}
>>> '{first} {last}'.format(**data)
'Hodor Hodor!'
```

https://riptutorial.com/zh-TW/python/topic/1019/

112:

Python。 Python/。

- hashlib.new
- hashlib.pbkdf2_hmacnamepasswordsaltroundsdklen = None

hashlib。 hashlib.new().update()hashlib.pbkdf2_hmac 。 b

```
  "This is a string"
 b"This is a buffer of bytes"
```

Examples

hashlibnew。

```
import hashlib

h = hashlib.new('sha256')
h.update(b'Nobody expects the Spanish Inquisition.')
h.digest()
# ==>
b'.\xdf\xda\xdaVR[\x12\x90\xff\x16\xfb\x17D\xcf\xb4\x82\xdd)\x14\xff\xbc\xb6Iy\x0c\x0eX\x9eF-
='
```

digestupdate。 hexdigest

```
import hashlib

h = hashlib.new('sha256')
h.update(b'Nobody expects the Spanish Inquisition.')
h.digest()
# ==>
b'.\xdf\xda\xdaVR[\x12\x90\xff\x16\xfb\x17D\xcf\xb4\x82\xdd)\x14\xff\xbc\xb6Iy\x0c\x0eX\x9eF-
='
```

hashlib.new。 Pythonhashlib.algorithms_available

```
import hashlib
hashlib.algorithms_available
# ==> {'sha256', 'DSA-SHA', 'SHA512', 'SHA224', 'dsaWithSHA', 'SHA', 'RIPEMD160', 'ecdsa-with-
SHA1', 'sha1', 'SHA384', 'md5', 'SHA1', 'MD5', 'MD4', 'SHA256', 'sha384', 'md4', 'ripemd160',
'sha224', 'sha512', 'DSA', 'dsaEncryption', 'sha', 'whirlpool'}
```

;。

hashlib.algorithms_guaranteed

```
import hashlib
```

```
hashlib.algorithms_available
# ==> {'sha256', 'DSA-SHA', 'SHA512', 'SHA224', 'dsaWithSHA', 'SHA', 'RIPEMD160', 'ecdsa-with-SHA1', 'sha1', 'SHA384', 'md5', 'SHA1', 'MD5', 'MD4', 'SHA256', 'sha384', 'md4', 'ripemd160', 'sha224', 'sha512', 'DSA', 'dsaEncryption', 'sha', 'whirlpool'}
```

hashlibPBKDF2。 。

```
import hashlib
import os

salt = os.urandom(16)
hash = hashlib.pbkdf2_hmac('sha256', b'password', salt, 100000)
```

PBKDF2SHA256。 。 。 。

binascii

```
import hashlib
import os

salt = os.urandom(16)
hash = hashlib.pbkdf2_hmac('sha256', b'password', salt, 100000)
```

PBKDF2bcryptscrypt。 Python。

。 。 。

hashlib

```
import hashlib

hasher = hashlib.new('sha256')
with open('myfile', 'r') as f:
    contents = f.read()
    hasher.update(contents)

print hasher.hexdigest()
```

```
import hashlib

hasher = hashlib.new('sha256')
with open('myfile', 'r') as f:
    contents = f.read()
    hasher.update(contents)

print hasher.hexdigest()
```

pycrypto

Python。 pycrypto 。 AES 。

```
import hashlib
import math
```

```
import os

from Crypto.Cipher import AES

IV_SIZE = 16    # 128 bit, fixed for the AES algorithm
KEY_SIZE = 32   # 256 bit meaning AES-256, can also be 128 or 192 bits
SALT_SIZE = 16  # This size is arbitrary

cleartext = b'Lorem ipsum'
password = b'highly secure encryption password'
salt = os.urandom(SALT_SIZE)
derived = hashlib.pbkdf2_hmac('sha256', password, salt, 100000,
                              dklen=IV_SIZE + KEY_SIZE)
iv = derived[0:IV_SIZE]
key = derived[IV_SIZE:]

encrypted = salt + AES.new(key, AES.MODE_CFB, iv).encrypt(cleartext)
```

AESIV。AES。。PBKDF2128256。

。。。

```
import hashlib
import math
import os

from Crypto.Cipher import AES

IV_SIZE = 16    # 128 bit, fixed for the AES algorithm
KEY_SIZE = 32   # 256 bit meaning AES-256, can also be 128 or 192 bits
SALT_SIZE = 16  # This size is arbitrary

cleartext = b'Lorem ipsum'
password = b'highly secure encryption password'
salt = os.urandom(SALT_SIZE)
derived = hashlib.pbkdf2_hmac('sha256', password, salt, 100000,
                              dklen=IV_SIZE + KEY_SIZE)
iv = derived[0:IV_SIZE]
key = derived[IV_SIZE:]

encrypted = salt + AES.new(key, AES.MODE_CFB, iv).encrypt(cleartext)
```

pycryptoRSA

RSA。RSA。 -。pycrypto。

```
import errno

from Crypto.Hash import SHA256
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5

message = b'This message is from me, I promise.'

try:
    with open('privkey.pem', 'r') as f:
        key = RSA.importKey(f.read())
```

```
except IOError as e:
    if e.errno != errno.ENOENT:
        raise
    # No private key, generate a new one. This can take a few seconds.
    key = RSA.generate(4096)
    with open('privkey.pem', 'wb') as f:
        f.write(key.exportKey('PEM'))
    with open('pubkey.pem', 'wb') as f:
        f.write(key.publickey().exportKey('PEM'))

hasher = SHA256.new(message)
signer = PKCS1_v1_5.new(key)
signature = signer.sign(hasher)
```

```
import errno

from Crypto.Hash import SHA256
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5

message = b'This message is from me, I promise.'

try:
    with open('privkey.pem', 'r') as f:
        key = RSA.importKey(f.read())
except IOError as e:
    if e.errno != errno.ENOENT:
        raise
    # No private key, generate a new one. This can take a few seconds.
    key = RSA.generate(4096)
    with open('privkey.pem', 'wb') as f:
        f.write(key.exportKey('PEM'))
    with open('pubkey.pem', 'wb') as f:
        f.write(key.publickey().exportKey('PEM'))

hasher = SHA256.new(message)
signer = PKCS1_v1_5.new(key)
signature = signer.sign(hasher)
```

PKCS1 v1.5。 pycryptoPKCS1 PSS `PKCS1_PSSPKCS1_PSSPKCS1_v1_5`。 。

pycryptoRSA

。 。 pycrypto 。

```
from Crypto.Cipher import PKCS1_OAEP
from Crypto.PublicKey import RSA

message = b'This is a very secret message.'

with open('pubkey.pem', 'rb') as f:
    key = RSA.importKey(f.read())
cipher = PKCS1_OAEP.new(key)
encrypted = cipher.encrypt(message)
```

```
from Crypto.Cipher import PKCS1_OAEP
```

```
from Crypto.PublicKey import RSA

message = b'This is a very secret message.'

with open('pubkey.pem', 'rb') as f:
    key = RSA.importKey(f.read())
cipher = PKCS1_OAEP.new(key)
encrypted = cipher.encrypt(message)
```

PKCS1 OAEP。 pycryptoPKCS1 v1.5。

https://riptutorial.com/zh-TW/python/topic/2598/

113: /

。 IDEAMODE CTRPython/。

RSA。 HashingSHA-1。 。 IDEA。

Python 2.7 https //www.python.org/downloads/

* **PyCrypto** https //pypi.python.org/pypi/pycrypto

* **PyCryptoPlus** https //github.com/doegox/python-cryptoplus

PyCrypto。 linuxalt + ctrl + tCMDshift ++Windows。 python setup.py installWindowsSure Python

PyCryptoPlus。

。 。

- 。 “import socket”IP。

  - - - - - - - - - -

  ```
  server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  host = raw_input("Server Address To Be Connected -> ")
  port = int(input("Port of The Server -> "))
  server.connect((host, port))
  ```

  - - - - - - - - - -

  ```
  server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  host = raw_input("Server Address To Be Connected -> ")
  port = int(input("Port of The Server -> "))
  server.connect((host, port))
  ```

  “socket.AF_INETsocket.SOCK_STREAM”accept。 “socket.AF_INET socket.SOCK_DGRAM”setblockingvalue。

- CLIENT。 。 CryptoRandomCrypto.PublicKeyRSA。

```
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
server.connect((host, port))
```

random_generator“ ”。 “ **Crypto.PublicKeyRSA** ”1024。 Public。

- CLIENTSHA-1。 SHA-1“import hashlib”。

  ```
  server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  ```

```
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
server.connect((host, port))
```

hash_objecthex_digest。hex_digestpublicServer。。。rsa`server_public_key = RSA.importKey(getpbk)` getpbk。

- 。“os”“key = os.urandom16”16“AES.MODE_CTR”SHA-1

```
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
server.connect((host, port))
```

en_digest。

- SERVER。

```
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
server.connect((host, port))
```

。

- CLIENT。eval。。

```
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
server.connect((host, port))
```

SHA-1。

IDEAMODE_CTRKEY。IDEA.MODE_CTR。

- IDEA16bit。MODE_CTR。40IDEA。。pythonstring [valuevalue]。。“key [16]”016。[1:17] [16]。IDEA.newIDEA3。KEYIDEAIDEA.MODE_CTRcounter =。counter =。counter =。lambdaKEY。Counter.Utillambdacounter =。Counter.Utilcrypto。

```
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
server.connect((host, port))
```

“ideaEncrypt”IDEA。

```
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
```

```
  server.connect((host, port))
```

wholeeMsg。HEXADECIMALupper。。

-

。。。

```
  server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  host = raw_input("Server Address To Be Connected -> ")
  port = int(input("Port of The Server -> "))
  server.connect((host, port))
```

。

Examples

```
import socket
import hashlib
import os
import time
import itertools
import threading
import sys
import Crypto.Cipher.AES as AES
from Crypto.PublicKey import RSA
from CryptoPlus.Cipher import IDEA

#server address and port number input from admin
host= raw_input("Server Address - > ")
port = int(input("Port - > "))
#boolean for checking server and port
check = False
done = False

def animate():
    for c in itertools.cycle(['....','.......','..........','............']):
        if done:
            break
        sys.stdout.write('\rCHECKING IP ADDRESS AND NOT USED PORT '+c)
        sys.stdout.flush()
        time.sleep(0.1)
    sys.stdout.write('\r -----SERVER STARTED. WAITING FOR CLIENT-----\n')
try:
    #setting up socket
    server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    server.bind((host,port))
    server.listen(5)
    check = True
except BaseException:
    print "-----Check Server Address or Port-----"
    check = False

if check is True:
    # server Quit
    shutdown = False
```

```
# printing "Server Started Message"
thread_load = threading.Thread(target=animate)
thread_load.start()

time.sleep(4)
done = True
#binding client and address
client,address = server.accept()
print ("CLIENT IS CONNECTED. CLIENT'S ADDRESS ->",address)
print ("\n-----WAITING FOR PUBLIC KEY & PUBLIC KEY HASH-----\n")

#client's message(Public Key)
getpbk = client.recv(2048)

#conversion of string to KEY
server_public_key = RSA.importKey(getpbk)

#hashing the public key in server side for validating the hash from client
hash_object = hashlib.sha1(getpbk)
hex_digest = hash_object.hexdigest()

if getpbk != "":
    print (getpbk)
    client.send("YES")
    gethash = client.recv(1024)
    print ("\n-----HASH OF PUBLIC KEY----- \n"+gethash)
if hex_digest == gethash:
    # creating session key
    key_128 = os.urandom(16)
    #encrypt CTR MODE session key
    en = AES.new(key_128,AES.MODE_CTR,counter = lambda:key_128)
    encrypto = en.encrypt(key_128)
    #hashing sha1
    en_object = hashlib.sha1(encrypto)
    en_digest = en_object.hexdigest()

    print ("\n-----SESSION KEY-----\n"+en_digest)

    #encrypting session key and public key
    E = server_public_key.encrypt(encrypto,16)
    print ("\n-----ENCRYPTED PUBLIC KEY AND SESSION KEY-----\n"+str(E))
    print ("\n-----HANDSHAKE COMPLETE-----")
    client.send(str(E))
    while True:
        #message from client
        newmess = client.recv(1024)
        #decoding the message from HEXADECIMAL to decrypt the ecrypted version of the message
only
        decoded = newmess.decode("hex")
        #making en_digest(session_key) as the key
        key = en_digest[:16]
        print ("\nENCRYPTED MESSAGE FROM CLIENT -> "+newmess)
        #decrypting message from the client
        ideaDecrypt = IDEA.new(key, IDEA.MODE_CTR, counter=lambda: key)
        dMsg = ideaDecrypt.decrypt(decoded)
        print ("\n**New Message**  "+time.ctime(time.time()) +" > "+dMsg+"\n")
        mess = raw_input("\nMessage To Client -> ")
        if mess != "":
            ideaEncrypt = IDEA.new(key, IDEA.MODE_CTR, counter=lambda : key)
            eMsg = ideaEncrypt.encrypt(mess)
            eMsg = eMsg.encode("hex").upper()
```

```
            if eMsg != "":
                print ("ENCRYPTED MESSAGE TO CLIENT-> " + eMsg)
            client.send(eMsg)
    client.close()
else:
    print ("\n-----PUBLIC KEY HASH DOESNOT MATCH-----\n")
```

```
import time
import socket
import threading
import hashlib
import itertools
import sys
from Crypto import Random
from Crypto.PublicKey import RSA
from CryptoPlus.Cipher import IDEA

#animating loading
done = False
def animate():
    for c in itertools.cycle(['....','.......','..........','............']):
        if done:
            break
        sys.stdout.write('\rCONFIRMING CONNECTION TO SERVER '+c)
        sys.stdout.flush()
        time.sleep(0.1)

#public key and private key
random_generator = Random.new().read
key = RSA.generate(1024,random_generator)
public = key.publickey().exportKey()
private = key.exportKey()

#hashing the public key
hash_object = hashlib.sha1(public)
hex_digest = hash_object.hexdigest()

#Setting up socket
server = socket.socket(socket.AF_INET,socket.SOCK_STREAM)

#host and port input user
host = raw_input("Server Address To Be Connected -> ")
port = int(input("Port of The Server -> "))
#binding the address and port
server.connect((host, port))
# printing "Server Started Message"
thread_load = threading.Thread(target=animate)
thread_load.start()

time.sleep(4)
done = True

def send(t,name,key):
    mess = raw_input(name + " : ")
    key = key[:16]
    #merging the message and the name
    whole = name+" : "+mess
    ideaEncrypt = IDEA.new(key, IDEA.MODE_CTR, counter=lambda : key)
    eMsg = ideaEncrypt.encrypt(whole)
    #converting the encrypted message to HEXADECIMAL to readable
```

```
    eMsg = eMsg.encode("hex").upper()
    if eMsg != "":
        print ("ENCRYPTED MESSAGE TO SERVER-> "+eMsg)
    server.send(eMsg)
def recv(t,key):
    newmess = server.recv(1024)
    print ("\nENCRYPTED MESSAGE FROM SERVER-> " + newmess)
    key = key[:16]
    decoded = newmess.decode("hex")
    ideaDecrypt = IDEA.new(key, IDEA.MODE_CTR, counter=lambda: key)
    dMsg = ideaDecrypt.decrypt(decoded)
    print ("\n**New Message From Server**  " + time.ctime(time.time()) + " : " + dMsg + "\n")

while True:
    server.send(public)
    confirm = server.recv(1024)
    if confirm == "YES":
        server.send(hex_digest)

    #connected msg
    msg = server.recv(1024)
    en = eval(msg)
    decrypt = key.decrypt(en)
    # hashing sha1
    en_object = hashlib.sha1(decrypt)
    en_digest = en_object.hexdigest()

    print ("\n-----ENCRYPTED PUBLIC KEY AND SESSION KEY FROM SERVER-----")
    print (msg)
    print ("\n-----DECRYPTED SESSION KEY-----")
    print (en_digest)
    print ("\n-----HANDSHAKE COMPLETE-----\n")
    alais = raw_input("\nYour Name -> ")

    while True:
        thread_send = threading.Thread(target=send,args=("------Sending Message------
",alais,en_digest))
        thread_recv = threading.Thread(target=recv,args=("------Recieving Message------
",en_digest))
        thread_send.start()
        thread_recv.start()

        thread_send.join()
        thread_recv.join()
        time.sleep(0.5)
    time.sleep(60)
    server.close()
```

/ https://riptutorial.com/zh-TW/python/topic/8710/-

114:

Examples

CHello World

Chello.chellogreet()

```
#include <Python.h>
#include <stdio.h>

#if PY_MAJOR_VERSION >= 3
#define IS_PY3K
#endif

static PyObject *hello_greet(PyObject *self, PyObject *args)
{
    const char *input;
    if (!PyArg_ParseTuple(args, "s", &input)) {
        return NULL;
    }
    printf("%s", input);
    Py_RETURN_NONE;
}

static PyMethodDef HelloMethods[] = {
    { "greet", hello_greet, METH_VARARGS, "Greet the user" },
    { NULL, NULL, 0, NULL }
};

#ifdef IS_PY3K
static struct PyModuleDef hellomodule = {
    PyModuleDef_HEAD_INIT, "hello", NULL, -1, HelloMethods
};

PyMODINIT_FUNC PyInit_hello(void)
{
    return PyModule_Create(&hellomodule);
}
#else
PyMODINIT_FUNC inithello(void)
{
    (void) Py_InitModule("hello", HelloMethods);
}
#endif
```

gcc

gcc /path/to/your/file/hello.c -o /path/to/your/file/hello

greet()hello.py

```
#include <Python.h>
#include <stdio.h>

#if PY_MAJOR_VERSION >= 3
```

```
#define IS_PY3K
#endif

static PyObject *hello_greet(PyObject *self, PyObject *args)
{
    const char *input;
    if (!PyArg_ParseTuple(args, "s", &input)) {
        return NULL;
    }
    printf("%s", input);
    Py_RETURN_NONE;
}

static PyMethodDef HelloMethods[] = {
    { "greet", hello_greet, METH_VARARGS, "Greet the user" },
    { NULL, NULL, 0, NULL }
};

#ifdef IS_PY3K
static struct PyModuleDef hellomodule = {
    PyModuleDef_HEAD_INIT, "hello", NULL, -1, HelloMethods
};

PyMODINIT_FUNC PyInit_hello(void)
{
    return PyModule_Create(&hellomodule);
}
#else
PyMODINIT_FUNC inithello(void)
{
    (void) Py_InitModule("hello", HelloMethods);
}
#endif
```

C Extensions

PythonC。

`PyObject_AsFileDescriptor`

```
PyObject *fobj;
int fd = PyObject_AsFileDescriptor(fobj);
if (fd < 0){
    return NULL;
}
```

python`PyFile_FromFd`。

```
PyObject *fobj;
int fd = PyObject_AsFileDescriptor(fobj);
if (fd < 0){
    return NULL;
}
```

Cc ++Boost

C ++BoostC。

C ++

hello.cppC ++

```
#include <boost/python/module.hpp>
#include <boost/python/list.hpp>
#include <boost/python/class.hpp>
#include <boost/python/def.hpp>

// Return a hello world string.
std::string get_hello_function()
{
   return "Hello world!";
}

// hello class that can return a list of count hello world strings.
class hello_class
{
public:

   // Taking the greeting message in the constructor.
   hello_class(std::string message) : _message(message) {}

   // Returns the message count times in a python list.
   boost::python::list as_list(int count)
   {
      boost::python::list res;
      for (int i = 0; i < count; ++i) {
         res.append(_message);
      }
      return res;
   }

private:
   std::string _message;
};


// Defining a python module naming it to "hello".
BOOST_PYTHON_MODULE(hello)
{
   // Here you declare what functions and classes that should be exposed on the module.

   // The get_hello_function exposed to python as a function.
   boost::python::def("get_hello", get_hello_function);

   // The hello_class exposed to python as a class.
   boost::python::class_<hello_class>("Hello", boost::python::init<std::string>())
      .def("as_list", &hello_class::as_list)
      ;
}
```

pythonpythonboost。 Ubuntu 12.04python 3.4gcc。 Boost。 Ubuntu

```
#include <boost/python/module.hpp>
#include <boost/python/list.hpp>
#include <boost/python/class.hpp>
#include <boost/python/def.hpp>

// Return a hello world string.
std::string get_hello_function()
{
   return "Hello world!";
}

// hello class that can return a list of count hello world strings.
class hello_class
{
public:

   // Taking the greeting message in the constructor.
   hello_class(std::string message) : _message(message) {}

   // Returns the message count times in a python list.
   boost::python::list as_list(int count)
   {
      boost::python::list res;
      for (int i = 0; i < count; ++i) {
         res.append(_message);
      }
      return res;
   }

private:
   std::string _message;
};


// Defining a python module naming it to "hello".
BOOST_PYTHON_MODULE(hello)
{
   // Here you declare what functions and classes that should be exposed on the module.

   // The get_hello_function exposed to python as a function.
   boost::python::def("get_hello", get_hello_function);

   // The hello_class exposed to python as a class.
   boost::python::class_<hello_class>("Hello", boost::python::init<std::string>())
      .def("as_list", &hello_class::as_list)
      ;
}
```

.sopython

```
#include <boost/python/module.hpp>
#include <boost/python/list.hpp>
#include <boost/python/class.hpp>
#include <boost/python/def.hpp>

// Return a hello world string.
std::string get_hello_function()
{
   return "Hello world!";
}
```

```
// hello class that can return a list of count hello world strings.
class hello_class
{
public:

   // Taking the greeting message in the constructor.
   hello_class(std::string message) : _message(message) {}

   // Returns the message count times in a python list.
   boost::python::list as_list(int count)
   {
      boost::python::list res;
      for (int i = 0; i < count; ++i) {
         res.append(_message);
      }
      return res;
   }

private:
   std::string _message;
};


// Defining a python module naming it to "hello".
BOOST_PYTHON_MODULE(hello)
{
   // Here you declare what functions and classes that should be exposed on the module.

   // The get_hello_function exposed to python as a function.
   boost::python::def("get_hello", get_hello_function);

   // The hello_class exposed to python as a class.
   boost::python::class_<hello_class>("Hello", boost::python::init<std::string>())
      .def("as_list", &hello_class::as_list)
      ;
}
```

example.pypython

```
#include <boost/python/module.hpp>
#include <boost/python/list.hpp>
#include <boost/python/class.hpp>
#include <boost/python/def.hpp>

// Return a hello world string.
std::string get_hello_function()
{
   return "Hello world!";
}

// hello class that can return a list of count hello world strings.
class hello_class
{
public:

   // Taking the greeting message in the constructor.
   hello_class(std::string message) : _message(message) {}

   // Returns the message count times in a python list.
```

```
   boost::python::list as_list(int count)
   {
      boost::python::list res;
      for (int i = 0; i < count; ++i) {
         res.append(_message);
      }
      return res;
   }

private:
   std::string _message;
};


// Defining a python module naming it to "hello".
BOOST_PYTHON_MODULE(hello)
{
   // Here you declare what functions and classes that should be exposed on the module.

   // The get_hello_function exposed to python as a function.
   boost::python::def("get_hello", get_hello_function);

   // The hello_class exposed to python as a class.
   boost::python::class_<hello_class>("Hello", boost::python::init<std::string>())
      .def("as_list", &hello_class::as_list)
      ;
}
```

python3 example.py

```
#include <boost/python/module.hpp>
#include <boost/python/list.hpp>
#include <boost/python/class.hpp>
#include <boost/python/def.hpp>

// Return a hello world string.
std::string get_hello_function()
{
   return "Hello world!";
}

// hello class that can return a list of count hello world strings.
class hello_class
{
public:

   // Taking the greeting message in the constructor.
   hello_class(std::string message) : _message(message) {}

   // Returns the message count times in a python list.
   boost::python::list as_list(int count)
   {
      boost::python::list res;
      for (int i = 0; i < count; ++i) {
         res.append(_message);
      }
      return res;
   }

private:
```

```
   std::string _message;
};


// Defining a python module naming it to "hello".
BOOST_PYTHON_MODULE(hello)
{
   // Here you declare what functions and classes that should be exposed on the module.

   // The get_hello_function exposed to python as a function.
   boost::python::def("get_hello", get_hello_function);

   // The hello_class exposed to python as a class.
   boost::python::class_<hello_class>("Hello", boost::python::init<std::string>())
      .def("as_list", &hello_class::as_list)
      ;
}
```

https://riptutorial.com/zh-TW/python/topic/557/

115: PythonSQL Server

Examples

```
$ pip install pymssql
```

```
import pymssql

SERVER = "servername"
USER = "username"
PASSWORD = "password"
DATABASE = "dbname"

connection = pymssql.connect(server=SERVER, user=USER,
                password=PASSWORD, database=DATABASE)

cursor = connection.cursor() # to access field as dictionary use cursor(as_dict=True)
cursor.execute("SELECT TOP 1 * FROM TableName")
row = cursor.fetchone()

######## CREATE TABLE ########
cursor.execute("""
CREATE TABLE posts (
    post_id INT PRIMARY KEY NOT NULL,
    message TEXT,
    publish_date DATETIME
)
""")

######## INSERT DATA IN TABLE ########
cursor.execute("""
    INSERT INTO posts VALUES(1, "Hey There", "11.23.2016")
""")
# commit your work to database
connection.commit()

######## ITERATE THROUGH RESULTS  ########
cursor.execute("SELECT TOP 10 * FROM posts ORDER BY publish_date DESC")
for row in cursor:
    print("Message: " + row[1] + " | " + "Date: " + row[2])
    # if you pass as_dict=True to cursor
    # print(row["message"])

connection.close()
```

SQLexecuteCRUD。

with pymssql.org

PythonSQL Server https://riptutorial.com/zh-TW/python/topic/7985/pythonsql-server

116:

- import *module_name*
- import *module_name.submodule_name*
- *module_name* import *
- from *module_name* import *submodule_name* [*class_name* *function_name* ... etc]
- *module_namesome_namenew_name*
- *module_name.submodule_name* import *class_name* [*function_name* ... etc]

Python◦ `if __name__ == '__main__':`◦

Examples

`import`

```
>>> import random
>>> print(random.randint(1, 10))
4
```

`import modulemodule.name` - ◦ `randomrandint`◦ `randomrandom.randintrandint` ◦

```
>>> import random
>>> print(random.randint(1, 10))
4
```

Python`main.pycustom.py` ◦

```
>>> import random
>>> print(random.randint(1, 10))
4
```

```
>>> import random
>>> print(random.randint(1, 10))
4
```

`import`

```
>>> import random
>>> print(random.randint(1, 10))
4
```

python◦ `import from` ◦ `import` ◦ `hello.pyworld.pyfunction` ◦ `import`◦

```
>>> import random
>>> print(random.randint(1, 10))
4
```

`import`◦

```
>>> import random
>>> print(random.randint(1, 10))
4
```

from

```
>>> import random
>>> print(random.randint(1, 10))
4
```

```
>>> import random
>>> print(random.randint(1, 10))
4
```

```
from random import randint # Syntax "from MODULENAME import NAME1[, NAME2[, ...]]"
print(randint(1, 10))      # Out: 5
```

from random python import randint。

```
from random import randint # Syntax "from MODULENAME import NAME1[, NAME2[, ...]]"
print(randint(1, 10))      # Out: 5
```

```
from random import randint # Syntax "from MODULENAME import NAME1[, NAME2[, ...]]"
print(randint(1, 10))      # Out: 5
```

```
from random import randint # Syntax "from MODULENAME import NAME1[, NAME2[, ...]]"
print(randint(1, 10))      # Out: 5
```

pythonrandom。 import random

```
from random import randint # Syntax "from MODULENAME import NAME1[, NAME2[, ...]]"
print(randint(1, 10))      # Out: 5
```

```
from module_name import *
```

```
from module_name import *
```

math。

。 from math import sqrt, ceilfrom math import sqrt, ceil

```
from module_name import *
```

。 SyntaxError 。

```
from module_name import *
```

```
from module_name import *
```


```
from module_name import *
```


__all__

`__all__from mymodule import *from mymodule import *`。

```
# mymodule.py

__all__ = ['imported_by_star']

imported_by_star = 42
not_imported_by_star = 21
```

`from mymodule import *imported_by_star` imported_by_star

```
# mymodule.py

__all__ = ['imported_by_star']

imported_by_star = 42
not_imported_by_star = 21
```

`not_imported_by_star`

```
# mymodule.py

__all__ = ['imported_by_star']

imported_by_star = 42
not_imported_by_star = 21
```

Python 2.x 2.7

`importlib`2.7Python

```
import importlib
random = importlib.import_module("random")
```

`importlib.import_module()`

```
import importlib
random = importlib.import_module("random")
```

Python`imp`。

Python 2.x 2.7

`imp.find_moduleimp.load_module`。

```
import importlib
random = importlib.import_module("random")
```

`__import__()sys.modules fromlistimportlib.import_module()`。

Python`sys.path`。 python。

```
import sys
sys.path.append("/path/to/directory/containing/your/module")
import mymodule
```

`mymodule`。

PEP8

PEP8

1. ```
   from math import sqrt, ceil      # Not recommended
   from math import sqrt            # Recommended
   from math import ceil
   ```
2.
   -
   -
   - /
3. 。 `from module import * module`。 `from module import *`。
4. ;。

```
from module.submodule import function
```

`module.submodulefunction`。

__import__

`__import__()`

```
if user_input == "os":
    os = __import__("os")

# equivalent to import os
```


```
if user_input == "os":
    os = __import__("os")

# equivalent to import os
```


。 。

import
```

```
import math
math.pi = 3
print(math.pi)    # 3
import math
print(math.pi)    # 3
```

◦ ◦ import

```
import math
math.pi = 3
print(math.pi)    # 3
import math
print(math.pi)    # 3
```

◦

Python 2

reload

Python 2.x 2.3

```
import math
math.pi = 3
print(math.pi)    # 3
import math
print(math.pi)    # 3
```

Python 3

reloadimportlib

Python 3.x 3.0

```
import math
math.pi = 3
print(math.pi)    # 3
import math
print(math.pi)    # 3
```

https://riptutorial.com/zh-TW/python/topic/249/

117:

Python 2object。

Examples

@property

@property。 。

foobar.py

```
class Foo(object):
    def __init__(self):
        self.__bar = None

    @property
    def bar(self):
        if self.__bar is None:
            self.__bar = some_expensive_lookup_operation()
        return self.__bar
```

```
class Foo(object):
    def __init__(self):
        self.__bar = None

    @property
    def bar(self):
        if self.__bar is None:
            self.__bar = some_expensive_lookup_operation()
        return self.__bar
```

@property

@property

```
class Cash(object):
    def __init__(self, value):
        self.value = value
    @property
    def formatted(self):
        return '${:.2f}'.format(self.value)
    @formatted.setter
    def formatted(self, new):
        self.value = float(new[1:])
```

```
class Cash(object):
    def __init__(self, value):
        self.value = value
```

```
    @property
    def formatted(self):
        return '${:.2f}'.format(self.value)
    @formatted.setter
    def formatted(self, new):
        self.value = float(new[1:])
```

gettersetterdeleter

`getter setterdeleter`

```
class BaseClass(object):
    @property
    def foo(self):
        return some_calculated_value()

    @foo.setter
    def foo(self, value):
        do_something_with_value(value)


class DerivedClass(BaseClass):
    @BaseClass.foo.setter
    def foo(self, value):
        do_something_different_with_value(value)
```

setterdeleter。

@。。Python 3.x

```
class A:
    p = 1234
    def getX (self):
        return self._x

    def setX (self, value):
        self._x = value

    def getY (self):
        return self._y

    def setY (self, value):
        self._y = 1000 + value    # Weird but possible

    def getY2 (self):
        return self._y

    def setY2 (self, value):
        self._y = value

    def getT    (self):
        return self._t

    def setT (self, value):
        self._t = value

    def getU (self):
```

```
        return self._u + 10000

    def setU (self, value):
        self._u = value - 5000

    x, y, y2 = property (getX, setX), property (getY, setY), property (getY2, setY2)
    t = property (getT, setT)
    u = property (getU, setU)

A.q = 5678

class B:
    def getZ (self):
        return self.z_

    def setZ (self, value):
        self.z_ = value

    z = property (getZ, setZ)

class C:
    def __init__ (self):
        self.offset = 1234

    def getW (self):
        return self.w_ + self.offset

    def setW (self, value):
        self.w_ = value - self.offset

    w = property (getW, setW)

a1 = A ()
a2 = A ()

a1.y2 = 1000
a2.y2 = 2000

a1.x = 5
a1.y = 6

a2.x = 7
a2.y = 8

a1.t = 77
a1.u = 88

print (a1.x, a1.y, a1.y2)
print (a2.x, a2.y, a2.y2)
print (a1.p, a2.p, a1.q, a2.q)

print (a1.t, a1.u)

b = B ()
c = C ()

b.z = 100100
c.z = 200200
c.w = 300300

print (a1.x, b.z, c.z, c.w)
```

```
c.w = 400400
c.z = 500500
b.z = 600600

print (a1.x, b.z, c.z, c.w)
```

https://riptutorial.com/zh-TW/python/topic/2050/

118:

- `x.title # Accesses the title attribute using the dot notation`
- `x.title = "Hello World" # Sets the property of the title attribute using the dot notation`
- `@property # Used as a decorator before the getter method for properties`
- `@title.setter # Used as a decorator before the setter method for properties`

Examples

。

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

book1 = Book(title="Right Ho, Jeeves", author="P.G. Wodehouse")
```

Python 。

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

book1 = Book(title="Right Ho, Jeeves", author="P.G. Wodehouse")
```

Python

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

book1 = Book(title="Right Ho, Jeeves", author="P.G. Wodehouse")
```

。 gettersetter。

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
```

。 gettersetter。 “”。 。

*author*gettersetter。

```
class Book:
    def __init__(self, title, author):
        self.title = title
```

```
        self.author = author
```

。

public。

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
```

。 Python。 @property。 setter@ attributeName.setter。

。

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
```

Python。 Python。

```
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author
```

https://riptutorial.com/zh-TW/python/topic/4392/

119:

Examples

。 falsey。

```
x = True
y = True
z = x and y # z = True

x = True
y = False
z = x and y # z = False

x = False
y = True
z = x and y # z = False

x = False
y = False
z = x and y # z = False

x = 1
y = 1
z = x and y # z = y, so z = 1, see `and` and `or` are not guaranteed to be a boolean

x = 0
y = 1
z = x and y # z = x, so z = 0 (see above)

x = 1
y = 0
z = x and y # z = y, so z = 0 (see above)

x = 0
y = 0
z = x and y # z = x, so z = 0 (see above)
```

`1`truthy `0`falsey。

truthy。 。

```
x = True
y = True
z = x or y # z = True

x = True
y = False
z = x or y # z = True

x = False
y = True
z = x or y # z = True

x = False
```

```
y = False
z = x or y # z = False

x = 1
y = 1
z = x or y # z = x, so z = 1, see `and` and `or` are not guaranteed to be a boolean

x = 1
y = 0
z = x or y # z = x, so z = 1 (see above)

x = 0
y = 1
z = x or y # z = y, so z = 1 (see above)

x = 0
y = 0
z = x or y # z = y, so z = 0 (see above)
```

`1`truthy `0`falsey。

```
x = True
y = not x # y = False

x = False
y = not x # y = True
```

Python 。

```
>>> def true_func():
...     print("true_func()")
...     return True
...
>>> def false_func():
...     print("false_func()")
...     return False
...
>>> true_func() or false_func()
true_func()
True
>>> false_func() or true_func()
false_func()
true_func()
True
>>> true_func() and false_func()
true_func()
false_func()
False
>>> false_func() and false_func()
false_func()
False
```

``

`or` 。 `or`

```
def or_(a, b):
    if a:
        return a
    else:
        return b
```

`and`

```
def or_(a, b):
    if a:
        return a
    else:
        return b
```

Python -

```
if 3.14 < x < 3.142:
    print("x is near pi")
```

`(3.14 < x) < 3.142` Python`3.14 < x and x < 3.142` 。

https://riptutorial.com/zh-TW/python/topic/1731/

120:

Python。 Python。

Python。

Examples

for

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

list.pop()。

。

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

。

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

。 alist 。

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

break'a' 。 。

---

for **list**

```
alist = [0, 1, 2]
```

```
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

item ◦ alist[2] enumerate()

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

---

while

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

zlist;

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

len()x

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

i ◦ zlist[i]1izlist[i] ◦

◦ while

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

。 。

Python /。

```
alist = [0, 1, 2]
for index, value in enumerate(alist):
    alist.pop(index)
print(alist)
# Out: [1]
```

```
def foo(li=[]):
    li.append(1)
    print(li)

foo([2])
# Out: [2, 1]
foo([3])
# Out: [3, 1]
```

```
def foo(li=[]):
    li.append(1)
    print(li)

foo([2])
# Out: [2, 1]
foo([3])
# Out: [3, 1]
```

。 li。

```
def foo(li=[]):
    li.append(1)
    print(li)

foo([2])
# Out: [2, 1]
foo([3])
# Out: [3, 1]
```

if not liFalse 。

```
def foo(li=[]):
    li.append(1)
    print(li)

foo([2])
# Out: [2, 1]
foo([3])
# Out: [3, 1]
```

None

```
def foo(li=[]):
    li.append(1)
```

```
    print(li)

foo([2])
# Out: [2, 1]
foo([3])
# Out: [3, 1]
```

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

3。 1

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

1li。

[[]] * 3list3list。 list list3。 li[0] li。

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

idlist

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

3list3。 id

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

。 append。

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

。

```
li = [[]] * 3
print(li)
```

```
# Out: [[], [], []]
```

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

for。

。

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

“==”TrueFalse

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

Pythonistas“”。

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

Python 2.6+

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

。 try/except。 。

Python。 duck typing 。 str()。 list()iterable。

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

foo。 bar。 。

object

。 Python 2.x。 。 。 “”。 object。

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```


```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```


Python 3.x。

init

JavaPHP。 。 Python。 。 。

- class。
- 。


```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```

```
li = [[]] * 3
print(li)
# Out: [[], [], []]
```


Python。

```
>>> -8 is (-7 - 1)
False
>>> -3 is (-2 - 1)
True
```


```
>>> -8 is (-7 - 1)
False
>>> -3 is (-2 - 1)
True
```


isTrue -3 256 -8 257 。

[-5, 256]。 isTrue ;False。

- 。

== identity is 。

Python`is`。

```
>>> -8 is (-7 - 1)
False
>>> -3 is (-2 - 1)
True
```

`'python'`Python`'python'`。

。

```
>>> -8 is (-7 - 1)
False
>>> -3 is (-2 - 1)
True
```

`==` identity `is` 。

int literals

Python。 `7`。 `bit_length`。 。

```
x = 7
x.bit_length()
# Out: 3
```

`7.bit_length()``SyntaxError`。 `7.27.bit_length()`。 。

`int` literals

```
x = 7
x.bit_length()
# Out: 3
```

`7..bit_length()` `float``bit_length()`。

`float`“”`float.`

```
x = 7
x.bit_length()
# Out: 3
```

```
if a == 3 or b == 3 or c == 3:
```

```
if a == 3 or b == 3 or c == 3:
```

; `or`== `if (a) or (b) or (c == 3):`

```
if a == 3 or b == 3 or c == 3:
```

`any()or`

```
if a == 3 or b == 3 or c == 3:
```

```
if a == 3 or b == 3 or c == 3:
```

```
if a == 3 or b == 3 or c == 3:
```

`in`。

```
if a == 3 or b == 3 or c == 3:
```

```
if a == 3 or b == 3 or c == 3:
```

sys.argv [0]

`sys.argv[0]`python。 。

```
# script.py
import sys

print(sys.argv[0])
print(sys.argv)
```

```
# script.py
import sys

print(sys.argv[0])
print(sys.argv)
```

PythonC ++ `std::map`

```
myDict = {'first': 1, 'second': 2, 'third': 3}
print(myDict)
# Out: {'first': 1, 'second': 2, 'third': 3}

print([k for k in myDict])
# Out: ['second', 'third', 'first']
```

Python。

/`collections.OrderedDict`

```
myDict = {'first': 1, 'second': 2, 'third': 3}
print(myDict)
# Out: {'first': 1, 'second': 2, 'third': 3}

print([k for k in myDict])
```

```
# Out: ['second', 'third', 'first']
```

`OrderedDict`。 。

Python 3.6。

Python 3.x 3.6

```
myDict = {'first': 1, 'second': 2, 'third': 3}
print(myDict)
# Out: {'first': 1, 'second': 2, 'third': 3}

print([k for k in myDict])
# Out: ['second', 'third', 'first']
```

“ ”。

GIL

Python GIL。 。

```
import math
from threading import Thread

def calc_fact(num):
    math.factorial(num)

num = 600000
t = Thread(target=calc_fact, daemon=True, args=[num])
print("About to calculate: {}!".format(num))
t.start()
print("Calculating...")
t.join()
print("Calculated")
```

`Calculating...`。 C `math.factorial` GIL。

。 Python。 。 C。

```
import math
from threading import Thread

def calc_fact(num):
    math.factorial(num)

num = 600000
t = Thread(target=calc_fact, daemon=True, args=[num])
print("About to calculate: {}!".format(num))
t.start()
print("Calculating...")
t.join()
print("Calculated")
```

`sleep`。 C。

```
import math
from threading import Thread

def calc_fact(num):
    math.factorial(num)

num = 600000
t = Thread(target=calc_fact, daemon=True, args=[num])
print("About to calculate: {}!".format(num))
t.start()
print("Calculating...")
t.join()
print("Calculated")
```

Python 2.x 2.7

```
i = 0
a = [i for i in range(3)]
print(i) # Outputs 2
```

Python 2“” 。 ***Python 3*** 。

Python 3.x 3.0

```
i = 0
a = [i for i in range(3)]
print(i) # Outputs 2
```

for

```
i = 0
a = [i for i in range(3)]
print(i) # Outputs 2
```

Python 2Python 3。

。

xyzab

```
def xyz():
  return a, b
```

xyzxyz

```
def xyz():
  return a, b
```

tABt 。

TypeErrortype tuple...

```
def xyz():
  return a, b
```

Pythonic JSON

```
my_var = 'bla';
api_key = 'key';
...lots of code here...
params = {"language": "en", my_var: api_key}
```

JavaScriptPython。 JavaScript`params`

```
my_var = 'bla';
api_key = 'key';
...lots of code here...
params = {"language": "en", my_var: api_key}
```

Python

```
my_var = 'bla';
api_key = 'key';
...lots of code here...
params = {"language": "en", my_var: api_key}
```

`my_var`。

https://riptutorial.com/zh-TW/python/topic/3553/

121: CPython

CPython。

NuGet**MongoDB.Bson**。

JSON。

- JSON

```
 BsonDocument argsBson = BsonDocument.Parse("{ 'x' : '1', 'y' : '2' }");
 string argsFile = string.Format("{0}\\{1}.txt", Path.GetDirectoryName(pyScriptPath),
Guid.NewGuid());
```

- Pythonpython.exepythonJSON

```
 BsonDocument argsBson = BsonDocument.Parse("{ 'x' : '1', 'y' : '2' }");
 string argsFile = string.Format("{0}\\{1}.txt", Path.GetDirectoryName(pyScriptPath),
Guid.NewGuid());
```

- PythonJSON

```
 BsonDocument argsBson = BsonDocument.Parse("{ 'x' : '1', 'y' : '2' }");
 string argsFile = string.Format("{0}\\{1}.txt", Path.GetDirectoryName(pyScriptPath),
Guid.NewGuid());
```

- CJSON

```
 BsonDocument argsBson = BsonDocument.Parse("{ 'x' : '1', 'y' : '2' }");
 string argsFile = string.Format("{0}\\{1}.txt", Path.GetDirectoryName(pyScriptPath),
Guid.NewGuid());
```

CPythonExcelPython。

ExcelDNAC - Excel。

GitHub 。

4 。

-
-
-
-
-

。

Examples

CPython

```
import sys
import json

# load input arguments from the text file
filename = sys.argv[ 1 ]
with open( filename ) as data_file:
    input_args = json.loads( data_file.read() )

# cast strings to floats
x, y = [ float(input_args.get( key )) for key in [ 'x', 'y' ] ]

print json.dumps( { 'sum' : x + y , 'subtract' : x - y } )
```

CPython

```
using MongoDB.Bson;
using System;
using System.Diagnostics;
using System.IO;
```

```
namespace python_csharp
{
    class Program
    {
        static void Main(string[] args)
        {
            // full path to .py file
            string pyScriptPath = "...../sum.py";
            // convert input arguments to JSON string
            BsonDocument argsBson = BsonDocument.Parse("{ 'x' : '1', 'y' : '2' }");

            bool saveInputFile = false;

            string argsFile = string.Format("{0}\\{1}.txt",
Path.GetDirectoryName(pyScriptPath), Guid.NewGuid());

            string outputString = null;
            // create new process start info
            ProcessStartInfo prcStartInfo = new ProcessStartInfo
            {
                // full path of the Python interpreter 'python.exe'
                FileName = "python.exe", // string.Format(@"""{0}""", "python.exe"),
                UseShellExecute = false,
                RedirectStandardOutput = true,
                CreateNoWindow = false
            };

            try
            {
                // write input arguments to .txt file
                using (StreamWriter sw = new StreamWriter(argsFile))
                {
                    sw.WriteLine(argsBson);
                    prcStartInfo.Arguments = string.Format("{0} {1}",
string.Format(@"""{0}""", pyScriptPath), string.Format(@"""{0}""", argsFile));
                }
                // start process
                using (Process process = Process.Start(prcStartInfo))
                {
                    // read standard output JSON string
                    using (StreamReader myStreamReader = process.StandardOutput)
                    {
                        outputString = myStreamReader.ReadLine();
                        process.WaitForExit();
                    }
                }
            }
            finally
            {
                // delete/save temporary .txt file
                if (!saveInputFile)
                {
                    File.Delete(argsFile);
                }
            }
            Console.WriteLine(outputString);
        }
    }
}
```

CPython https://riptutorial.com/zh-TW/python/topic/10759/c-python

122:

python switch。 PEP PEP-3103 。

pythonswitch。

- http://stackoverflow.com/questions/60208/replacements-for-switch-statement-in-python
- http://code.activestate.com/recipes/269708-some-python-style-switches/
- http://code.activestate.com/recipes/410692-readable-switch-construction-without-lambdas-or-di/
- ...

Examples

if / else。

`switch` / `caseif` / `else`

```
def switch(value):
    if value == 1:
        return "one"
    if value == 2:
        return "two"
    if value == 42:
        return "the answer to the question about life, the universe and everything"
    raise Exception("No case found!")
```

```
def switch(value):
    if value == 1:
        return "one"
    if value == 2:
        return "two"
    if value == 42:
        return "the answer to the question about life, the universe and everything"
    raise Exception("No case found!")
```

```
switch = {
    1: lambda: 'one',
    2: lambda: 'two',
    42: lambda: 'the answer of life the universe and everything',
}
```

```
switch = {
    1: lambda: 'one',
    2: lambda: 'two',
    42: lambda: 'the answer of life the universe and everything',
}
```

get。 `default_case` 。

```
switch = {
    1: lambda: 'one',
    2: lambda: 'two',
    42: lambda: 'the answer of life the universe and everything',
}
```

```
switch = {
    1: lambda: 'one',
    2: lambda: 'two',
    42: lambda: 'the answer of life the universe and everything',
}
```

/。 `getattr()`“case”。

`__call__()`。

```
class SwitchBase:
    def switch(self, case):
        m = getattr(self, 'case_{}'.format(case), None)
        if not m:
            return self.default
        return m

    __call__ = switch
```

`SwitchBasecase`

```
class SwitchBase:
    def switch(self, case):
        m = getattr(self, 'case_{}'.format(case), None)
        if not m:
            return self.default
        return m

    __call__ = switch
```

```
class SwitchBase:
    def switch(self, case):
        m = getattr(self, 'case_{}'.format(case), None)
        if not m:
            return self.default
        return m

    __call__ = switch
```

if / elsetrue

```
class Switch:
    def __init__(self, value):
        self._val = value
    def __enter__(self):
        return self
    def __exit__(self, type, value, traceback):
        return False # Allows traceback to occur
    def __call__(self, cond, *mconds):
```

```
        return self._val in (cond,)+mconds
```

`switch` / `case`

```
class Switch:
    def __init__(self, value):
        self._val = value
    def __enter__(self):
        return self
    def __exit__(self, type, value, traceback):
        return False # Allows traceback to occur
    def __call__(self, cond, *mconds):
        return self._val in (cond,)+mconds
```

```
class Switch:
    def __init__(self, value):
        self._val = value
    def __enter__(self):
        return self
    def __exit__(self, type, value, traceback):
        return False # Allows traceback to occur
    def __call__(self, cond, *mconds):
        return self._val in (cond,)+mconds
```

Nota Bene

- pypi 。

https://riptutorial.com/zh-TW/python/topic/4268/

123: CSV

.CSV。 。

“/ path /” “mode”	CSV
“”	
csv.writer	CSV
csv.writerfile **delimiter ='**	

```
open( path, "wb")
```

"wb" - 。

"wb"bWindows。

```
csv.writer ( csv_file, delimiter=',' )
```

,。 ,。

Examples

```
import csv

#------ We will write to CSV in this function ------------

def csv_writer(data, path):

    #Open CSV file whose path we passed.
    with open(path, "wb") as csv_file:

        writer = csv.writer(csv_file, delimiter=',')
        for line in data:
            writer.writerow(line)



#---- Define our list here, and call function ------------

if __name__ == "__main__":

    """
    data = our list that we want to write.
    Split it so we get a list of lists.
    """
    data = ["first_name,last_name,age".split(","),
            "John,Doe,22".split(","),
            "Jane,Doe,31".split(","),
            "Jack,Reacher,27".split(",")
```

```
            ]

    # Path to CSV file we want to write to.
    path = "output.csv"
    csv_writer(data, path)
```

CSV

```
def append_to_csv(input_string):
    with open("fileName.csv", "a") as csv_file:
        csv_file.write(input_row + "\n")
```

CSV https://riptutorial.com/zh-TW/python/topic/10862/csv

124:

。 。 Python。

- <boolean expression>
- <iterable><variable>
- <variable><number>
- <variable><start_number><end_number>
- <variable><start_number><end_number><step_size>
- for i<variable> in enumerate<iterable>with index i
- zip<variable1><variable2><iterable1><iterable2>

	`x < 10`
	`iterable`

Examples

`for`

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

`range`for。

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

python> = 3python <= 2。 for。

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

Python`enumerate`

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

`enumerateindex item` 。

```
for x in ['one', 'two', 'three', 'four']:
```

```
    print(x)
```

maplambdalambda

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

```
for x in ['one', 'two', 'three', 'four']:
    print(x)
```

Python 3.x mapprint(list(x)) http://www.riptutorial.com/python// 8186 / map-- http://www.riptutorial.com/python/topic/809/incompatibilities-moving-from-python-2-to-python-3 。

forlistdict 。

```
for i in [0, 1, 2, 3, 4]:
    print(i)
```

for。

i。 0 1 2 。

```
for i in [0, 1, 2, 3, 4]:
    print(i)
```

rangefor

```
for i in [0, 1, 2, 3, 4]:
    print(i)
```

for。 50。

---

for__getitem____getitem____iter__。 __iter__nextiterable。

---

break

break“”

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

break。 break 。 break。

break4

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

breakforPython

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

34。

else break。

---

continue

continue。 break continue

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

24 continueprint(i)i == 2i == 4 。

---

breakcontinue。 forwhile

```
i = 0
while i < 7:
    print(i)
```

```
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

Python-breakreturn。

returnbreak

return。

returnbreak

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

return

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

```
i = 0
while i < 7:
    print(i)
    if i == 4:
        print("Breaking from loop")
        break
    i += 1
```

“else”

forwhileelse。

elseforwhilefalse while。

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
```

```
else:
    print('done')
```

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

`else break`

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

`else。 else。`

Donald Knuth`ifgotoelse。`

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

`else`。

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

`elsefor`。 True。

`for...else`

```
for i in range(3):
    print(i)
else:
    print('done')

i = 0
while i < 3:
    print(i)
    i += 1
else:
    print('done')
```

`else`“ *if not break* ”“ *if not found* ”。

[Python-ideas] for ... else thread for Summary pythonforwhile'else'

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

Python 2

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

```
d = {"a": 1, "b": 2, "c": 3}
```

Python 2 `.keys()` `.values()``.items()`list。 `.iterkeys()` `.itervalues()`.iteritems() 。

`.keys()`.iterkeys() `.values()`.itervalues() `.items()`.iteritems()iter*。 。 list。

Python 3。

Loop

`while`false 。 4。

```
i = 0
while i < 4:
    #loop statements
    i = i + 1
```

for while。 myObject。

```
i = 0
while i < 4:
    #loop statements
    i = i + 1
```

whileTrue

```
i = 0
while i < 4:
    #loop statements
    i = i + 1
```

whilebreakreturn。

```
i = 0
while i < 4:
    #loop statements
    i = i + 1
```

passnullPythonforwhile。 。

```
for x in range(10):
    pass #we don't want to do anything, or are not ready to do anything here, so we'll pass
```

。 for。 pass。

passwhile。

```
for x in range(10):
    pass #we don't want to do anything, or are not ready to do anything here, so we'll pass
```

。 。 Python。 。

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```

---

for

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```

forlstS。

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```

。 enumerate。

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```

idxs。

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```

Python`range`。

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```


```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```


。 12。 `for`。


```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```

```
lst = ['alpha', 'bravo', 'charlie', 'delta', 'echo']
```


。

“”

Pythondo-untildo-while。 `while True`break。

```
a = 10
while True:
    a = a-1
    print(a)
    if a<7:
        break
print('Done.')
```


```
a = 10
while True:
    a = a-1
    print(a)
    if a<7:
        break
print('Done.')
```


```
collection = [('a', 'b', 'c'), ('x', 'y', 'z'), ('1', '2', '3')]
```


```
collection = [('a', 'b', 'c'), ('x', 'y', 'z'), ('1', '2', '3')]
```

```
collection = [('a', 'b', 'c'), ('x', 'y', 'z'), ('1', '2', '3')]
```

```
collection = [('a', 'b', 'c'), ('x', 'y', 'z'), ('1', '2', '3')]
```


。

https://riptutorial.com/zh-TW/python/topic/237/

125:

Python。 。 //。 Donald Knuth

“97。 3。 “

Examples

。 //。 Donald Knuth

“97。 3”

`cProfile` `profile` `line_profiler` `timeit` 。 。

`cProfile` `cProfile` `cProfile` 。 [https://docs.python.org/2/library/profile.html] [1]

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

。

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

`line_profiler` [https://github.com/rkern/line_profiler] [1]。 。 。 kernprof。

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

kernprof`LineProfiler`profile`__builtins__`。 `@profile`@profile 。

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

kernprof`script_to_profile.py.lprof` 。 kernprof[-v / - view]。

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

`timeit`python shell。 `list()`list()。 `setup-s`。

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

```
import cProfile
def f(x):
    return "42!"
cProfile.run('f(12)')
```

https://riptutorial.com/zh-TW/python/topic/5889/

126:

Examples

dict.get。 。 dict[key]KeyError。

```
def add_student():
    try:
        students['count'] += 1
    except KeyError:
        students['count'] = 1
```

```
def add_student():
    try:
        students['count'] += 1
    except KeyError:
        students['count'] = 1
```

。

```
x = True
y = False
x, y = y, x
x
# False
y
# True
```

Python。

```
# Good examples, using implicit truth testing
if attr:
    # do something

if not attr:
    # do something

# Bad examples, using specific types
if attr == 1:
    # do something

if attr == True:
    # do something

if attr != '':
    # do something

# If you are looking to specifically check for None, use 'is' or 'is not'
if attr is None:
    # do something
```

。

False。

“__main__”

__name__。

```
import sys

def main():
    # Your code starts here

    # Don't forget to provide a return code
    return 0

if __name__ == "__main__":
    sys.exit(main())
```

;

```
import sys

def main():
    # Your code starts here

    # Don't forget to provide a return code
    return 0

if __name__ == "__main__":
    sys.exit(main())
```

import。 import __main__

```
import sys

def main():
    # Your code starts here

    # Don't forget to provide a return code
    return 0

if __name__ == "__main__":
    sys.exit(main())
```

https://riptutorial.com/zh-TW/python/topic/3070/

127:

Examples

Python 3 `print`。

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

`print`

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

`print+`

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

`+`。 `string1"bar"3.14`。

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

print

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

+。

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

endprint

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
```

```
# out: bar
print(baz)
# out: 3.14
```

file

```
print('hello world!')
# out: hello world!

foo = 1
bar = 'bar'
baz = 3.14

print(foo)
# out: 1
print(bar)
# out: bar
print(baz)
# out: 3.14
```

。 print。

sep 。

```
>>> print('apples','bannas', 'cherries', sep=', ')
apple, bannas, cherries
>>> print('apple','banna', 'cherries', sep=', ')
apple, banna, cherries
>>>
```

end end

endprint()。 "。

```
>>> print('apples','bannas', 'cherries', sep=', ')
apple, bannas, cherries
>>> print('apple','banna', 'cherries', sep=', ')
apple, banna, cherries
>>>
```

file sys.stdout。

stdoutStringIO。 。

```
>>> print('apples','bannas', 'cherries', sep=', ')
apple, bannas, cherries
>>> print('apple','banna', 'cherries', sep=', ')
apple, banna, cherries
>>>
```

flush。

https://riptutorial.com/zh-TW/python/topic/1360/

128: abc

Examples

ABCMeta

。

。

NotImplementedError。 。

```
class Fruit:

    def check_ripeness(self):
        raise NotImplementedError("check_ripeness method not implemented!")


class Apple(Fruit):
    pass


a = Apple()
a.check_ripeness() # raises NotImplementedError
```

。 abc

```
class Fruit:

    def check_ripeness(self):
        raise NotImplementedError("check_ripeness method not implemented!")


class Apple(Fruit):
    pass


a = Apple()
a.check_ripeness() # raises NotImplementedError
```

```
class Fruit:

    def check_ripeness(self):
        raise NotImplementedError("check_ripeness method not implemented!")


class Apple(Fruit):
    pass


a = Apple()
a.check_ripeness() # raises NotImplementedError
```

/ABCMeta@abstractmethod

ABCs。

Van Rossum。 Base“MontyPython”。

```
class MontyPython:
    def joke(self):
        raise NotImplementedError()

    def punchline(self):
        raise NotImplementedError()

class ArgumentClinic(MontyPython):
    def joke(self):
        return "Hahahahahah"
```

`punchline()`。

```
class MontyPython:
    def joke(self):
        raise NotImplementedError()

    def punchline(self):
        raise NotImplementedError()

class ArgumentClinic(MontyPython):
    def joke(self):
        return "Hahahahahah"
```

ArgumentClinic。 punchline。

ABC。

```
class MontyPython:
    def joke(self):
        raise NotImplementedError()

    def punchline(self):
        raise NotImplementedError()

class ArgumentClinic(MontyPython):
    def joke(self):
        return "Hahahahahah"
```

TypeError

```
class MontyPython:
    def joke(self):
        raise NotImplementedError()

    def punchline(self):
        raise NotImplementedError()

class ArgumentClinic(MontyPython):
```

```
    def joke(self):
        return "Hahahahahah"
```

TypeErrors

```
class MontyPython:
    def joke(self):
        raise NotImplementedError()

    def punchline(self):
        raise NotImplementedError()

class ArgumentClinic(MontyPython):
    def joke(self):
        return "Hahahahahah"
```

abc https://riptutorial.com/zh-TW/python/topic/5442/-abc-

129:

Examples

python

pythondocstring。

```
#!/usr/local/bin/python3

import ast
import sys

""" The data we collect.  Each key is a function name; each value is a dict
with keys: firstline, sigend, docend, and lastline and values of line numbers
where that happens. """
functions = {}

def process(functions):
    """ Handle the function data stored in functions. """
    for funcname,data in functions.items():
        print("function:",funcname)
        print("\tstarts at line:",data['firstline'])
        print("\tsignature ends at line:",data['sigend'])
        if ( data['sigend'] < data['docend'] ):
            print("\tdocstring ends at line:",data['docend'])
        else:
            print("\tno docstring")
        print("\tfunction ends at line:",data['lastline'])
        print()

class FuncLister(ast.NodeVisitor):
    def visit_FunctionDef(self, node):
        """ Recursively visit all functions, determining where each function
        starts, where its signature ends, where the docstring ends, and where
        the function ends. """
        functions[node.name] = {'firstline':node.lineno}
        sigend = max(node.lineno,lastline(node.args))
        functions[node.name]['sigend'] = sigend
        docstring = ast.get_docstring(node)
        docstringlength = len(docstring.split('\n')) if docstring else -1
        functions[node.name]['docend'] = sigend+docstringlength
        functions[node.name]['lastline'] = lastline(node)
        self.generic_visit(node)

def lastline(node):
    """ Recursively find the last line of a node """
    return max( [ node.lineno if hasattr(node,'lineno') else -1 , ]
                +[lastline(child) for child in ast.iter_child_nodes(node)] )

def readin(pythonfilename):
    """ Read the file name and store the function data into functions. """
    with open(pythonfilename) as f:
        code = f.read()
    FuncLister().visit(ast.parse(code))
```

```
def analyze(file,process):
    """ Read the file and process the function data. """
    readin(file)
    process(functions)

if __name__ == '__main__':
    if len(sys.argv)>1:
        for file in sys.argv[1:]:
            analyze(file,process)
    else:
        analyze(sys.argv[0],process)
```

https://riptutorial.com/zh-TW/python/topic/5370/

130:

。 。 。 Python。

- x << y
- x >> y
- xyAND
- x | yOR
- ~xNOT
- x ^ y #Titwise XOR

Examples

AND

&**AND** 。

```
# 0 & 0 = 0
# 0 & 1 = 0
# 1 & 0 = 0
# 1 & 1 = 1

# 60 = 0b111100
# 30 = 0b011110
60 & 30
# Out: 28
# 28 = 0b11100

bin(60 & 30)
# Out: 0b11100
```

OR

|“”。

```
# 0 | 0 = 0
# 0 | 1 = 1
# 1 | 0 = 1
# 1 | 1 = 1

# 60 = 0b111100
# 30 = 0b011110
60 | 30
# Out: 62
# 62 = 0b111110
```

```
bin(60 | 30)
# Out: 0b111110
```

^**XOR** 11。 1。

```
# 0 ^ 0 = 0
# 0 ^ 1 = 1
# 1 ^ 0 = 1
# 1 ^ 1 = 0

# 60 = 0b111100
# 30 = 0b011110
60 ^ 30
# Out: 34
# 34 = 0b100010

bin(60 ^ 30)
# Out: 0b100010
```

<<“”。

```
# 2 = 0b10
2 << 2
# Out: 8
# 8 = 0b1000

bin(2 << 2)
# Out: 0b1000
```

12

```
# 2 = 0b10
2 << 2
# Out: 8
# 8 = 0b1000

bin(2 << 2)
# Out: 0b1000
```

n2**n

```
# 2 = 0b10
2 << 2
# Out: 8
# 8 = 0b1000

bin(2 << 2)
# Out: 0b1000
```

>>“”。

```
# 8 = 0b1000
8 >> 2
# Out: 2
```

```
# 2 = 0b10

bin(8 >> 2)
# Out: 0b10
```

12

```
# 8 = 0b1000
8 >> 2
# Out: 2
# 2 = 0b10

bin(8 >> 2)
# Out: 0b10
```

n2**n


```
# 8 = 0b1000
8 >> 2
# Out: 2
# 2 = 0b10

bin(8 >> 2)
# Out: 0b10
```


NOT

~。 - 10。

80000 00000111 1111 0127 1xxx xxxx。

0000 0000	0	0
0000 0001	1	1
0000 0010	2	2
0111 1110	126	126
0111 1111	127	127
1000 0000	128	-128
1000 0001	129	-127
1000 0010	130	-126
1111 1110	254	-2
1111 1111	255	-1

1010 0110166(128 * 1) + (64 * 0) + (32 * 1) + (16 * 0) + (8 * 0) + (4 * 1) + (2 * 1) + (1 * 0) -90(128 * 1) - (64 * 0) - (32 * 1) - (16 * 0) - (8 * 0) - (4 * 1) - (2 * 1) - (1 * 0) 。

-128 1000 0000 。 0000 0000 -1 1111 1111 。

~n = -n - 1 。

```
# 0 = 0b0000 0000
~0
# Out: -1
# -1 = 0b1111 1111

# 1 = 0b0000 0001
~1
# Out: -2
# -2 = 1111 1110

# 2 = 0b0000 0010
~2
# Out: -3
# -3 = 0b1111 1101

# 123 = 0b0111 1011
~123
# Out: -124
# -124 = 0b1000 0100
```

```
~n -> -|n+1|

~-n -> |n-1|
```

.....

```
# 0 = 0b0000 0000
~0
# Out: -1
# -1 = 0b1111 1111

# 1 = 0b0000 0001
~1
# Out: -2
# -2 = 1111 1110

# 2 = 0b0000 0010
~2
# Out: -3
# -3 = 0b1111 1101

# 123 = 0b0111 1011
~123
# Out: -124
# -124 = 0b1000 0100
```

Bitwise~

```
a = 0b001
a &= 0b010
```

```
# a = 0b000

a = 0b001
a |= 0b010
# a = 0b011

a = 0b001
a <<= 2
# a = 0b100

a = 0b100
a >>= 2
# a = 0b001

a = 0b101
a ^= 0b011
# a = 0b110
```

https://riptutorial.com/zh-TW/python/topic/730/

131:

Examples

```
min(7,2,1,5)
# Output: 1

max(7,2,1,5)
# Output: 7
```

/

```
list_of_tuples = [(0, 10), (1, 15), (2, 8)]
min(list_of_tuples)
# Output: (0, 10)
```

`key` -argument

```
list_of_tuples = [(0, 10), (1, 15), (2, 8)]
min(list_of_tuples)
# Output: (0, 10)
```

maxmin

`maxmin`

```
min([])
```

ValueErrorminarg

Python 3`default`

```
min([])
```

`sorted` `dict`

```
adict = {'a': 3, 'b': 5, 'c': 1}
min(adict)
# Output: 'a'
max(adict)
# Output: 'c'
sorted(adict)
# Output: ['a', 'b', 'c']
```

`.items()`

```
adict = {'a': 3, 'b': 5, 'c': 1}
min(adict)
```

```
# Output: 'a'
max(adict)
# Output: 'c'
sorted(adict)
# Output: ['a', 'b', 'c']
```

sorted OrderedDictdict

```
adict = {'a': 3, 'b': 5, 'c': 1}
min(adict)
# Output: 'a'
max(adict)
# Output: 'c'
sorted(adict)
# Output: ['a', 'b', 'c']
```

---

key

```
adict = {'a': 3, 'b': 5, 'c': 1}
min(adict)
# Output: 'a'
max(adict)
# Output: 'c'
sorted(adict)
# Output: ['a', 'b', 'c']
```

```
sorted((7, 2, 1, 5))                # tuple
# Output: [1, 2, 5, 7]

sorted(['c', 'A', 'b'])             # list
# Output: ['A', 'b', 'c']

sorted({11, 8, 1})                  # set
# Output: [1, 8, 11]

sorted({'11': 5, '3': 2, '10': 15})  # dict
# Output: ['10', '11', '3']          # only iterates over the keys

sorted('bdca')                      # string
# Output: ['a','b','c','d']
```

list ;。

sorted

```
min([2, 7, 5])
# Output: 2
sorted([2, 7, 5])[0]
# Output: 2
```

sortedorder max。

```
min([2, 7, 5])
```

```
# Output: 2
sorted([2, 7, 5])[0]
# Output: 2
```

```
min([2, 7, 5])
# Output: 2
sorted([2, 7, 5])[0]
# Output: 2
```

<>。

min maxsorted。 6__lt__ __gt__ __ge__ __le__ __ne__ __ge__ __le__ __ne____eq__

```
class IntegerContainer(object):
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return "{}({})".format(self.__class__.__name__, self.value)

    def __lt__(self, other):
        print('{!r} - Test less than {!r}'.format(self, other))
        return self.value < other.value

    def __le__(self, other):
        print('{!r} - Test less than or equal to {!r}'.format(self, other))
        return self.value <= other.value

    def __gt__(self, other):
        print('{!r} - Test greater than {!r}'.format(self, other))
        return self.value > other.value

    def __ge__(self, other):
        print('{!r} - Test greater than or equal to {!r}'.format(self, other))
        return self.value >= other.value

    def __eq__(self, other):
        print('{!r} - Test equal to {!r}'.format(self, other))
        return self.value == other.value

    def __ne__(self, other):
        print('{!r} - Test not equal to {!r}'.format(self, other))
        return self.value != other.value
```

。

```
class IntegerContainer(object):
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return "{}({})".format(self.__class__.__name__, self.value)

    def __lt__(self, other):
        print('{!r} - Test less than {!r}'.format(self, other))
        return self.value < other.value
```

```
    def __le__(self, other):
        print('{!r} - Test less than or equal to {!r}'.format(self, other))
        return self.value <= other.value

    def __gt__(self, other):
        print('{!r} - Test greater than {!r}'.format(self, other))
        return self.value > other.value

    def __ge__(self, other):
        print('{!r} - Test greater than or equal to {!r}'.format(self, other))
        return self.value >= other.value

    def __eq__(self, other):
        print('{!r} - Test equal to {!r}'.format(self, other))
        return self.value == other.value

    def __ne__(self, other):
        print('{!r} - Test not equal to {!r}'.format(self, other))
        return self.value != other.value
```

`reverse=True` `sorted` `__lt__`

```
class IntegerContainer(object):
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return "{}({})".format(self.__class__.__name__, self.value)

    def __lt__(self, other):
        print('{!r} - Test less than {!r}'.format(self, other))
        return self.value < other.value

    def __le__(self, other):
        print('{!r} - Test less than or equal to {!r}'.format(self, other))
        return self.value <= other.value

    def __gt__(self, other):
        print('{!r} - Test greater than {!r}'.format(self, other))
        return self.value > other.value

    def __ge__(self, other):
        print('{!r} - Test greater than or equal to {!r}'.format(self, other))
        return self.value >= other.value

    def __eq__(self, other):
        print('{!r} - Test equal to {!r}'.format(self, other))
        return self.value == other.value

    def __ne__(self, other):
        print('{!r} - Test not equal to {!r}'.format(self, other))
        return self.value != other.value
```

`sorted` `__gt__`

```
class IntegerContainer(object):
    def __init__(self, value):
        self.value = value
```

```
    def __repr__(self):
        return "{}({})".format(self.__class__.__name__, self.value)

    def __lt__(self, other):
        print('{!r} - Test less than {!r}'.format(self, other))
        return self.value < other.value

    def __le__(self, other):
        print('{!r} - Test less than or equal to {!r}'.format(self, other))
        return self.value <= other.value

    def __gt__(self, other):
        print('{!r} - Test greater than {!r}'.format(self, other))
        return self.value > other.value

    def __ge__(self, other):
        print('{!r} - Test greater than or equal to {!r}'.format(self, other))
        return self.value >= other.value

    def __eq__(self, other):
        print('{!r} - Test equal to {!r}'.format(self, other))
        return self.value == other.value

    def __ne__(self, other):
        print('{!r} - Test not equal to {!r}'.format(self, other))
        return self.value != other.value
```

```
__lt____gt____gt__TypeError
```

```
class IntegerContainer(object):
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return "{}({})".format(self.__class__.__name__, self.value)

    def __lt__(self, other):
        print('{!r} - Test less than {!r}'.format(self, other))
        return self.value < other.value

    def __le__(self, other):
        print('{!r} - Test less than or equal to {!r}'.format(self, other))
        return self.value <= other.value

    def __gt__(self, other):
        print('{!r} - Test greater than {!r}'.format(self, other))
        return self.value > other.value

    def __ge__(self, other):
        print('{!r} - Test greater than or equal to {!r}'.format(self, other))
        return self.value >= other.value

    def __eq__(self, other):
        print('{!r} - Test equal to {!r}'.format(self, other))
        return self.value == other.value

    def __ne__(self, other):
        print('{!r} - Test not equal to {!r}'.format(self, other))
        return self.value != other.value
```

TypeErrorunorderableIntegerContainer<IntegerContainer

---

functools.total_ordering。 total_ordering __eq__ __ne__ __lt__ __le__ __ge____gt__

```
class IntegerContainer(object):
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return "{}({})".format(self.__class__.__name__, self.value)

    def __lt__(self, other):
        print('{!r} - Test less than {!r}'.format(self, other))
        return self.value < other.value

    def __le__(self, other):
        print('{!r} - Test less than or equal to {!r}'.format(self, other))
        return self.value <= other.value

    def __gt__(self, other):
        print('{!r} - Test greater than {!r}'.format(self, other))
        return self.value > other.value

    def __ge__(self, other):
        print('{!r} - Test greater than or equal to {!r}'.format(self, other))
        return self.value >= other.value

    def __eq__(self, other):
        print('{!r} - Test equal to {!r}'.format(self, other))
        return self.value == other.value

    def __ne__(self, other):
        print('{!r} - Test not equal to {!r}'.format(self, other))
        return self.value != other.value
```

> __eq__。 。

NN

heapqnlargestnsmallest

```
import heapq

# get 5 largest items from the range

heapq.nlargest(5, range(10))
# Output: [9, 8, 7, 6, 5]

heapq.nsmallest(5, range(10))
# Output: [0, 1, 2, 3, 4]
```

iterable。 。

min maxsortedkey。

1000

```
import heapq

# get 5 largest items from the range

heapq.nlargest(5, range(10))
# Output: [9, 8, 7, 6, 5]

heapq.nsmallest(5, range(10))
# Output: [0, 1, 2, 3, 4]
```

fnlargest。 ;nlargestlen。 len。

1000

```
import heapq

# get 5 largest items from the range

heapq.nlargest(5, range(10))
# Output: [9, 8, 7, 6, 5]

heapq.nsmallest(5, range(10))
# Output: [0, 1, 2, 3, 4]
```

。

https://riptutorial.com/zh-TW/python/topic/252/-

132:

Examples

◦ `__get__()__set__()__get__()`◦ ◦ ◦ ◦

AttributeError**get** ◦ **set** ◦

```
descr.__get__(self, obj, type=None) --> value
descr.__set__(self, obj, value) --> None
descr.__delete__(self, obj) --> None
```


```
descr.__get__(self, obj, type=None) --> value
descr.__set__(self, obj, value) --> None
descr.__delete__(self, obj) --> None
```


◦

◦

```
 >>> oscillator = Oscillator(freq=100.0)  # Set frequency to 100.0 Hz
>>> oscillator.period  # Period is 1 / frequency, i.e. 0.01 seconds
0.01
>>> oscillator.period = 0.02  # Set period to 0.02 seconds
>>> oscillator.freq # The frequency is automatically adjusted
50.0
>>> oscillator.freq = 200.0  # Set the frequency to 200.0 Hz
>>> oscillator.period  # The period is automatically adjusted
0.005
```

“”


```
 >>> oscillator = Oscillator(freq=100.0)  # Set frequency to 100.0 Hz
>>> oscillator.period  # Period is 1 / frequency, i.e. 0.01 seconds
0.01
>>> oscillator.period = 0.02  # Set period to 0.02 seconds
>>> oscillator.freq # The frequency is automatically adjusted
50.0
>>> oscillator.freq = 200.0  # Set the frequency to 200.0 Hz
>>> oscillator.period  # The period is automatically adjusted
0.005
```


“”。

```
 >>> oscillator = Oscillator(freq=100.0)  # Set frequency to 100.0 Hz
>>> oscillator.period  # Period is 1 / frequency, i.e. 0.01 seconds
0.01
>>> oscillator.period = 0.02  # Set period to 0.02 seconds
>>> oscillator.freq # The frequency is automatically adjusted
50.0
>>> oscillator.freq = 200.0  # Set the frequency to 200.0 Hz
```

```
>>> oscillator.period  # The period is automatically adjusted
0.005
```

```
 >>> oscillator = Oscillator(freq=100.0)  # Set frequency to 100.0 Hz
>>> oscillator.period  # Period is 1 / frequency, i.e. 0.01 seconds
0.01
>>> oscillator.period = 0.02  # Set period to 0.02 seconds
>>> oscillator.freq # The frequency is automatically adjusted
50.0
>>> oscillator.freq = 200.0  # Set the frequency to 200.0 Hz
>>> oscillator.period  # The period is automatically adjusted
0.005
```

https://riptutorial.com/zh-TW/python/topic/3405/

133: /

Python。

。

Python。 Exception。 。

Examples

CustomErrorException。 raise。

```
class CustomError(Exception):
       pass

x = 1

if x == 1:
    raise CustomError('This is custom error')
```

```
class CustomError(Exception):
       pass

x = 1

if x == 1:
    raise CustomError('This is custom error')
```

Exception

```
class CustomError(Exception):
     pass

try:
    raise CustomError('Can you catch me ?')
except CustomError as e:
    print ('Catched CustomError :{}'.format(e))
except Exception as e:
    print ('Generic exception: {}'.format(e))
```

```
class CustomError(Exception):
     pass

try:
    raise CustomError('Can you catch me ?')
except CustomError as e:
    print ('Catched CustomError :{}'.format(e))
except Exception as e:
    print ('Generic exception: {}'.format(e))
```

/ https://riptutorial.com/zh-TW/python/topic/10882/-

134:

Examples

mixin。 mixin。

。 Mixins。 mixin。

mixinsmixin。

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
        print "Mixin2"

class MyClass(Mixin1, Mixin2):
    pass
```

MyClasstest

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
        print "Mixin2"

class MyClass(Mixin1, Mixin2):
    pass
```

Mixin1。 。

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
        print "Mixin2"

class MyClass(Mixin1, Mixin2):
    pass
```

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
```

```
        print "Mixin2"

class MyClass(Mixin1, Mixin2):
    pass
```

Mixins。

Python 3.x 3.0

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
        print "Mixin2"

class MyClass(Mixin1, Mixin2):
    pass
```

Python 3.6 PEP 487 `__init_subclass__`。 。

Python 3.x 3.6

```
class Base:
    plugins = []

    def __init_subclass__(cls, **kwargs):
        super().__init_subclass__(**kwargs)
        cls.plugins.append(cls)

    def test(self):
        print("Base.")

class PluginA(Base):
    def test(self):
        super().test()
        print("Plugin A.")


class PluginB(Base):
    def test(self):
        super().test()
        print("Plugin B.")
```

```
class Base:
    plugins = []

    def __init_subclass__(cls, **kwargs):
        super().__init_subclass__(**kwargs)
        cls.plugins.append(cls)

    def test(self):
        print("Base.")

class PluginA(Base):
    def test(self):
```

```
        super().test()
        print("Plugin A.")


class PluginB(Base):
    def test(self):
        super().test()
        print("Plugin B.")
```

https://riptutorial.com/zh-TW/python/topic/4724/

135:

nO(n)◦ bisect.bisect_left()O(log(n))◦

Examples

str.indexstr.rindexstr.findstr.rfind

Stringindexstr.find◦ ◦

```
astring = 'Hello on StackOverflow'
astring.index('o')  # 4
astring.rindex('o') # 20

astring.find('o')   # 4
astring.rfind('o')  # 20
```

index / rindexfind / rfind

```
astring = 'Hello on StackOverflow'
astring.index('o')  # 4
astring.rindex('o') # 20

astring.find('o')   # 4
astring.rfind('o')  # 20
```

```
astring = 'Hello on StackOverflow'
astring.index('o')  # 4
astring.rindex('o') # 20

astring.find('o')   # 4
astring.rfind('o')  # 20
```

ValueError

```
astring = 'Hello on StackOverflow'
astring.index('o')  # 4
astring.rindex('o') # 20

astring.find('o')   # 4
astring.rfind('o')  # 20
```

Python in◦

```
alist = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
5 in alist   # True
10 in alist  # False
```

```
alist = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
5 in alist   # True
10 in alist  # False
```

```
alist = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
5 in alist   # True
10 in alist  # False
```

```
alist = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
5 in alist   # True
10 in alist  # False
```

dictin ◦ ◦ ◦

```
alist = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
5 in alist   # True
10 in alist  # False
```

list.indextuple.index

listtupleindex -method

```
alist = [10, 16, 26, 5, 2, 19, 105, 26]
# search for 16 in the list
alist.index(16) # 1
alist[1]        # 16

alist.index(15)
```

ValueError15

```
alist = [10, 16, 26, 5, 2, 19, 105, 26]
# search for 16 in the list
alist.index(16) # 1
alist[1]        # 16

alist.index(15)
```

dict

dict◦

```
def getKeysForValue(dictionary, value):
    foundkeys = []
    for keys in dictionary:
        if dictionary[key] == value:
            foundkeys.append(key)
    return foundkeys
```

```
def getKeysForValue(dictionary, value):
    foundkeys = []
```

```
    for keys in dictionary:
        if dictionary[key] == value:
            foundkeys.append(key)
    return foundkeys
```

```
def getKeysForValue(dictionary, value):
    foundkeys = []
    for keys in dictionary:
        if dictionary[key] == value:
            foundkeys.append(key)
    return foundkeys
```

`keyslist`

```
def getKeysForValue(dictionary, value):
    foundkeys = []
    for keys in dictionary:
        if dictionary[key] == value:
            foundkeys.append(key)
    return foundkeys
```

```
def getKeysForValue(dictionary, value):
    foundkeys = []
    for keys in dictionary:
        if dictionary[key] == value:
            foundkeys.append(key)
    return foundkeys
```

`dict StopIteration - Exception`

```
def getKeysForValue(dictionary, value):
    foundkeys = []
    for keys in dictionary:
        if dictionary[key] == value:
            foundkeys.append(key)
    return foundkeys
```

StopIteration

bisect.bisect_left

`bisect.bisect_left()` [1]

```
import bisect

def index_sorted(sorted_seq, value):
    """Locate the leftmost value exactly equal to x or raise a ValueError"""
    i = bisect.bisect_left(sorted_seq, value)
    if i != len(sorted_seq) and sorted_seq[i] == value:
        return i
    raise ValueError

alist = [i for i in range(1, 100000, 3)] # Sorted list from 1 to 100000 with step 3
index_sorted(alist, 97285) # 32428
```

```
index_sorted(alist, 4)     # 1
index_sorted(alist, 97286)
```

ValueError

- 500

```
import bisect

def index_sorted(sorted_seq, value):
    """Locate the leftmost value exactly equal to x or raise a ValueError"""
    i = bisect.bisect_left(sorted_seq, value)
    if i != len(sorted_seq) and sorted_seq[i] == value:
        return i
    raise ValueError

alist = [i for i in range(1, 100000, 3)] # Sorted list from 1 to 100000 with step 3
index_sorted(alist, 97285) # 32428
index_sorted(alist, 4)     # 1
index_sorted(alist, 97286)
```

```
import bisect

def index_sorted(sorted_seq, value):
    """Locate the leftmost value exactly equal to x or raise a ValueError"""
    i = bisect.bisect_left(sorted_seq, value)
    if i != len(sorted_seq) and sorted_seq[i] == value:
        return i
    raise ValueError

alist = [i for i in range(1, 100000, 3)] # Sorted list from 1 to 100000 with step 3
index_sorted(alist, 97285) # 32428
index_sorted(alist, 4)     # 1
index_sorted(alist, 97286)
```

tuple list dict。

```
def outer_index(nested_sequence, value):
    return next(index for index, inner in enumerate(nested_sequence)
                      for item in inner
                      if item == value)

alist_of_tuples = [(4, 5, 6), (3, 1, 'a'), (7, 0, 4.3)]
outer_index(alist_of_tuples, 'a')  # 1
outer_index(alist_of_tuples, 4.3)  # 2
```

```
def outer_index(nested_sequence, value):
    return next(index for index, inner in enumerate(nested_sequence)
                      for item in inner
                      if item == value)

alist_of_tuples = [(4, 5, 6), (3, 1, 'a'), (7, 0, 4.3)]
outer_index(alist_of_tuples, 'a')  # 1
outer_index(alist_of_tuples, 4.3)  # 2
```

next

。

__ content____iter__

in for__contains__ __iter__ -method。

list list

```
class ListList:
    def __init__(self, value):
        self.value = value
        # Create a set of all values for fast access
        self.setofvalues = set(item for sublist in self.value for item in sublist)

    def __iter__(self):
        print('Using __iter__.')
        # A generator over all sublist elements
        return (item for sublist in self.value for item in sublist)

    def __contains__(self, value):
        print('Using __contains__.')
        # Just lookup if the value is in the set
        return value in self.setofvalues

        # Even without the set you could use the iter method for the contains-check:
        # return any(item == value for item in iter(self))
```

in

```
class ListList:
    def __init__(self, value):
        self.value = value
        # Create a set of all values for fast access
        self.setofvalues = set(item for sublist in self.value for item in sublist)

    def __iter__(self):
        print('Using __iter__.')
        # A generator over all sublist elements
        return (item for sublist in self.value for item in sublist)

    def __contains__(self, value):
        print('Using __contains__.')
        # Just lookup if the value is in the set
        return value in self.setofvalues

        # Even without the set you could use the iter method for the contains-check:
        # return any(item == value for item in iter(self))
```

__contains__

```
class ListList:
    def __init__(self, value):
        self.value = value
        # Create a set of all values for fast access
        self.setofvalues = set(item for sublist in self.value for item in sublist)

    def __iter__(self):
```

```
        print('Using __iter__.')
        # A generator over all sublist elements
        return (item for sublist in self.value for item in sublist)

    def __contains__(self, value):
        print('Using __contains__.')
        # Just lookup if the value is in the set
        return value in self.setofvalues

        # Even without the set you could use the iter method for the contains-check:
        # return any(item == value for item in iter(self))
```

in for i in a __iter____contains__。

https://riptutorial.com/zh-TW/python/topic/350/

136:

Examples

`+operator` -function `operator.add` for `+`

```
1 + 1
# Output: 2
from operator import add
add(1, 1)
# Output: 2
```


```
1 + 1
# Output: 2
from operator import add
add(1, 1)
# Output: 2
```


Python 。

Methodcaller

`lambda`

```
alist = ['wolf', 'sheep', 'duck']
list(filter(lambda x: x.startswith('d'), alist))     # Keep only elements that start with 'd'
# Output: ['duck']
```


```
alist = ['wolf', 'sheep', 'duck']
list(filter(lambda x: x.startswith('d'), alist))     # Keep only elements that start with 'd'
# Output: ['duck']
```


Itemgetter

`itemgetter`

```
from itertools import groupby
from operator import itemgetter
adict = {'a': 1, 'b': 5, 'c': 1}

dict((i, dict(v)) for i, v in groupby(adict.items(), itemgetter(1)))
# Output: {1: {'a': 1, 'c': 1}, 5: {'b': 5}}
```

`lambda`

```
from itertools import groupby
from operator import itemgetter
adict = {'a': 1, 'b': 5, 'c': 1}

dict((i, dict(v)) for i, v in groupby(adict.items(), itemgetter(1)))
```

```
# Output: {1: {'a': 1, 'c': 1}, 5: {'b': 5}}
```

```
from itertools import groupby
from operator import itemgetter
adict = {'a': 1, 'b': 5, 'c': 1}

dict((i, dict(v)) for i, v in groupby(adict.items(), itemgetter(1)))
# Output: {1: {'a': 1, 'c': 1}, 5: {'b': 5}}
```

https://riptutorial.com/zh-TW/python/topic/257/

137: XML

XML。XML。APIXML;XML。

Examples

ElementTree

ElementTree.xml

```
import xml.etree.ElementTree as ET
tree = ET.parse("yourXMLfile.xml")
root = tree.getroot()
```

。

```
import xml.etree.ElementTree as ET
tree = ET.parse("yourXMLfile.xml")
root = tree.getroot()
```

```
import xml.etree.ElementTree as ET
tree = ET.parse("yourXMLfile.xml")
root = tree.getroot()
```

`.find.findall`

```
import xml.etree.ElementTree as ET
tree = ET.parse("yourXMLfile.xml")
root = tree.getroot()
```

XML

xmlxml

```
import xml.etree.ElementTree as ET
tree = ET.parse('sample.xml')
root=tree.getroot()
element = root[0] #get first child of root element
```

```
import xml.etree.ElementTree as ET
tree = ET.parse('sample.xml')
root=tree.getroot()
element = root[0] #get first child of root element
```

Element.remove

```
import xml.etree.ElementTree as ET
```

```
tree = ET.parse('sample.xml')
root=tree.getroot()
element = root[0] #get first child of root element
```

ElementTree.writexmlxml。

```
import xml.etree.ElementTree as ET
tree = ET.parse('sample.xml')
root=tree.getroot()
element = root[0] #get first child of root element
```

XML

```
import xml.etree.ElementTree as ET
```

ElementXML

```
import xml.etree.ElementTree as ET
```

SubElementgive

```
import xml.etree.ElementTree as ET
```

dumpxml。

```
import xml.etree.ElementTree as ET
```

ElementTreexmlwrite

```
import xml.etree.ElementTree as ET
```

Commentxml。

```
import xml.etree.ElementTree as ET
```

iterparseXML

XML。 。 iterparseXML。

ElementTree

```
import xml.etree.ElementTree as ET
```

.xml

```
import xml.etree.ElementTree as ET
```

/。 “end”

```
import xml.etree.ElementTree as ET
```


```
import xml.etree.ElementTree as ET
```


XPathXML

2.7 `ElementTree`XPath。 XPathxmlSQL。 `findfindall`XPath。 xml

```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
    </Books>
</Catalog>
```


```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
    </Books>
</Catalog>
```


'''

```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
```

```
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
    </Books>
</Catalog>
```

id = 5

```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
    </Books>
</Catalog>
```

```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
    </Books>
</Catalog>
```

```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
```

```
    </Books>
</Catalog>
```

```
 <Catalog>
    <Books>
        <Book id="1" price="7.95">
            <Title>Do Androids Dream of Electric Sheep?</Title>
            <Author>Philip K. Dick</Author>
        </Book>
        <Book id="5" price="5.95">
            <Title>The Colour of Magic</Title>
            <Author>Terry Pratchett</Author>
        </Book>
        <Book id="7" price="6.95">
            <Title>The Eye of The World</Title>
            <Author>Robert Jordan</Author>
        </Book>
    </Books>
</Catalog>
```

XML https://riptutorial.com/zh-TW/python/topic/479/xml

138:

Shelvepython。 shelveAPIPythonpickle。 Python。 。 pickleanydbm。

;`close() shelve.open()`

```
with shelve.open('spam') as db:
    db['eggs'] = 'eggs'
```

`shelvepickle`。 pickle。

1 。 dbm.ndbmdbm.gnu。 dbm。 dbm - 。

2./。 。 。 UnixUnix。

Examples

```
import shelve
 database = shelve.open(filename.suffix)
 object = Object()
 database['key'] = object
```

```
import shelve

d = shelve.open(filename)  # open -- file may get suffix added by low-level
                           # library

d[key] = data              # store data at key (overwrites old data if
                           # using an existing key)
data = d[key]              # retrieve a COPY of data at key (raise KeyError
                           # if no such key)
del d[key]                 # delete data stored at key (raises KeyError
                           # if no such key)

flag = key in d            # true if the key exists
klist = list(d.keys())     # a list of all existing keys (slow!)

# as d was opened WITHOUT writeback=True, beware:
d['xx'] = [0, 1, 2]        # this works as expected, but...
d['xx'].append(3)          # *this doesn't!* -- d['xx'] is STILL [0, 1, 2]!

# having opened d without writeback=True, you need to code carefully:
temp = d['xx']             # extracts the copy
temp.append(5)             # mutates the copy
d['xx'] = temp             # stores the copy right back, to persist it

# or, d=shelve.open(filename,writeback=True) would let you just code
# d['xx'].append(5) and have it work as expected, BUT it would also
# consume more memory and make the d.close() operation slower.

d.close()                  # close it
```

DbfilenameShelf。anydbm。**shelve.open**

```
import shelve

s = shelve.open('test_shelf.db')
try:
    s['key1'] = { 'int': 10, 'float':9.5, 'string':'Sample data' }
finally:
    s.close()
```

```
import shelve

s = shelve.open('test_shelf.db')
try:
    s['key1'] = { 'int': 10, 'float':9.5, 'string':'Sample data' }
finally:
    s.close()
```

```
import shelve

s = shelve.open('test_shelf.db')
try:
    s['key1'] = { 'int': 10, 'float':9.5, 'string':'Sample data' }
finally:
    s.close()
```

dbm。shelve。

```
import shelve

s = shelve.open('test_shelf.db')
try:
    s['key1'] = { 'int': 10, 'float':9.5, 'string':'Sample data' }
finally:
    s.close()
```

。anydbm。

。 。

```
import shelve

s = shelve.open('test_shelf.db')
try:
    print s['key1']
    s['key1']['new_value'] = 'this was not here before'
finally:
    s.close()

s = shelve.open('test_shelf.db', writeback=True)
try:
    print s['key1']
finally:
    s.close()
```

'key1'。

```
import shelve

s = shelve.open('test_shelf.db')
try:
    print s['key1']
    s['key1']['new_value'] = 'this was not here before'
finally:
    s.close()

s = shelve.open('test_shelf.db', writeback=True)
try:
    print s['key1']
finally:
    s.close()
```

。 。 。

```
import shelve

s = shelve.open('test_shelf.db')
try:
    print s['key1']
    s['key1']['new_value'] = 'this was not here before'
finally:
    s.close()

s = shelve.open('test_shelf.db', writeback=True)
try:
    print s['key1']
finally:
    s.close()
```

。 。 。 。

```
import shelve

s = shelve.open('test_shelf.db')
try:
    print s['key1']
    s['key1']['new_value'] = 'this was not here before'
finally:
    s.close()

s = shelve.open('test_shelf.db', writeback=True)
try:
    print s['key1']
finally:
    s.close()
```

https://riptutorial.com/zh-TW/python/topic/10629/

139:

`collectionsdict list tupleset◦ MutableSetItemsView ◦`

1. UserDict
2. UserList
3. UserString

UserDictdict◦ ◦ data◦ ;◦

Examples

collections.Counter

Counterdict◦ ◦

```
import collections
counts = collections.Counter([1,2,3])
```

count◦ `Counter({1: 1, 2: 1, 3: 1})`

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

4 + 2 + 0 - 3

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

0

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

```
import collections
counts = collections.Counter([1,2,3])
```

collections.defaultdict

collections.defaultdict default_factory`dict`。 。 `None` 。

```
>>> state_capitals = collections.defaultdict(str)
>>> state_capitals
defaultdict(<class 'str'>, {})
```

defaultdictdefaultdictdefault_factory。

`defaultdictstr int listdict`default_factory

```
>>> state_capitals = collections.defaultdict(str)
>>> state_capitals
defaultdict(<class 'str'>, {})
```

defaultdict。

```
>>> state_capitals = collections.defaultdict(str)
>>> state_capitals
defaultdict(<class 'str'>, {})
```

`int`

```
>>> state_capitals = collections.defaultdict(str)
>>> state_capitals
defaultdict(<class 'str'>, {})
```

```
>>> state_capitals = collections.defaultdict(str)
>>> state_capitals
defaultdict(<class 'str'>, {})
```

`list`default_factory。

```
>>> state_capitals = collections.defaultdict(str)
>>> state_capitals
defaultdict(<class 'str'>, {})
```

collections.OrderedDict

Python。

```
>>> d = {'foo': 5, 'bar': 6}
>>> print(d)
{'foo': 5, 'bar': 6}
>>> d['baz'] = 7
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6}
>>> d['foobar'] = 8
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6, 'foobar': 8}
```

。

for。

collections.OrderedDict。 OrderedDict

```
>>> d = {'foo': 5, 'bar': 6}
>>> print(d)
{'foo': 5, 'bar': 6}
>>> d['baz'] = 7
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6}
>>> d['foobar'] = 8
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6, 'foobar': 8}
```

OrderedDict

```
>>> d = {'foo': 5, 'bar': 6}
>>> print(d)
{'foo': 5, 'bar': 6}
>>> d['baz'] = 7
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6}
>>> d['foobar'] = 8
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6, 'foobar': 8}
```

OrderedDict。

```
>>> d = {'foo': 5, 'bar': 6}
>>> print(d)
{'foo': 5, 'bar': 6}
>>> d['baz'] = 7
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6}
>>> d['foobar'] = 8
>>> print(a)
{'baz': 7, 'foo': 5, 'bar': 6, 'foobar': 8}
```

OrderedDict。

collections.namedtuple

namedtuplePerson

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

。

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

namedtuple

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

namedtuple'Person' typename 。 typename

```
Person = namedtuple('Person', ['age', 'height', 'name'])
```

collections.deque

dequeappenditerable。 iterabledeque。

Deques“deck”“”。 Dequesdequeo1。

pop0insert0vOn。

2.4。

maxlenNone deques。 deque。 deque。 dequesUnix。 。

2.6maxlen。

```
>>> from collections import deque
>>> d = deque('ghi')                 # make a new deque with three items
>>> for elem in d:                   # iterate over the deque's elements
...     print elem.upper()
G
H
I
```

```
>>> d.append('j')                     # add a new entry to the right side
>>> d.appendleft('f')                 # add a new entry to the left side
>>> d                                 # show the representation of the deque
deque(['f', 'g', 'h', 'i', 'j'])

>>> d.pop()                           # return and remove the rightmost item
'j'
>>> d.popleft()                       # return and remove the leftmost item
'f'
>>> list(d)                           # list the contents of the deque
['g', 'h', 'i']
>>> d[0]                              # peek at leftmost item
'g'
>>> d[-1]                             # peek at rightmost item
'i'

>>> list(reversed(d))                 # list the contents of a deque in reverse
['i', 'h', 'g']
>>> 'h' in d                          # search the deque
True
>>> d.extend('jkl')                   # add multiple elements at once
>>> d
deque(['g', 'h', 'i', 'j', 'k', 'l'])
>>> d.rotate(1)                       # right rotation
>>> d
deque(['l', 'g', 'h', 'i', 'j', 'k'])
>>> d.rotate(-1)                      # left rotation
>>> d
deque(['g', 'h', 'i', 'j', 'k', 'l'])

>>> deque(reversed(d))                # make a new deque in reverse order
deque(['l', 'k', 'j', 'i', 'h', 'g'])
>>> d.clear()                         # empty the deque
>>> d.pop()                           # cannot pop from an empty deque
Traceback (most recent call last):
  File "<pyshell#6>", line 1, in -toplevel-
    d.pop()
IndexError: pop from an empty deque

>>> d.extendleft('abc')               # extendleft() reverses the input order
>>> d
deque(['c', 'b', 'a'])
```

https //docs.python.org/2/library/collections.html

collections.ChainMap

ChainMap**3.3**

mapsChainMap。 dicts。

ChainMap。 pythonDjangoContext。 。 update()。

ChainMap。 。 webPOSTGETDjangoFlask。 ChainMap。

maps。 。 。

```
import collections

# define two dictionaries with at least some keys overlapping.
dict1 = {'apple': 1, 'banana': 2}
dict2 = {'coconut': 1, 'date': 1, 'apple': 3}

# create two ChainMaps with different ordering of those dicts.
combined_dict = collections.ChainMap(dict1, dict2)
reverse_ordered_dict = collections.ChainMap(dict2, dict1)
```


```
import collections

# define two dictionaries with at least some keys overlapping.
dict1 = {'apple': 1, 'banana': 2}
dict2 = {'coconut': 1, 'date': 1, 'apple': 3}

# create two ChainMaps with different ordering of those dicts.
combined_dict = collections.ChainMap(dict1, dict2)
reverse_ordered_dict = collections.ChainMap(dict2, dict1)
```


https://riptutorial.com/zh-TW/python/topic/498/

140:

Examples

ceiltrunc

round mathfloor ceiltrunc◦

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)        #  2
round(y)        # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)     #  1.6
round(y, 1)     # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)   #  1
math.floor(y)   # -2

# get the smallest integer greater than x
math.ceil(x)    #  2
math.ceil(y)    # -1

# drop fractional part of x
math.trunc(x)   #  1, equivalent to math.floor for positive numbers
math.trunc(y)   # -1, equivalent to math.ceil for negative numbers
```

Python 2.x 2.7

floor ceil truncroundfloat ◦

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)        #  2
round(y)        # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)     #  1.6
round(y, 1)     # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)   #  1
math.floor(y)   # -2
```

```
# get the smallest integer greater than x
math.ceil(x)   #  2
math.ceil(y)   # -1

# drop fractional part of x
math.trunc(x)  #  1, equivalent to math.floor for positive numbers
math.trunc(y)  # -1, equivalent to math.ceil for negative numbers
```

round。

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)       #  2
round(y)       # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)    #  1.6
round(y, 1)    # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)  #  1
math.floor(y)  # -2

# get the smallest integer greater than x
math.ceil(x)   #  2
math.ceil(y)   # -1

# drop fractional part of x
math.trunc(x)  #  1, equivalent to math.floor for positive numbers
math.trunc(y)  # -1, equivalent to math.ceil for negative numbers
```

Python 3.x 3.0

floor ceiltruncIntegralroundIntegral。

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)       #  2
round(y)       # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)    #  1.6
round(y, 1)    # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)  #  1
math.floor(y)  # -2
```

```
# get the smallest integer greater than x
math.ceil(x)   #  2
math.ceil(y)   # -1

# drop fractional part of x
math.trunc(x)  #  1, equivalent to math.floor for positive numbers
math.trunc(y)  # -1, equivalent to math.ceil for negative numbers
```

round◦ ◦

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)       #  2
round(y)       # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)    #  1.6
round(y, 1)    # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)  #  1
math.floor(y)  # -2

# get the smallest integer greater than x
math.ceil(x)   #  2
math.ceil(y)   # -1

# drop fractional part of x
math.trunc(x)  #  1, equivalent to math.floor for positive numbers
math.trunc(y)  # -1, equivalent to math.ceil for negative numbers
```

◦ ◦

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)       #  2
round(y)       # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)    #  1.6
round(y, 1)    # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)  #  1
math.floor(y)  # -2

# get the smallest integer greater than x
```

```
math.ceil(x)   #  2
math.ceil(y)   # -1

# drop fractional part of x
math.trunc(x)  #  1, equivalent to math.floor for positive numbers
math.trunc(y)  # -1, equivalent to math.ceil for negative numbers
```

PythonC ++Java。

```
x = 1.55
y = -1.55

# round to the nearest integer
round(x)       #  2
round(y)       # -2

# the second argument gives how many decimal places to round to (defaults to 0)
round(x, 1)    #  1.6
round(y, 1)    # -1.6

# math is a module so import it first, then use it.
import math

# get the largest integer less than x
math.floor(x)  #  1
math.floor(y)  # -2

# get the smallest integer greater than x
math.ceil(x)   #  2
math.ceil(y)   # -1

# drop fractional part of x
math.trunc(x)  #  1, equivalent to math.floor for positive numbers
math.trunc(y)  # -1, equivalent to math.ceil for negative numbers
```

`math.log(x)`ex 。

```
math.log(math.e)  # 1.0
math.log(1)       # 0.0
math.log(100)     # 4.605170185988092
```

`math.log`1。 1`math.log1p` 1

```
math.log(math.e)  # 1.0
math.log(1)       # 0.0
math.log(100)     # 4.605170185988092
```

`math.log10`10

```
math.log(math.e)  # 1.0
math.log(1)       # 0.0
math.log(100)     # 4.605170185988092
```

Python 2.x 2.3.0

math.log(x, base) math.log(x, base)basexlog(x) / log(base) ◦

```
math.log(math.e)  # 1.0
math.log(1)       # 0.0
math.log(100)     # 4.605170185988092
```

Python 2.6 math.copysign(x, y)x y ◦ float ◦

Python 2.x 2.6

```
math.copysign(-2, 3)    # 2.0
math.copysign(3, -3)    # -3.0
math.copysign(4, 14.2)  # 4.0
math.copysign(1, -0.0)  # -1.0, on a platform which supports signed zero
```

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

/

math

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

Python 3.x 3.5

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

math.atanmath.atan2

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

```
math.hypot(2, 4) # Just a shorthand for SquareRoot(2**2 + 4**2)
# Out: 4.47213595499958
```

math◦

- `math.pi` - pi
- `math.e` - *e*

```
>>> from math import pi, e
>>> pi
3.141592653589793
>>> e
2.718281828459045
>>>
```

Python 3.5NaN“”。 `float()`。

Python 3.x 3.5

```
>>> from math import pi, e
>>> pi
3.141592653589793
>>> e
2.718281828459045
>>>
```

Python“j”“J”。

```
1j         # Equivalent to the square root of -1.
1j * 1j    # = (-1+0j)
```

NaN“”

PythonNaN“”

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

Python 3.5`math.inf``math.nan`

Python 3.x 3.5

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

`inf-inf``nan`

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

`isinf`

```
pos_inf = float('inf')     # positive infinity
```

```
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

Python 3.2

Python 3.x 3.2

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

`float`

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

`ZeroDivisionError`。

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

NaN

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

NaN。 `isnan`

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

NaN“”

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

NaNNaN。 -1“NaN”。

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

Python 3.x 3.5

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

NaNInfinity`float`Python 3.5+ `math`

Python 3.x 3.5

```
pos_inf = float('inf')     # positive infinity
neg_inf = float('-inf')    # negative infinity
not_a_num = float('nan')   # NaN ("not a number")
```

timeit

```
> python -m timeit 'for x in xrange(50000): b = x**3'
10 loops, best of 3: 51.2 msec per loop
> python -m timeit 'from math import pow' 'for x in xrange(50000): b = pow(x,3)'
100 loops, best of 3: 9.15 msec per loop
```

`**`math.pow。 pow

```
> python -m timeit 'for x in xrange(50000): b = x**3'
10 loops, best of 3: 51.2 msec per loop
> python -m timeit 'from math import pow' 'for x in xrange(50000): b = pow(x,3)'
100 loops, best of 3: 9.15 msec per loop
```

cmath

`cmathmath`。

Python。 `import cmath` 。

`j` `J` `i` 。

```
z = 1 + 3j
```

`1jj`。

```
z = 1 + 3j
```

`real` `imag` `conjugate`

```
z = 1 + 3j
```

`abscomplex`

```
z = 1 + 3j
```

`complex`

```
z = 1 + 3j
```

`sqrt`

```
z = 1 + 3j
```

`sqrt。 math`

```
z = 1 + 3j
```

```
z = 1 + 3j
```

“”。 IEEE 754“”。 Python

```
z = 1 + 3j
```

`cmathmath。`

`sqrt exp log log10 sin cos tan asin acos atan sinh cosh tanh asinh acosh atanh 。 math.atan2` arctangent。

```
z = 1 + 3j
```

`pie 。 floatcomplex 。`

```
z = 1 + 3j
```

`cmathisinf`Python 3.2+`isfinite 。` “ InfinityNaN ”。 。

```
z = 1 + 3j
```

`cmathisnan。` “ InfinityNaN ”。 “”“”。

```
z = 1 + 3j
```

`math.infmath.nancmath`Python 3.5

Python 3.x 3.5

```
z = 1 + 3j
```

Python 3.5 `cmathmathisclose`。

Python 3.x 3.5

```
z = 1 + 3j
```

https://riptutorial.com/zh-TW/python/topic/230/

141:

- unpickled_string = pickle.loadsstring
- unpickled_string = pickle.loadfile_object
- pickled_string = pickle.dumps["'cmplx'{'object'None}]pickle.HIGHEST_PROTOCOL
- pickle.dump"'cmplx'{'object'None}]file_objectpickle.HIGHEST_PROTOCOL
- unjsoned_string = json.loadsstring
- unjsoned_string = json.loadfile_object
- jsoned_string = json.dumps'a"b"c'[1,2,3]
- json.dump'a"b"c'[1,2,3]file_object

`protocol`	`picklecPickle` /。 `pickle.HIGHEST_PROTOCOL` 。

JSON

-
-
- pickle

JSON

- Pythonic
- 。

- Pythonic
- 。

Pickle

-
-

Examples

JSON

JSON

int float boolean string listdict 。 - > JSON Wiki

JSON -

```
import json

families = (['John'], ['Mark', 'David', {'name': 'Avraham'}])
```

```
# Dumping it into string
json_families = json.dumps(families)
# [["John"], ["Mark", "David", {"name": "Avraham"}]]

# Dumping it to file
with open('families.json', 'w') as json_file:
    json.dump(families, json_file)

# Loading it from string
json_families = json.loads(json_families)

# Loading it from file
with open('families.json', 'r') as json_file:
    json_families = json.load(json_file)
```

JSONJSON-Module 。

Pickle

-

```
# Importing pickle
try:
    import cPickle as pickle  # Python 2
except ImportError:
    import pickle  # Python 3

# Creating Pythonic object:
class Family(object):
    def __init__(self, names):
        self.sons = names

    def __str__(self):
        return ' '.join(self.sons)

my_family = Family(['John', 'David'])

# Dumping to string
pickle_data = pickle.dumps(my_family, pickle.HIGHEST_PROTOCOL)

# Dumping to file
with open('family.p', 'w') as pickle_file:
    pickle.dump(families, pickle_file, pickle.HIGHEST_PROTOCOL)

# Loading from string
my_family = pickle.loads(pickle_data)

# Loading from file
with open('family.p', 'r') as pickle_file:
    my_family = pickle.load(pickle_file)
```

PicklePickle。

pickle。 。

https://riptutorial.com/zh-TW/python/topic/3347/

142:

Python。API。APIPEP 249。

APIPython。Python。 PEP-249

Examples

MySQLdbMySQL

connect。 。

executecommit。

。

Dbconnect。

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):

        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()

    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

。 execute。

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):

        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()
```

```
    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

- -

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):

        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()

    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

- -

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):

        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()

    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):
```

```
        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()

    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):

        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()

    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

```
import MySQLdb

class Dbconnect(object):

    def __init__(self):

        self.dbconection = MySQLdb.connect(host='host_example',
                                           port=int('port_example'),
                                           user='user_example',
                                           passwd='pass_example',
                                           db='schema_example')
        self.dbcursor = self.dbconection.cursor()

    def commit_db(self):
        self.dbconection.commit()

    def close_db(self):
        self.dbcursor.close()
        self.dbconection.close()
```

SQLite

SQLite。 。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

`users.db`。 SQL。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

SQLite

1. sqlite

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())
```

```
conn.close()
```

2. Connection。 example.db

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

`:memory:`RAM

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

3. `Connection` `Cursor``execute()`SQL

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())
```

```
conn.close()
```

Connection

1. `isolation_level`

   - `DEFERRED` `IMMEDIATE` `EXCLUSIVE` ◦

2. `cursor`

   SQL◦

3. `commit()`

   ◦

4. `rollback()`

   `commit()`

5. `close()`

   - `commit()` ◦ `commit()` `close()` ◦

6. `total_changes`

   ◦

7. `execute` `executemany` `executescript`

   ◦

8. `row_factory`

   ◦

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

Cursor

1. `execute(sql[, parameters])`

   SQL。 SQLSQL。 sqlite3? “qmark style”:`name` “named style”。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

   `%s`SQLSQLSQL 。

2. `executemany(sql, seq_of_parameters)`

   sqlSQL。 sqlite3。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

   executemany。 。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()
```

```
c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

3. `executescript(sql_script)`

   SQL。 `COMMIT`SQL。

   `sql_script`str或bytes。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

   SQL`SELECT`。 `SELECT`fetchone()fetchall()。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

4. `fetchone()`

   None。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()
```

```
c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

5. `fetchmany(size=cursor.arraysize)`

   size。 sizefetchmany。 。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

6. `fetchall()`

   。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

SQLitePython

SQLiteNULLINTEGERREALTEXTBLOB。

SQLPython。

```
import sqlite3

conn = sqlite3.connect("users.db")
c = conn.cursor()

c.execute("CREATE TABLE user (name text, age integer)")

c.execute("INSERT INTO user VALUES ('User A', 42)")
c.execute("INSERT INTO user VALUES ('User B', 43)")

conn.commit()

c.execute("SELECT * FROM user")
print(c.fetchall())

conn.close()
```

psycopg2PostgreSQL

psycopg2PostgreSQL。 PostgreSQL。

> Python DB API 2.0

```
import psycopg2

# Establish a connection to the database.
# Replace parameter values with database credentials.
conn = psycopg2.connect(database="testpython",
                        user="postgres",
                        host="localhost",
                        password="abc123",
                        port="5432")

# Create a cursor. The cursor allows you to execute database queries.
cur = conn.cursor()

# Create a table. Initialise the table name, the column names and data type.
cur.execute("""CREATE TABLE FRUITS (
                    id          INT ,
                    fruit_name  TEXT,
                    color       TEXT,
                    price       REAL
            )""")
conn.commit()
conn.close()
```

```
import psycopg2

# Establish a connection to the database.
# Replace parameter values with database credentials.
conn = psycopg2.connect(database="testpython",
```

```
                        user="postgres",
                        host="localhost",
                        password="abc123",
                        port="5432")

# Create a cursor. The cursor allows you to execute database queries.
cur = conn.cursor()

# Create a table. Initialise the table name, the column names and data type.
cur.execute("""CREATE TABLE FRUITS (
                    id          INT ,
                    fruit_name  TEXT,
                    color       TEXT,
                    price       REAL
            )""")
conn.commit()
conn.close()
```

```
import psycopg2

# Establish a connection to the database.
# Replace parameter values with database credentials.
conn = psycopg2.connect(database="testpython",
                        user="postgres",
                        host="localhost",
                        password="abc123",
                        port="5432")

# Create a cursor. The cursor allows you to execute database queries.
cur = conn.cursor()

# Create a table. Initialise the table name, the column names and data type.
cur.execute("""CREATE TABLE FRUITS (
                    id          INT ,
                    fruit_name  TEXT,
                    color       TEXT,
                    price       REAL
            )""")
conn.commit()
conn.close()
```

```
import psycopg2

# Establish a connection to the database.
# Replace parameter values with database credentials.
conn = psycopg2.connect(database="testpython",
                        user="postgres",
                        host="localhost",
                        password="abc123",
                        port="5432")

# Create a cursor. The cursor allows you to execute database queries.
cur = conn.cursor()

# Create a table. Initialise the table name, the column names and data type.
cur.execute("""CREATE TABLE FRUITS (
                    id          INT ,
                    fruit_name  TEXT,
```

```
                    color       TEXT,
                    price       REAL
            )""")
conn.commit()
conn.close()
```

psycopg2 :)

Oracle

- cx_Oracle -
- Oracle - Windows x64 Linux x64

- cx_Oracle

  ```
  sudo rpm -i <YOUR_PACKAGE_FILENAME>
  ```

- Oracle

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

['12''1''0''2''0']

SELECT

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

Python

10'SYSADMIN''IT-INFRA'7

23''''6

INSERT

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

Oracle//`commit` 。 。

INSERT

。 SQL。

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

```
ORACLE_HOME=<PATH_TO_INSTANTCLIENT>
PATH=$ORACLE_HOME:$PATH
LD_LIBRARY_PATH=<PATH_TO_INSTANTCLIENT>:$LD_LIBRARY_PATH
```

close。 。

PEP 249`connect()``connect() Connection`。 。 。

```
import MyDBAPI

con = MyDBAPI.connect(*database_dependent_args)
```

1

```
import MyDBAPI

con = MyDBAPI.connect(*database_dependent_args)
```

。 `Connection.__del___``Connection.__del___` 。 。

2

```
import MyDBAPI

con = MyDBAPI.connect(*database_dependent_args)
```

。

3

```
import MyDBAPI

con = MyDBAPI.connect(*database_dependent_args)
```

。 。

4

```
import MyDBAPI
```

```
con = MyDBAPI.connect(*database_dependent_args)
```

Cursor◦ ◦

sqlalchemy

sqlalchemy

```
from sqlalchemy import create_engine
from sqlalchemy.engine.url import URL


url = URL(drivername='mysql',
          username='user',
          password='passwd',
          host='host',
          database='db')

engine = create_engine(url)  # sqlalchemy engine
```

pandasmysql

```
from sqlalchemy import create_engine
from sqlalchemy.engine.url import URL


url = URL(drivername='mysql',
          username='user',
          password='passwd',
          host='host',
          database='db')

engine = create_engine(url)  # sqlalchemy engine
```

https://riptutorial.com/zh-TW/python/topic/4240/

143:

Examples

iterablecollections.Counter

```
from collections import Counter

c = Counter(["a", "b", "c", "d", "a", "b", "a", "c", "d"])
c
# Out: Counter({'a': 3, 'b': 2, 'c': 2, 'd': 2})
c["a"]
# Out: 3

c[7]     # not in the list (7 occurred 0 times!)
# Out: 0
```

`collections.Counter`iterable。

`dictcollections.Mapping` likeCounter

```
from collections import Counter

c = Counter(["a", "b", "c", "d", "a", "b", "a", "c", "d"])
c
# Out: Counter({'a': 3, 'b': 2, 'c': 2, 'd': 2})
c["a"]
# Out: 3

c[7]     # not in the list (7 occurred 0 times!)
# Out: 0
```

-scollections.Counter.most_common

`collections.CounterMapping`

```
from collections import Counter
adict = {'a': 5, 'b': 3, 'c': 5, 'd': 2, 'e':2, 'q': 5}
Counter(adict.values())
# Out: Counter({2: 2, 3: 1, 5: 3})
```

`most_common` -method

```
from collections import Counter
adict = {'a': 5, 'b': 3, 'c': 5, 'd': 2, 'e':2, 'q': 5}
Counter(adict.values())
# Out: Counter({2: 2, 3: 1, 5: 3})
```

list.counttuple.count

```
alist = [1, 2, 3, 4, 1, 2, 1, 3, 4]
alist.count(1)
# Out: 3

atuple = ('bear', 'weasel', 'bear', 'frog')
atuple.count('bear')
# Out: 2
atuple.count('fox')
# Out: 0
```

str.count

```
astring = 'thisisashorttext'
astring.count('t')
# Out: 4
```


```
astring = 'thisisashorttext'
astring.count('t')
# Out: 4
```


`collections.Counter`


```
astring = 'thisisashorttext'
astring.count('t')
# Out: 4
```


numpy

numpy。

```
>>> import numpy as np
>>> a=np.array([0,3,4,3,5,4,7])
>>> print np.sum(a==3)
2
```

1。 。 dtype。

numpy。 bincount。 bincount1d。


```
>>> import numpy as np
>>> a=np.array([0,3,4,3,5,4,7])
>>> print np.sum(a==3)
2
```


numpynumpy。

https://riptutorial.com/zh-TW/python/topic/476/

144:

Python“”CJava。 。

b	1
B	1
c	1
u	2unicode
h	2
H	2
i	2
I	2
w	4unicode
l	4
L	4
f	4
d	8

Examples

。 Python。

pythonpython。 Python。

PythonPython。

python`array`。 。 python`array`

```
from array import *
```

`array`。

```
from array import *
```

`arrayIdentifierName typecode`python `Initializers`。

Typecodes。 。

python

```
from array import *
```

typecode`i` 。 2。

5

```
from array import *
```

。 Python。

```
my_array = array('i', [1,2,3,4,5])
print(my_array[1])
# 2
print(my_array[2])
# 3
print(my_array[0])
# 1
```

append

```
my_array = array('i', [1,2,3,4,5])
my_array.append(6)
# array('i', [1, 2, 3, 4, 5, 6])
```

`6`。

insert

`insert()`。

```
my_array = array('i', [1,2,3,4,5])
my_array.insert(0,0)
#array('i', [0, 1, 2, 3, 4, 5])
```

00。 。

extendpython

`extend()`python。

```
my_array = array('i', [1,2,3,4,5])
my_extnd_array = array('i', [7,8,9,10])
my_array.extend(my_extnd_array)
```

```
# array('i', [1, 2, 3, 4, 5, 7, 8, 9, 10])
```

my_array`my_extnd_array`。

fromlist

```
my_array = array('i', [1,2,3,4,5])
c=[11,12,13]
my_array.fromlist(c)
# array('i', [1, 2, 3, 4, 5, 11, 12, 13])
```

11,1213`c``my_array`。

remove

```
my_array = array('i', [1,2,3,4,5])
my_array.remove(4)
# array('i', [1, 2, 3, 5])
```

4。

pop

`pop`。

```
my_array = array('i', [1,2,3,4,5])
my_array.pop()
# array('i', [1, 2, 3, 4])
```

`5`。

index

`index()`。 。

```
my_array = array('i', [1,2,3,4,5])
print(my_array.index(5))
# 5
my_array = array('i', [1,2,3,3,5])
print(my_array.index(3))
# 3
```

reversepython

`reverse()` - 。

```
my_array = array('i', [1,2,3,4,5])
my_array.reverse()
# array('i', [5, 4, 3, 2, 1])
```

buffer_info

。

```
my_array = array('i', [1,2,3,4,5])
my_array.buffer_info()
(33881712, 5)
```

count

`count()`。 3。

```
my_array = array('i', [1,2,3,3,5])
my_array.count(3)
# 2
```

tostring

`tostring()`。

```
my_char_array = array('c', ['g','e','e','k'])
# array('c', 'geek')
print(my_char_array.tostring())
# geek
```

tolistpython

Python `listtolist()`。

```
my_array = array('i', [1,2,3,4,5])
c = my_array.tolist()
# [1, 2, 3, 4, 5]
```

fromstringchar

`fromstring()`

```
my_char_array = array('c', ['g','e','e','k'])
my_char_array.fromstring("stuff")
print(my_char_array)
#array('c', 'geekstuff')
```

https://riptutorial.com/zh-TW/python/topic/4866/

145: I / O.

Python。 Python/。 Python。

- file_object = openfilename [access_mode] [buffering]

access_mode	

Python`open() encoding`。 。

`linuxutf-8`MACWindows。

```
import sys
sys.getdefaultencoding()
```

python。

。

```
import sys
sys.getdefaultencoding()
```

Examples

`mode`。

- `'r'` - 。 。 。 。
- `'w'` - 。 。
- `'a'` - 。 。 。
- `'rb'` - 。 `r`。 。
- `'r+'` - 。 `rw` 。
- `'rb+'` - 。 `r+`
- `'wb'` - 。 `w`。
- `'w+'` - 。 `r+`。 。
- `'wb+'`

- ◦ w+◦

- 'ab' - ◦ a◦
- 'a+' - ◦ w+◦ ◦
- 'ab+' - ◦ a+◦

```
with open(filename, 'r') as f:
    f.read()
with open(filename, 'w') as f:
    f.write(filedata)
with open(filename, 'a') as f:
    f.write('\n' + newdata)
```

	[R	R +	w ^	W +		A +
	✔	✔	✘	✔	✘	✔
	✘	✔	✔	✔	✔	✔
	✘	✘	✔	✔	✔	✔
	✘	✘	✔	✔	✘	✘

Python 3exclusive creation◦

- 'x' - FileExistsError
- 'xb' - ◦ x◦
- 'x+' - ◦ w+◦ FileExistsError ◦
- 'xb+' - ◦ x+

	X	X +
	✘	✔
	✔	✔
	✔	✔
	✘	✘

pythonic

Python 3.x 3.3

```
 with open(filename, 'r') as f:
     f.read()
 with open(filename, 'w') as f:
     f.write(filedata)
 with open(filename, 'a') as f:
     f.write('\n' + newdata)
```

Python 2

Python 2.x 2.0

```
 with open(filename, 'r') as f:
     f.read()
 with open(filename, 'w') as f:
     f.write(filedata)
 with open(filename, 'a') as f:
     f.write('\n' + newdata)
```

```
with open('myfile.txt', 'r') as fp:
    for line in fp:
        print(line)
```

`readline()`。

```
with open('myfile.txt', 'r') as fp:
    for line in fp:
        print(line)
```

forreadline。

`readlines()`

```
with open('myfile.txt', 'r') as fp:
    for line in fp:
        print(line)
```

0

1

i / o`with`。 。

```
with open('myfile.txt') as in_file:
    content = in_file.read()

print(content)
```

`withclose`

```
with open('myfile.txt') as in_file:
    content = in_file.read()

print(content)
```

`with`

```
with open('myfile.txt') as in_file:
    content = in_file.read()

print(content)
```

```
with open('myfile.txt', 'w') as f:
    f.write("Line 1")
    f.write("Line 2")
    f.write("Line 3")
    f.write("Line 4")
```

`myfile.txt`

12Line 3Line 4

Python

```
with open('myfile.txt', 'w') as f:
    f.write("Line 1")
    f.write("Line 2")
    f.write("Line 3")
    f.write("Line 4")
```

1
2
3
4

`os.linesep`;`\n`。

`encoding`open

```
with open('myfile.txt', 'w') as f:
    f.write("Line 1")
    f.write("Line 2")
    f.write("Line 3")
    f.write("Line 4")
```

print。 Python 2Python 3。

Python 3.x 3.0

```
with open('myfile.txt', 'w') as f:
    f.write("Line 1")
    f.write("Line 2")
    f.write("Line 3")
    f.write("Line 4")
```

Python 2

Python 2.x 2.0

```
with open('myfile.txt', 'w') as f:
    f.write("Line 1")
    f.write("Line 2")
    f.write("Line 3")
    f.write("Line 4")
```

writeprint。

```
with open(input_file, 'r') as in_file, open(output_file, 'w') as out_file:
    for line in in_file:
        out_file.write(line)
```

- `shutil`

```
with open(input_file, 'r') as in_file, open(output_file, 'w') as out_file:
    for line in in_file:
        out_file.write(line)
```

EAFP`try`。

```
import errno

try:
    with open(path) as f:
        # File exists
except IOError as e:
    # Raise the exception if it is not ENOENT (No such file or directory)
    if e.errno != errno.ENOENT:
        raise
    # No such file or directory
```

。

- `os`

  ```
  import errno

  try:
      with open(path) as f:
          # File exists
  except IOError as e:
      # Raise the exception if it is not ENOENT (No such file or directory)
      if e.errno != errno.ENOENT:
          raise
      # No such file or directory
  ```

Python 3.x 3.4

- `pathlib`

  ```
  import errno

  try:
      with open(path) as f:
  ```

```
        # File exists
except IOError as e:
    # Raise the exception if it is not ENOENT (No such file or directory)
    if e.errno != errno.ENOENT:
        raise
    # No such file or directory
```

EAFP

```
import errno

try:
    with open(path) as f:
        # File exists
except IOError as e:
    # Raise the exception if it is not ENOENT (No such file or directory)
    if e.errno != errno.ENOENT:
        raise
    # No such file or directory
```

```
import shutil
source='//192.168.1.2/Daily Reports'
destination='D:\\Reports\\Today'
shutil.copytree(source, destination)
```

。

os.walk

```
import os
for root, folders, files in os.walk(root_dir):
    for filename in files:
        print root, filename
```

root_dir“。”。

Python 3.x 3.5

os.scandir

```
import os
for root, folders, files in os.walk(root_dir):
    for filename in files:
        print root, filename
```

itertools

```
import itertools

with open('myfile.txt', 'r') as f:
    for line in itertools.islice(f, 12, 30):
        # do something here
```

1320python0。 10

next()。

readline()

mmap

mmap。 。

```
import mmap

with open('filename.ext', 'r') as fd:
    # 0: map the whole file
    mm = mmap.mmap(fd.fileno(), 0)

    # print characters at indices 5 through 10
    print mm[5:10]

    # print the line starting from mm's current position
    print mm.readline()

    # write a character to the 5th index
    mm[5] = 'a'

    # return mm's position to the beginning of the file
    mm.seek(0)

    # close the mmap object
    mm.close()
```

```
import fileinput

replacements = {'Search1': 'Replace1',
                'Search2': 'Replace2'}

for line in fileinput.input('filename.txt', inplace=True):
    for search_for in replacements:
        replace_with = replacements[search_for]
        line = line.replace(search_for, replace_with)
    print(line, end='')
```

```
>>> import os
>>> os.stat(path_to_file).st_size == 0
```

```
>>> import os
>>> os.stat(path_to_file).st_size == 0
```

。

```
>>> import os
>>> os.stat(path_to_file).st_size == 0
```

bool。

I / O. https://riptutorial.com/zh-TW/python/topic/267/i---o-

146:

Examples

PythonPython 23`print` with `()`

```
class Parent(object):
    def introduce(self):
        print("Hello!")

    def print_name(self):
        print("Parent")


class Child(Parent):
    def print_name(self):
        print("Child")


p = Parent()
c = Child()

p.introduce()
p.print_name()

c.introduce()
c.print_name()

$ python basic_override.py
Hello!
Parent
Hello!
Child
```

`ChildParent◦ ◦ introduceChildParentChild◦`

`Childprint_name◦ c.print_name()"Parent" ◦ ChildParentprint_namec.print_name()"Child"◦`

https://riptutorial.com/zh-TW/python/topic/3131/

147:

Python。

Examples

`datetime`Python 3.2+`%z`。

UTC`+HHMM-HHMM -HHMM -HHMM`。

Python 3.x 3.2

```
import datetime
dt = datetime.datetime.strptime("2016-04-15T08:27:18-0500", "%Y-%m-%dT%H:%M:%S%z")
```

Python`dateutil``datetime`。

```
import datetime
dt = datetime.datetime.strptime("2016-04-15T08:27:18-0500", "%Y-%m-%dT%H:%M:%S%z")
```

`dt``datetime`

```
import datetime
dt = datetime.datetime.strptime("2016-04-15T08:27:18-0500", "%Y-%m-%dT%H:%M:%S%z")
```

。 。 `timedelta`

```
import datetime

today = datetime.date.today()
print('Today:', today)

yesterday = today - datetime.timedelta(days=1)
print('Yesterday:', yesterday)

tomorrow = today + datetime.timedelta(days=1)
print('Tomorrow:', tomorrow)

print('Time between tomorrow and yesterday:', tomorrow - yesterday)
```

```
import datetime

today = datetime.date.today()
print('Today:', today)

yesterday = today - datetime.timedelta(days=1)
print('Yesterday:', yesterday)

tomorrow = today + datetime.timedelta(days=1)
print('Tomorrow:', tomorrow)
```

```
print('Time between tomorrow and yesterday:', tomorrow - yesterday)
```

datetime - ◦

```
import datetime

# Date object
today = datetime.date.today()
new_year = datetime.date(2017, 01, 01) #datetime.date(2017, 1, 1)

# Time object
noon = datetime.time(12, 0, 0) #datetime.time(12, 0)

# Current datetime
now = datetime.datetime.now()

# Datetime object
millenium_turn = datetime.datetime(2000, 1, 1, 0, 0, 0) #datetime.datetime(2000, 1, 1, 0, 0)
```

TypeError◦

```
import datetime

# Date object
today = datetime.date.today()
new_year = datetime.date(2017, 01, 01) #datetime.date(2017, 1, 1)

# Time object
noon = datetime.time(12, 0, 0) #datetime.time(12, 0)

# Current datetime
now = datetime.datetime.now()

# Datetime object
millenium_turn = datetime.datetime(2000, 1, 1, 0, 0, 0) #datetime.datetime(2000, 1, 1, 0, 0)
```

◦ datetimetimedelta

```
import datetime

# The size of each step in days
day_delta = datetime.timedelta(days=1)

start_date = datetime.date.today()
end_date = start_date + 7*day_delta

for i in range((end_date - start_date).days):
    print(start_date + i*day_delta)
```

```
import datetime

# The size of each step in days
day_delta = datetime.timedelta(days=1)

start_date = datetime.date.today()
```

```
end_date = start_date + 7*day_delta

for i in range((end_date - start_date).days):
    print(start_date + i*day_delta)
```

dateutil“”。

CST - tzinfo。

```
from dateutil import tz
from dateutil.parser import parse

ET = tz.gettz('US/Eastern')
CT = tz.gettz('US/Central')
MT = tz.gettz('US/Mountain')
PT = tz.gettz('US/Pacific')

us_tzinfos = {'CST': CT, 'CDT': CT,
              'EST': ET, 'EDT': ET,
              'MST': MT, 'MDT': MT,
              'PST': PT, 'PDT': PT}

dt_est = parse('2014-01-02 04:00:00 EST', tzinfos=us_tzinfos)
dt_pst = parse('2016-03-11 16:00:00 PST', tzinfos=us_tzinfos)
```

```
from dateutil import tz
from dateutil.parser import parse

ET = tz.gettz('US/Eastern')
CT = tz.gettz('US/Central')
MT = tz.gettz('US/Mountain')
PT = tz.gettz('US/Pacific')

us_tzinfos = {'CST': CT, 'CDT': CT,
              'EST': ET, 'EDT': ET,
              'MST': MT, 'MDT': MT,
              'PST': PT, 'PDT': PT}

dt_est = parse('2014-01-02 04:00:00 EST', tzinfos=us_tzinfos)
dt_pst = parse('2016-03-11 16:00:00 PST', tzinfos=us_tzinfos)
```

pytz

```
from dateutil import tz
from dateutil.parser import parse

ET = tz.gettz('US/Eastern')
CT = tz.gettz('US/Central')
MT = tz.gettz('US/Mountain')
PT = tz.gettz('US/Pacific')

us_tzinfos = {'CST': CT, 'CDT': CT,
              'EST': ET, 'EDT': ET,
              'MST': MT, 'MDT': MT,
              'PST': PT, 'PDT': PT}

dt_est = parse('2014-01-02 04:00:00 EST', tzinfos=us_tzinfos)
```

```
dt_pst = parse('2016-03-11 16:00:00 PST', tzinfos=us_tzinfos)
```

pytzpytz

```
from dateutil import tz
from dateutil.parser import parse

ET = tz.gettz('US/Eastern')
CT = tz.gettz('US/Central')
MT = tz.gettz('US/Mountain')
PT = tz.gettz('US/Pacific')

us_tzinfos = {'CST': CT, 'CDT': CT,
              'EST': ET, 'EDT': ET,
              'MST': MT, 'MDT': MT,
              'PST': PT, 'PDT': PT}

dt_est = parse('2014-01-02 04:00:00 EST', tzinfos=us_tzinfos)
dt_pst = parse('2016-03-11 16:00:00 PST', tzinfos=us_tzinfos)
```

localize

```
from dateutil import tz
from dateutil.parser import parse

ET = tz.gettz('US/Eastern')
CT = tz.gettz('US/Central')
MT = tz.gettz('US/Mountain')
PT = tz.gettz('US/Pacific')

us_tzinfos = {'CST': CT, 'CDT': CT,
              'EST': ET, 'EDT': ET,
              'MST': MT, 'MDT': MT,
              'PST': PT, 'PDT': PT}

dt_est = parse('2014-01-02 04:00:00 EST', tzinfos=us_tzinfos)
dt_pst = parse('2016-03-11 16:00:00 PST', tzinfos=us_tzinfos)
```

datetime。 tzinfotzinfoUTC。

UTCPython 3.2+ datetimetimezone tzinfotimedeltaname

Python 3.x 3.2

```
from datetime import datetime, timedelta, timezone
JST = timezone(timedelta(hours=+9))

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=JST)
print(dt)
# 2015-01-01 12:00:00+09:00

print(dt.tzname())
# UTC+09:00

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=timezone(timedelta(hours=9), 'JST'))
print(dt.tzname)
# 'JST'
```

3.2Python dateutil ◦ dateutiltzoffset 2.5.3dateutil.tz.tzoffset(tzname, offset)offset

Python 3.x 3.2
Python 2.x 2.7

```
from datetime import datetime, timedelta, timezone
JST = timezone(timedelta(hours=+9))

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=JST)
print(dt)
# 2015-01-01 12:00:00+09:00

print(dt.tzname())
# UTC+09:00

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=timezone(timedelta(hours=9), 'JST'))
print(dt.tzname)
# 'JST'
```

python◦ pytzdateutil◦

dateutiltz ◦ tz.gettz()datetime

```
from datetime import datetime, timedelta, timezone
JST = timezone(timedelta(hours=+9))

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=JST)
print(dt)
# 2015-01-01 12:00:00+09:00

print(dt.tzname())
# UTC+09:00

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=timezone(timedelta(hours=9), 'JST'))
print(dt.tzname)
# 'JST'
```

2.5.3 dateutil◦ dateutil2015-11-01 1:30 EDT-4 ◦

pytz pytz◦ localizepytz

```
from datetime import datetime, timedelta, timezone
JST = timezone(timedelta(hours=+9))

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=JST)
print(dt)
# 2015-01-01 12:00:00+09:00

print(dt.tzname())
# UTC+09:00

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=timezone(timedelta(hours=9), 'JST'))
print(dt.tzname)
# 'JST'
```

pytzUTC pytznormalize()

```
from datetime import datetime, timedelta, timezone
JST = timezone(timedelta(hours=+9))

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=JST)
print(dt)
# 2015-01-01 12:00:00+09:00

print(dt.tzname())
# UTC+09:00

dt = datetime(2015, 1, 1, 12, 0, 0, tzinfo=timezone(timedelta(hours=9), 'JST'))
print(dt.tzname)
# 'JST'
```

dateutil“”。

```
from dateutil.parser import parse

dt = parse("Today is January 1, 2047 at 8:21:00AM", fuzzy=True)
print(dt)
```

dt*datetime* datetime.datetime(2047, 1, 1, 8, 21) 。

。

```
from datetime import datetime
from dateutil import tz

utc = tz.tzutc()
local = tz.tzlocal()

utc_now = datetime.utcnow()
utc_now # Not timezone-aware.

utc_now = utc_now.replace(tzinfo=utc)
utc_now # Timezone-aware.

local_now = utc_now.astimezone(local)
local_now # Converted to local time.
```

ISO 8601

PythonISO 8601。 strptime。 datetimeISO 86016

```
str(datetime.datetime(2016, 7, 22, 9, 25, 59, 555555))
# '2016-07-22 09:25:59.555555'
```

0

```
str(datetime.datetime(2016, 7, 22, 9, 25, 59, 555555))
# '2016-07-22 09:25:59.555555'
```

strptime。 strptime' does not support at all parsing minute timezones that have a in it, thus

2016-07-22 092559 + 0300 can be parsed, but the standard format 2016-07-22 09:25:59 +0300`。

iso8601ISO 8601。

T

```
str(datetime.datetime(2016, 7, 22, 9, 25, 59, 555555))
# '2016-07-22 09:25:59.555555'
```

iso8601.parse_dateUTC。 default_zone。 None

```
str(datetime.datetime(2016, 7, 22, 9, 25, 59, 555555))
# '2016-07-22 09:25:59.555555'
```

datetimePOSIX timestampITC datetime。

197011。

```
import time
from datetime import datetime
seconds_since_epoch=time.time()  #1469182681.709

utc_date=datetime.utcfromtimestamp(seconds_since_epoch) #datetime.datetime(2016, 7, 22, 10,
18, 1, 709000)
```

calendar

```
import calendar
from datetime import date

def monthdelta(date, delta):
    m, y = (date.month+delta) % 12, date.year + ((date.month)+delta-1) // 12
    if not m: m = 12
    d = min(date.day, calendar.monthrange(y, m)[1])
    return date.replace(day=d,month=m, year=y)

next_month = monthdelta(date.today(), 1) #datetime.date(2016, 10, 23)
```

dateutils

```
import calendar
from datetime import date

def monthdelta(date, delta):
    m, y = (date.month+delta) % 12, date.year + ((date.month)+delta-1) // 12
    if not m: m = 12
    d = min(date.day, calendar.monthrange(y, m)[1])
    return date.replace(day=d,month=m, year=y)

next_month = monthdelta(date.today(), 1) #datetime.date(2016, 10, 23)
```

timedelta

```
from datetime import datetime, timedelta
```

```
now = datetime.now()
then = datetime(2016, 5, 23)     # datetime.datetime(2016, 05, 23, 0, 0, 0)
```

datetime

```
from datetime import datetime, timedelta
now = datetime.now()
then = datetime(2016, 5, 23)     # datetime.datetime(2016, 05, 23, 0, 0, 0)
```

deltatimedelta

```
from datetime import datetime, timedelta
now = datetime.now()
then = datetime(2016, 5, 23)     # datetime.datetime(2016, 05, 23, 0, 0, 0)
```

n

n

```
from datetime import datetime, timedelta
now = datetime.now()
then = datetime(2016, 5, 23)     # datetime.datetime(2016, 05, 23, 0, 0, 0)
```

```
from datetime import datetime, timedelta
now = datetime.now()
then = datetime(2016, 5, 23)     # datetime.datetime(2016, 05, 23, 0, 0, 0)
```

ISO 8601

```
from datetime import datetime

datetime.now().isoformat()
# Out: '2016-07-31T23:08:20.886783'
```

```
from datetime import datetime

datetime.now().isoformat()
# Out: '2016-07-31T23:08:20.886783'
```

```
from datetime import datetime

datetime.now().isoformat()
# Out: '2016-07-31T23:08:20.886783'
```

ISO 8601ISO 8601 。

https://riptutorial.com/zh-TW/python/topic/484/

148:

Examples

```
from datetime import datetime

a = datetime(2016,10,06,0,0,0)
b = datetime(2016,10,01,23,59,59)

a-b
# datetime.timedelta(4, 1)

(a-b).days
# 4
(a-b).total_seconds()
# 518399.0
```

datetime

C 。

```
from datetime import datetime
datetime_string = 'Oct 1 2016, 00:00:00'
datetime_string_format = '%b %d %Y, %H:%M:%S'
datetime.strptime(datetime_string, datetime_string_format)
# datetime.datetime(2016, 10, 1, 0, 0)
```

datetime

C 。

```
from datetime import datetime
datetime_for_string = datetime(2016,10,1,0,0)
datetime_string_format = '%b %d %Y, %H:%M:%S'
datetime.strftime(datetime_for_string,datetime_string_format)
# Oct 01 2016, 00:00:00
```

https://riptutorial.com/zh-TW/python/topic/7284/

149:

Examples

```
from PIL import Image

im = Image.open("Image.bmp")
```

JPEG

```
from __future__ import print_function
import os, sys
from PIL import Image

for infile in sys.argv[1:]:
    f, e = os.path.splitext(infile)
    outfile = f + ".jpg"
    if infile != outfile:
        try:
            Image.open(infile).save(outfile)
        except IOError:
            print("cannot convert", infile)
```

https://riptutorial.com/zh-TW/python/topic/6841/

150:

PEP 435 3.4Python。

Examples

Python 2.43.3

Python 3.3Python 3.4Python 2.4。 PyPIenum34 backport。

```
pip install enum34
```

Python 3.4+

```
pip install enum34
```

```
class Color(Enum):
    red = 1
    green = 2
    blue = 3

[c for c in Color]  # [<Color.red: 1>, <Color.green: 2>, <Color.blue: 3>]
```

https://riptutorial.com/zh-TW/python/topic/947/

151:

ifelifelsePythonTrueFalse。 Python。

- <expression> if <conditional> else <expression> #Ternary Operator

Examples

ifelif

Python`if elif else`。

```
number = 5

if number > 2:
    print("Number is bigger than 2.")
elif number < 2:  # Optional clause (you can have multiple elifs)
    print("Number is smaller than 2.")
else:  # Optional clause (you can only have one else)
    print("Number is 2.")
```

```
Number is bigger than 2
```

`else ifelif`。

“”

。 。

- CRubyJavaPython“”。
- “”。

```
n = 5

"Greater than 2" if n > 2 else "Smaller than or equal to 2"
# Out: 'Greater than 2'
```

- True。

Tenary

```
n = 5

"Greater than 2" if n > 2 else "Smaller than or equal to 2"
# Out: 'Greater than 2'
```

lambda。

```
if condition:
    body
```

if。 True if。 False 。

```
if condition:
    body
```

```
if condition:
    body
```

```
if condition:
    body
else:
    body
```

Falseelse。

```
if condition:
    body
else:
    body
```

TrueFalse TrueFalse 。 Pythonicif-else。

True and。 False

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

---

orTrueTrue 。

```
>>> 1 and 2
2

>>> 1 and 0
0
```

```
>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

◦ ◦

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

`print_me`Python`0` `False` `False` ◦ `print_me`◦

◦

2. - `if (a) and (b > 2)` ◦ `a` `bool(a)` `True` ◦

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

◦

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

◦ - if (a == 3) or (4) or (6) ◦ bool(4)bool(6)True

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

in。

```
>>> 1 and 2
2

>>> 1 and 0
0

>>> 1 and "Hello World"
"Hello World"

>>> "" and "Pancakes"
""
```

False ◦

-
-
- 0 0L 0.0 0j
- '' "" () []
- {}
- __bool____len__0False

PythonTrue ◦

Falsey◦ if foo() is None if foo()

cmp

Python 2`cmp`。 。

`'greater than'x > y 'less than'x < y 'equal'x == y`。

```
['equal', 'greater than', 'less than', ][cmp(x,y)]

# x,y = 1,1 output: 'equal'
# x,y = 1,2 output: 'less than'
# x,y = 2,1 output: 'greater than'
```

`cmp(x,y)`

x <y	-1
x == y	0
x> y	1

Python 3。 Python 3`functoolscmp_to_key(func)`。

Python。

```
[value_false, value_true][<conditional-test>]
```

```
[value_false, value_true][<conditional-test>]
```

`n<=15False` Python0。 Python

```
[value_false, value_true][<conditional-test>]
```

Python 2.x 2.7

`__cmp__`30,1-1cmpxy0objecs1x> y -1x <y

01-1。

```
[value_false, value_true][<conditional-test>]
```

Python。

```
[value_false, value_true][<conditional-test>]
```

`()`lambda/。 。

```
[value_false, value_true][<conditional-test>]
```

None

None ◦ aDate ◦

is None◦

```
if aDate is None:
    aDate=datetime.date.today()
```

Pythonic is None== None ◦

not NoneTrue◦

```
if aDate is None:
    aDate=datetime.date.today()
```

Pythonic◦

```
if aDate is None:
    aDate=datetime.date.today()
```

◦ aDatenot None ◦ is None datetime.date.today()aDate ◦

https://riptutorial.com/zh-TW/python/topic/1111/

152:

PythonZIP

```
import sys
sys.path.append("package.zip")
```

Examples

Python。

`module.py`

```
def hi():
    print("Hello world!")
```

`my_script.py`

```
def hi():
    print("Hello world!")
```

```
def hi():
    print("Hello world!")
```

PythonCC ++。

`package`

- `__init__.py`
- `dog.py`
- `hi.py`

`__init__.py`

```
from package.dog import woof
from package.hi import hi
```

`dog.py`

```
from package.dog import woof
from package.hi import hi
```

`hi.py`

```
from package.dog import woof
from package.hi import hi
```

Python`__init__.py`。 `import package` `import package` `__init__.py`。 `package.hi` `package.woof`。

https://riptutorial.com/zh-TW/python/topic/3142/

153:

os.F_OK	accessmode。
os.R_OK	accessmode。
os.W_OK	accessmode。
os.X_OK	accessmode。

Examples

os.access

`os.access`。

```
import os
path = "/home/myFiles/directory1"

## Check if path exists
os.access(path, os.F_OK)

## Check if path is Readable
os.access(path, os.R_OK)

## Check if path is Wriable
os.access(path, os.W_OK)

## Check if path is Execuatble
os.access(path, os.E_OK)
```

```
import os
path = "/home/myFiles/directory1"

## Check if path exists
os.access(path, os.F_OK)

## Check if path is Readable
os.access(path, os.R_OK)

## Check if path is Wriable
os.access(path, os.W_OK)

## Check if path is Execuatble
os.access(path, os.E_OK)
```

`TrueFalse` 。 unixwindows。

https://riptutorial.com/zh-TW/python/topic/1262/

154:

Python re。

。 'amount\D+\d+'amount amount=100 amount is 3 amount is equal to: 33

-
- re.matchpatternstringflag = 0#OutstringpatternNone
- re.searchpatternstringflag = 0#OutstringpatternNone
- re.findallpatternstringflag = 0#Out[]
- re.finditerpatternstringflag = 0#Outre.findall
- re.subpatternreplacementstringflag = 0#out
-
- precompiled_pattern = re.compilepatternflag = 0
- precompiled_pattern.matchstring#Out
- precompiled_pattern.searchstring#Out
- precompiled_pattern.findallstring#out
- precompiled_pattern.substring / pattern / functionstring#Out

Examples

re.match()

```
import re

pattern = r"123"
string = "123zzb"

re.match(pattern, string)
# Out: <_sre.SRE_Match object; span=(0, 3), match='123'>

match = re.match(pattern, string)

match.group()
# Out: '123'
```

r 。

\“”“” \n \t \ \t \r 。 。

r"\n"

2 \n ◦ \d◦ r"\d" "\\d" r"\d" ◦

```
import re

pattern = r"123"
string = "123zzb"

re.match(pattern, string)
# Out: <_sre.SRE_Match object; span=(0, 3), match='123'>

match = re.match(pattern, string)

match.group()
# Out: '123'
```

◦ re.search

```
import re

pattern = r"123"
string = "123zzb"

re.match(pattern, string)
# Out: <_sre.SRE_Match object; span=(0, 3), match='123'>

match = re.match(pattern, string)

match.group()
# Out: '123'
```

```
pattern = r"(your base)"
sentence = "All your base are belong to us."

match = re.search(pattern, sentence)
match.group(1)
# Out: 'your base'

match = re.search(r"(belong.*)", sentence)
match.group(1)
# Out: 'belong to us.'
```

re.match◦ re.findall ◦

^

```
pattern = r"(your base)"
sentence = "All your base are belong to us."

match = re.search(pattern, sentence)
match.group(1)
# Out: 'your base'

match = re.search(r"(belong.*)", sentence)
match.group(1)
# Out: 'belong to us.'
```

$

```
pattern = r"(your base)"
sentence = "All your base are belong to us."

match = re.search(pattern, sentence)
match.group(1)
# Out: 'your base'

match = re.search(r"(belong.*)", sentence)
match.group(1)
# Out: 'belong to us.'
```

^$

```
pattern = r"(your base)"
sentence = "All your base are belong to us."

match = re.search(pattern, sentence)
match.group(1)
# Out: 'your base'

match = re.search(r"(belong.*)", sentence)
match.group(1)
# Out: 'belong to us.'
```

◦ group()◦

```
match.group() # Group without argument returns the entire match found
# Out: '123'
match.group(0) # Specifying 0 gives the same result as specifying no argument
# Out: '123'
```

group()◦

;◦

groups()◦

```
match.group() # Group without argument returns the entire match found
# Out: '123'
match.group(0) # Specifying 0 gives the same result as specifying no argument
# Out: '123'
```

```
match.group() # Group without argument returns the entire match found
# Out: '123'
match.group(0) # Specifying 0 gives the same result as specifying no argument
# Out: '123'
```


◦

(?:)。 “”。

```
match.group() # Group without argument returns the entire match found
# Out: '123'
match.group(0) # Specifying 0 gives the same result as specifying no argument
# Out: '123'
```

11+2211 11+ 。 +。 +。

[]

```
match = re.search(r'[b]', 'a[b]c')
match.group()
# Out: 'b'
```

```
match = re.search(r'[b]', 'a[b]c')
match.group()
# Out: 'b'
```

re.escape()

```
match = re.search(r'[b]', 'a[b]c')
match.group()
# Out: 'b'
```

re.escape()

```
match = re.search(r'[b]', 'a[b]c')
match.group()
# Out: 'b'
```

re.sub。

```
re.sub(r"t[0-9][0-9]", "foo", "my name t13 is t44 what t99 ever t44")
# Out: 'my name foo is foo what foo ever foo'
```

```
re.sub(r"t[0-9][0-9]", "foo", "my name t13 is t44 what t99 ever t44")
# Out: 'my name foo is foo what foo ever foo'
```

ID“10” \10“ID10”。 \g<i>

```
re.sub(r"t[0-9][0-9]", "foo", "my name t13 is t44 what t99 ever t44")
# Out: 'my name foo is foo what foo ever foo'
```

```
re.sub(r"t[0-9][0-9]", "foo", "my name t13 is t44 what t99 ever t44")
# Out: 'my name foo is foo what foo ever foo'
```

```
re.findall(r"[0-9]{2,3}", "some 1 text 12 is 945 here 4445588899")
# Out: ['12', '945', '444', '558', '889']
```

`"[0-9]{2,3}"r`python;“”。

`re.finditer()re.findall()SRE_Match`

```
re.findall(r"[0-9]{2,3}", "some 1 text 12 is 945 here 4445588899")
# Out: ['12', '945', '444', '558', '889']
```

```
import re

precompiled_pattern = re.compile(r"(\d+)")
matches = precompiled_pattern.search("The answer is 41!")
matches.group(1)
# Out: 41

matches = precompiled_pattern.search("Or was it 42?")
matches.group(1)
# Out: 42
```

。 Python docs SO “”。

```
import re

precompiled_pattern = re.compile(r"(\d+)")
matches = precompiled_pattern.search("The answer is 41!")
matches.group(1)
# Out: 41

matches = precompiled_pattern.search("Or was it 42?")
matches.group(1)
# Out: 42
```

re.match。

azAZ0-9

```
import re

def is_allowed(string):
    characherRegex = re.compile(r'[^a-zA-Z0-9.]')
    string = characherRegex.search(string)
    return not bool(string)

print (is_allowed("abyzABYZ0099"))
# Out: 'True'

print (is_allowed("#*@#$%^"))
# Out: 'False'
```

`[^a-zA-Z0-9.][^a-z0-9.]`。

http //stackoverflow.com/a/1325265/2697955

。

```
import re
data = re.split(r'\s+', 'James 94 Samantha 417 Scarlett 74')
print( data )
# Output: ['James', '94', 'Samantha', '417', 'Scarlett', '74']
```

。 flags。

re.searchre。

```
m = re.search("b", "ABC")
m is None
# Out: True

m = re.search("b", "ABC", flags=re.IGNORECASE)
m.group()
# Out: 'B'

m = re.search("a.b", "A\nBC", flags=re.IGNORECASE)
m is None
# Out: True

m = re.search("a.b", "A\nBC", flags=re.IGNORECASE|re.DOTALL)
m.group()
# Out: 'A\nB'
```

re.IGNORECASE re.I	
re.DOTALL re.S	.
re.MULTILINE re.M	^ $
re.DEBUG	

(?iLmsux) 'i"L"m"s"u"x'。

;re.Ire.Lre.Mre.Sre.UUnicodere.X。 re.compile。

x。 。 。

`re.finditer`

re.finditer。 re.findall

```
import re
text = 'You can try to find an ant in this string'
pattern = 'an?\w' # find 'an' either with or without a following word character

for match in re.finditer(pattern, text):
    # Start index of match (integer)
```

```
    sStart = match.start()

    # Final index of match (integer)
    sEnd = match.end()

    # Complete match (string)
    sGroup = match.group()

    # Print match
    print('Match "{}" found at: [{},{}]'.format(sGroup, sStart,sEnd))
```

```
import re
text = 'You can try to find an ant in this string'
pattern = 'an?\w' # find 'an' either with or without a following word character

for match in re.finditer(pattern, text):
    # Start index of match (integer)
    sStart = match.start()

    # Final index of match (integer)
    sEnd = match.end()

    # Complete match (string)
    sGroup = match.group()

    # Print match
    print('Match "{}" found at: [{},{}]'.format(sGroup, sStart,sEnd))
```

。

```
An apple a day keeps the doctor away (I eat an apple everyday).
```

“”**regex**。

```
An apple a day keeps the doctor away (I eat an apple everyday).
```

```
An apple a day keeps the doctor away (I eat an apple everyday).
```

“apple”。

- `(*SKIP)`“”。 `(*FAIL)`。
- `(*SKIP)`。 `(*SKIP)`。

https://riptutorial.com/zh-TW/python/topic/632/--

155:

- = -
- == -
- `>` -
- `<` -
- `>=` -
- `<=` -
- `is` -
- is not = test

X	
ÿ	

Examples

```
x > y
x < y
```

。

```
x > y
x < y
```

。

```
x > y
x < y
```

“”"gamma" < "OMEGA"。

```
x > y
x < y
```

<>。

```
x != y
```

True xyFalse ◦

```
x != y
```

```
x == y
```

xy◦ int 12'12' ◦

```
x == y
```

◦ ◦ TrueFalse ◦

```
x > y > z
```

```
x > y > z
```

TrueTrue ◦

```
x > y > z
```

OP◦

0 != 1 != 0True 0 != 0False ◦ x != y != zx yz◦ ==◦

```
x > y > z
```

True True ◦ ◦ “”

```
x > y > z
```

False False ◦

a > exp > bexp◦

```
x > y > z
```

expa > exp◦

`is` vs` ==`

is== ◦

a == bab

。

a is bab 。

```
a = 'Python is fun!'
b = 'Python is fun!'
a == b # returns True
a is b # returns False

a = [1, 2, 3, 4, 5]
b = a      # b references a
a == b     # True
a is b     # True
b = a[:]   # b now references a copy of a
a == b     # True
a is b     # False [!!]
```

isid(a) == id(b) 。

。 Python isTrue 。

```
a = 'Python is fun!'
b = 'Python is fun!'
a == b # returns True
a is b # returns False

a = [1, 2, 3, 4, 5]
b = a      # b references a
a == b     # True
a is b     # True
b = a[:]   # b now references a copy of a
a == b     # True
a is b     # False [!!]
```

。

```
a = 'Python is fun!'
b = 'Python is fun!'
a == b # returns True
a is b # returns False

a = [1, 2, 3, 4, 5]
b = a      # b references a
a == b     # True
a is b     # True
b = a[:]   # b now references a copy of a
a == b     # True
a is b     # False [!!]
```

isNone

```
a = 'Python is fun!'
b = 'Python is fun!'
a == b # returns True
a is b # returns False
```

```
a = [1, 2, 3, 4, 5]
b = a      # b references a
a == b     # True
a is b     # True
b = a[:]   # b now references a copy of a
a == b     # True
a is b     # False [!!]
```

isis“”。

```
a = 'Python is fun!'
b = 'Python is fun!'
a == b # returns True
a is b # returns False

a = [1, 2, 3, 4, 5]
b = a      # b references a
a == b     # True
a is b     # True
b = a[:]   # b now references a copy of a
a == b     # True
a is b     # False [!!]
```

__eq____ne__==!=。 __lt__ < __le__ <= __gt__ > __gt__ > __ge__ >。 Python ==not <not >

```
class Foo(object):
    def __init__(self, item):
        self.my_item = item
    def __eq__(self, other):
        return self.my_item == other.my_item

a = Foo(5)
b = Foo(5)
a == b     # True
a != b     # False
a is b     # False
```

other。

```
class Foo(object):
    def __init__(self, item):
        self.my_item = item
    def __eq__(self, other):
        return self.my_item == other.my_item

a = Foo(5)
b = Foo(5)
a == b     # True
a != b     # False
a is b     # False
```

isinstance()。

Python

Java

```
if("asgdsrf" == 0) {
    //do stuff
}
```

.....◦ ◦ Python - `False` ◦

```
if("asgdsrf" == 0) {
    //do stuff
}
```

`False` ◦

https://riptutorial.com/zh-TW/python/topic/248/

156:

- class *ClassName* *MainClass* *Mixin1* *Mixin2* ...*ClassName* mainfirstclass *MainClass* mixins *Mixin1* *Mixin2*。
- class *ClassName* *Mixin1* *MainClass* *Mixin2* ...'main';mixin

mixin。 mixin *FooFoo*`isinstance(instance, Foo)`true

Examples

Mixin。 ;。

`Car` `BoatPlane` 。 `travel`。 ;。 `Vehicle`

```
class Vehicle(object):
   """A generic vehicle class."""

   def __init__(self, position):
       self.position = position

   def travel(self, destination):
       route = calculate_route(from=self.position, to=destination)
       self.move_along(route)

class Car(Vehicle):
   ...

class Boat(Vehicle):
   ...

class Plane(Vehicle):
   ...
```

`car.travel("Montana")` `boat.travel("Hawaii")` `plane.travel("France")` `travel`

`Car``play_song_on_station` `Clock` 。 `CarClock` `Machine` 。 ; `BoatPlane`。 mixin。 Pythonmixin

```
class Vehicle(object):
   """A generic vehicle class."""

   def __init__(self, position):
       self.position = position

   def travel(self, destination):
       route = calculate_route(from=self.position, to=destination)
       self.move_along(route)

class Car(Vehicle):
   ...

class Boat(Vehicle):
   ...
```

```
class Plane(Vehicle):
   ...
```

`Foomain_supermixin`。

`Car`Clock`Car`

```
class Vehicle(object):
   """A generic vehicle class."""

   def __init__(self, position):
       self.position = position

   def travel(self, destination):
       route = calculate_route(from=self.position, to=destination)
       self.move_along(route)

class Car(Vehicle):
   ...

class Boat(Vehicle):
   ...

class Plane(Vehicle):
   ...
```

`car.play_song_on_station(98.7)clock.play_song_on_station(101.3)`
`boat.play_song_on_station(100.5)`

mixins“”。 mixins/。

Mixins“”。 mixin。 mixinPython。

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
        print "Mixin2"

class BaseClass(object):
    def test(self):
        print "Base"

class MyClass(BaseClass, Mixin1, Mixin2):
    pass
```

Mixin2Mixin1BaseClass。

```
class Mixin1(object):
    def test(self):
        print "Mixin1"

class Mixin2(object):
    def test(self):
```

```
        print "Mixin2"

class BaseClass(object):
    def test(self):
        print "Base"

class MyClass(BaseClass, Mixin1, Mixin2):
    pass
```

Base。

https://riptutorial.com/zh-TW/python/topic/4359/

157:

FlaskPythonWebPintrestTwilioLinkedin。 FlaskWeb。

- @ app.route“/ urlpath”methods = [“GET”“POST”“DELETE”“PUTS”“HEAD”“OPTIONS”]
- @ app.route“/ urlpath / <param>”methods = [“GET”“POST”“DELETE”“PUTS”“HEAD” “OPTIONS”]

Examples

```
# Imports the Flask class
from flask import Flask
# Creates an app and checks if its the main or imported
app = Flask(__name__)

# Specifies what URL triggers hello_world()
@app.route('/')
# The function run on the index route
def hello_world():
    # Returns the text to be displayed
    return "Hello World!"

# If this script isn't an import
if __name__ == "__main__":
    # Run the app until stopped
    app.run()
```

。 `127.0.0.1`**localhost** 。 **5000**。 WebWebURL `localhost:5000127.0.0.1:5000` 。 。

`app.run()` 。 `127.0.0.1` `0.0.0.0`WebURLIP。 5000`80` 80。 TrueFlask。

```
# Imports the Flask class
from flask import Flask
# Creates an app and checks if its the main or imported
app = Flask(__name__)

# Specifies what URL triggers hello_world()
@app.route('/')
# The function run on the index route
def hello_world():
    # Returns the text to be displayed
    return "Hello World!"

# If this script isn't an import
if __name__ == "__main__":
    # Run the app until stopped
    app.run()
```

URL

FlaskURL。 URL。 Flask`www.example.com` 。

```
@app.route("/")
def index():
    return "You went to www.example.com"

@app.route("/about")
def about():
    return "You went to www.example.com/about"

@app.route("/users/guido-van-rossum")
    return "You went to www.example.com/guido-van-rossum"
```

/ users /URL。 `@app.route()` FlaskURL

```
@app.route("/")
def index():
    return "You went to www.example.com"

@app.route("/about")
def about():
    return "You went to www.example.com/about"

@app.route("/users/guido-van-rossum")
    return "You went to www.example.com/guido-van-rossum"
```

HTTP

HTTP**GETPOST** 。 FlaskHTTPURL。WebURL。 GETURLPOST。DELETEPUT HTTP。

```
@app.route("/login", methods=["GET"])
def login_form():
    return "This is the login form"
@app.route("/login", methods=["POST"])
def login_auth():
    return "Processing your data"
@app.route("/login", methods=["DELETE", "PUT"])
def deny():
    return "This method is not allowed"
```

flask`request`。

```
@app.route("/login", methods=["GET"])
def login_form():
    return "This is the login form"
@app.route("/login", methods=["POST"])
def login_auth():
    return "Processing your data"
@app.route("/login", methods=["DELETE", "PUT"])
def deny():
    return "This method is not allowed"
```

POST`request`

```
@app.route("/login", methods=["GET"])
def login_form():
    return "This is the login form"
```

```
@app.route("/login", methods=["POST"])
def login_auth():
    return "Processing your data"
@app.route("/login", methods=["DELETE", "PUT"])
def deny():
    return "This method is not allowed"
```

`render_template()`HTML

```
from flask import Flask
from flask import render_template
app = Flask(__name__)

@app.route("/about")
def about():
    return render_template("about-us.html")

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=80, debug=True)
```

`about-us.html` 。

```
from flask import Flask
from flask import render_template
app = Flask(__name__)

@app.route("/about")
def about():
    return render_template("about-us.html")

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=80, debug=True)
```

HTML

```
<link rel="stylesheet" type="text/css", href="{{url_for('static', filename='styles/about-style.css')}}">
```

styles`about-style.css` 。 。

Meteor.jsFlask。 FlaskJinja Templating。 HTML。

HTML。 HTML。

```
@app.route("/users/<username>)
def profile(username):
    joinedDate = get_joined_date(username) # This function's code is irrelevant
    awards = get_awards(username) # This function's code is irrelevant
    # The joinDate is a string and awards is an array of strings
    return render_template("profile.html", username=username, joinDate=joinDate, awards=awards)
```

`render_template()`。 `profile.html`

```
@app.route("/users/<username>)
def profile(username):
    joinedDate = get_joined_date(username) # This function's code is irrelevant
    awards = get_awards(username) # This function's code is irrelevant
    # The joinDate is a string and awards is an array of strings
    return render_template("profile.html", username=username, joinDate=joinDate,
awards=awards)
```

- `{% ... %}`
- `{{ ... }}`
- `{# ... #}`
- `{# ... ##`

`request`。

```
from flask import request
```

`request.method` `request.form` `request.args` URL/。

```
from flask import request
```

URL

`www.example.com/api/users/guido-van-rossum?key=pa55w0Rd`

POST `request`

```
from flask import request
```

URLcookie。

```
from flask import request
```

https://riptutorial.com/zh-TW/python/topic/8682/

158: `exec``eval`

- eval[globals = None [locals = None]]
- EXEC
- exec
- exec

expression	code
object	code
globals	。 locals。 globals() 。
locals	。 globals。 globals()locals()globalslocals globals 。

exec globalslocals 。 globalslocals globals 。

globals__builtins__ keyPython。 printisinstanceglobals__builtins__None 。 。

Python 2; Python 3Python 2。 <s>

- exec object
- exec object in globals
- exec object in globals, locals

Examples

exec

```
>>> code = """for i in range(5):\n    print('Hello world!')"""
>>> exec(code)
Hello world!
Hello world!
Hello world!
Hello world!
Hello world!
```

eval

```
>>> expression = '5 + 3 * a'
>>> a = 5
>>> result = eval(expression)
>>> result
20
```

compile;eval。 。 compile3'eval' 。

```
>>> code = compile('a * b + c', '<string>', 'eval')
>>> code
<code object <module> at 0x7f0e51a58830, file "<string>", line 1>
>>> a, b, c = 1, 2, 3
>>> eval(code)
5
```

eval

```
>>> variables = {'a': 6, 'b': 7}
>>> eval('a * b', globals=variables)
42
```


```
>>> variables = {'a': 6, 'b': 7}
>>> eval('a * b', globals=variables)
42
```


`defaultdict`


```
>>> variables = {'a': 6, 'b': 7}
>>> eval('a * b', globals=variables)
42
```


ast.literal_evalPython

Python`ast.literal_evaleval`。 。

```
>>> import ast
>>> code = """(1, 2, {'foo': 'bar'})"""
>>> object = ast.literal_eval(code)
>>> object
(1, 2, {'foo': 'bar'})
>>> type(object)
<class 'tuple'>
```


```
>>> import ast
>>> code = """(1, 2, {'foo': 'bar'})"""
>>> object = ast.literal_eval(code)
>>> object
(1, 2, {'foo': 'bar'})
>>> type(object)
<class 'tuple'>
```


`()`CPython。 CPython。

execevalast.literal_eval

`evalexec`。 `ast.literal_eval`。 。

Python。

`exec``eval` https://riptutorial.com/zh-TW/python/topic/2251/-exec--eval-

159: Matplotlib

Matplotlib https://matplotlib.org/NumPy2D。 。 https://matplotlib.org/2.0.2/gallery.html https://matplotlib.org/2.0.2/examples/index.htmlhttp //www.riptutorial.com/topic/881

Examples

Matplotlib

Matplotlib

```
# Plotting tutorials in Python
# Launching a simple plot

import numpy as np
import matplotlib.pyplot as plt

# angle varying between 0 and 2*pi
x = np.linspace(0, 2.0*np.pi, 101)
y = np.sin(x)                        # sine function

plt.plot(x, y)
plt.show()
```

;。

```
# Plotting tutorials in Python
# Enhancing a plot

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2.0*np.pi, 101)
y = np.sin(x)

# values for making ticks in x and y axis
xnumbers = np.linspace(0, 7, 15)
ynumbers = np.linspace(-1, 1, 11)

plt.plot(x, y, color='r', label='sin') # r - red colour
plt.xlabel("Angle in Radians")
plt.ylabel("Magnitude")
plt.title("Plot of some trigonometric functions")
plt.xticks(xnumbers)
plt.yticks(ynumbers)
plt.legend()
plt.grid()
plt.axis([0, 6.5, -1.1, 1.1]) # [xstart, xend, ystart, yend]
plt.show()
```

MATLAB

。

```
# Plotting tutorials in Python
# Adding Multiple plots by superimposition
# Good for plots sharing similar x, y limits
# Using single plot command and legend

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2.0*np.pi, 101)
y = np.sin(x)
z = np.cos(x)

# values for making ticks in x and y axis
xnumbers = np.linspace(0, 7, 15)
ynumbers = np.linspace(-1, 1, 11)

plt.plot(x, y, 'r', x, z, 'g') # r, g - red, green colour
plt.xlabel("Angle in Radians")
plt.ylabel("Magnitude")
plt.title("Plot of some trigonometric functions")
plt.xticks(xnumbers)
plt.yticks(ynumbers)
plt.legend(['sine', 'cosine'])
plt.grid()
plt.axis([0, 6.5, -1.1, 1.1]) # [xstart, xend, ystart, yend]
plt.show()
```

- Pythonic

```
# Plotting tutorials in Python
# Adding Multiple plots by superimposition
# Good for plots sharing similar x, y limits
# Using multiple plot commands
# Much better and preferred than previous

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2.0*np.pi, 101)
y = np.sin(x)
z = np.cos(x)

# values for making ticks in x and y axis
xnumbers = np.linspace(0, 7, 15)
ynumbers = np.linspace(-1, 1, 11)

plt.plot(x, y, color='r', label='sin') # r - red colour
plt.plot(x, z, color='g', label='cos') # g - green colour
plt.xlabel("Angle in Radians")
plt.ylabel("Magnitude")
plt.title("Plot of some trigonometric functions")
plt.xticks(xnumbers)
plt.yticks(ynumbers)
plt.legend()
plt.grid()
plt.axis([0, 6.5, -1.1, 1.1]) # [xstart, xend, ystart, yend]
plt.show()
```

XYtwinx

xy。 **twinx**。

```
# Plotting tutorials in Python
# Adding Multiple plots by twin x axis
# Good for plots having different y axis range
# Separate axes and figure objects
# replicate axes object and plot curves
# use axes to set attributes

# Note:
# Grid for second curve unsuccessful : let me know if you find it! :(

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2.0*np.pi, 101)
y = np.sin(x)
z = np.sinh(x)

# separate the figure object and axes object
# from the plotting object
fig, ax1 = plt.subplots()

# Duplicate the axes with a different y axis
# and the same x axis
ax2 = ax1.twinx() # ax2 and ax1 will have common x axis and different y axis
```

```
# plot the curves on axes 1, and 2, and get the curve handles
curve1, = ax1.plot(x, y, label="sin", color='r')
curve2, = ax2.plot(x, z, label="sinh", color='b')

# Make a curves list to access the parameters in the curves
curves = [curve1, curve2]

# add legend via axes 1 or axes 2 object.
# one command is usually sufficient
# ax1.legend() # will not display the legend of ax2
# ax2.legend() # will not display the legend of ax1
ax1.legend(curves, [curve.get_label() for curve in curves])
# ax2.legend(curves, [curve.get_label() for curve in curves]) # also valid

# Global figure properties
plt.title("Plot of sine and hyperbolic sine")
plt.show()
```

twinyYX

twinyyx。 。

```
# Plotting tutorials in Python
# Adding Multiple plots by twin y axis
# Good for plots having different x axis range
# Separate axes and figure objects
# replicate axes object and plot curves
# use axes to set attributes
```

```
import numpy as np
import matplotlib.pyplot as plt

y = np.linspace(0, 2.0*np.pi, 101)
x1 = np.sin(y)
x2 = np.sinh(y)

# values for making ticks in x and y axis
ynumbers = np.linspace(0, 7, 15)
xnumbers1 = np.linspace(-1, 1, 11)
xnumbers2 = np.linspace(0, 300, 7)

# separate the figure object and axes object
# from the plotting object
fig, ax1 = plt.subplots()

# Duplicate the axes with a different x axis
# and the same y axis
ax2 = ax1.twiny() # ax2 and ax1 will have common y axis and different x axis

# plot the curves on axes 1, and 2, and get the axes handles
curve1, = ax1.plot(x1, y, label="sin", color='r')
curve2, = ax2.plot(x2, y, label="sinh", color='b')

# Make a curves list to access the parameters in the curves
curves = [curve1, curve2]

# add legend via axes 1 or axes 2 object.
# one command is usually sufficient
# ax1.legend() # will not display the legend of ax2
# ax2.legend() # will not display the legend of ax1
# ax1.legend(curves, [curve.get_label() for curve in curves])
ax2.legend(curves, [curve.get_label() for curve in curves]) # also valid

# x axis labels via the axes
ax1.set_xlabel("Magnitude", color=curve1.get_color())
ax2.set_xlabel("Magnitude", color=curve2.get_color())

# y axis label via the axes
ax1.set_ylabel("Angle/Value", color=curve1.get_color())
# ax2.set_ylabel("Magnitude", color=curve2.get_color()) # does not work
# ax2 has no property control over y axis

# y ticks - make them coloured as well
ax1.tick_params(axis='y', colors=curve1.get_color())
# ax2.tick_params(axis='y', colors=curve2.get_color()) # does not work
# ax2 has no property control over y axis

# x axis ticks via the axes
ax1.tick_params(axis='x', colors=curve1.get_color())
ax2.tick_params(axis='x', colors=curve2.get_color())

# set x ticks
ax1.set_xticks(xnumbers1)
ax2.set_xticks(xnumbers2)

# set y ticks
ax1.set_yticks(ynumbers)
# ax2.set_yticks(ynumbers) # also works
```

```
# Grids via axes 1 # use this if axes 1 is used to
# define the properties of common x axis
# ax1.grid(color=curve1.get_color())

# To make grids using axes 2
ax1.grid(color=curve2.get_color())
ax2.grid(color=curve2.get_color())
ax1.xaxis.grid(False)

# Global figure properties
plt.title("Plot of sine and hyperbolic sine")
plt.show()
```

Matplotlib https://riptutorial.com/zh-TW/python/topic/10264/matplotlib

160:

Examples

。

```
class A(object):
    # func: A user-defined function object
    #
    # Note that func is a function object when it's defined,
    # and an unbound method object when it's retrieved.
    def func(self):
        pass

    # classMethod: A class method
    @classmethod
    def classMethod(self):
        pass

class B(object):
    # unboundMeth: A unbound user-defined method object
    #
    # Parent.func is an unbound user-defined method object here,
    # because it's retrieved.
    unboundMeth = A.func

a = A()
b = B()

print A.func
# output: <unbound method A.func>
print a.func
# output: <bound method A.func of <__main__.A object at 0x10e9ab910>>
print B.unboundMeth
# output: <unbound method A.func>
print b.unboundMeth
# output: <unbound method A.func>
print A.classMethod
# output: <bound method type.classMethod of <class '__main__.A'>>
print a.classMethod
# output: <bound method type.classMethod of <class '__main__.A'>>
```

;。

```
class A(object):
    # func: A user-defined function object
    #
    # Note that func is a function object when it's defined,
    # and an unbound method object when it's retrieved.
    def func(self):
        pass

    # classMethod: A class method
    @classmethod
    def classMethod(self):
        pass
```

```
class B(object):
    # unboundMeth: A unbound user-defined method object
    #
    # Parent.func is an unbound user-defined method object here,
    # because it's retrieved.
    unboundMeth = A.func

a = A()
b = B()

print A.func
# output: <unbound method A.func>
print a.func
# output: <bound method A.func of <__main__.A object at 0x10e9ab910>>
print B.unboundMeth
# output: <unbound method A.func>
print b.unboundMeth
# output: <unbound method A.func>
print A.classMethod
# output: <bound method type.classMethod of <class '__main__.A'>>
print a.classMethod
# output: <bound method type.classMethod of <class '__main__.A'>>
```

∞。

```
import turtle, time, random #tell python we need 3 different modules
turtle.speed(0) #set draw speed to the fastest
turtle.colormode(255) #special colormode
turtle.pensize(4) #size of the lines that will be drawn
def triangle(size): #This is our own function, in the parenthesis is a variable we have
defined that will be used in THIS FUNCTION ONLY. This fucntion creates a right triangle
    turtle.forward(size) #to begin this function we go forward, the amount to go forward by is
the variable size
    turtle.right(90) #turn right by 90 degree
    turtle.forward(size) #go forward, again with variable
    turtle.right(135) #turn right again
    turtle.forward(size * 1.5) #close the triangle. thanks to the Pythagorean theorem we know
that this line must be 1.5 times longer than the other two(if they are equal)
while(1): #INFINITE LOOP
    turtle.setpos(random.randint(-200, 200), random.randint(-200, 200)) #set the draw point to
a random (x,y) position
    turtle.pencolor(random.randint(1, 255), random.randint(1, 255), random.randint(1, 255))
#randomize the RGB color
    triangle(random.randint(5, 55)) #use our function, because it has only one variable we can
simply put a value in the parenthesis. The value that will be sent will be random between 5 -
55, end the end it really just changes ow big the triangle is.
    turtle.pencolor(random.randint(1, 255), random.randint(1, 255), random.randint(1, 255))
#randomize color again
```

https://riptutorial.com/zh-TW/python/topic/3965/

161:

yield (an_expression for x in an_iterator) ◦

- yield <expr>
- <expr>
- <var> = yield <expr>
- <iter>

Examples

◦ next() Python 3.x __next__() __iter__ ◦ ◦

```
# naive partial implementation of the Python 2.x xrange()
def xrange(n):
    i = 0
    while i < n:
        yield i
        i += 1

# looping
for i in xrange(10):
    print(i)  # prints the values 0, 1, ..., 9

# unpacking
a, b, c = xrange(3)  # 0, 1, 2

# building a list
l = list(xrange(10))  # [0, 1, ..., 9]
```

next

next() ◦

```
def nums():
    yield 1
    yield 2
    yield 3
generator = nums()

next(generator, None)  # 1
next(generator, None)  # 2
next(generator, None)  # 3
next(generator, None)  # None
next(generator, None)  # None
# ...
```

next(iterator[, default]) ◦ ◦ StopIteration ◦

send() ◦

```
def accumulator():
    total = 0
    value = None
    while True:
        # receive sent value
        value = yield total
        if value is None: break
        # aggregate values
        total += value

generator = accumulator()

# advance until the first "yield"
next(generator)      # 0

# from this point on, the generator aggregates values
generator.send(1)    # 1
generator.send(10)   # 11
generator.send(100)  # 111
# ...

# Calling next(generator) is equivalent to calling generator.send(None)
next(generator)      # StopIteration
```

- `next(generator)` `yieldtotal`0.。
- `generator.send(x)` `xyieldvalue` 。 。
- `next(generator)` `None`。 `NoneNone`。

。

```
generator = (i * 2 for i in range(3))

next(generator)  # 0
next(generator)  # 2
next(generator)  # 4
next(generator)  # raises StopIteration
```

。 。

```
generator = (i * 2 for i in range(3))

next(generator)  # 0
next(generator)  # 2
next(generator)  # 4
next(generator)  # raises StopIteration
```

Python `[0, 1, 2, 3]` 。

。 。

```
expression = (x**2 for x in range(10))
```

100x = 0。

yield returnreturn

```
expression = (x**2 for x in range(10))
```

```
expression = (x**2 for x in range(10))
```

```
expression = (x**2 for x in range(10))
```

```
expression = (x**2 for x in range(10))
```

```
expression = (x**2 for x in range(10))
```

function() print

list “”

```
expression = (x**2 for x in range(10))
```

next()

next() Python yield yield next() yield

Python yield StopIteration

```
expression = (x**2 for x in range(10))
```

Python 2 .next() Python 3 .__next__()

None

Fibonacci

```
def fib(a=0, b=1):
    """Generator that yields Fibonacci numbers. `a` and `b` are the seed values"""
    while True:
        yield a
```

```
        a, b = b, a + b

f = fib()
print(', '.join(str(next(f)) for _ in range(10)))
```

0,1,1,2,3,5,8,13,21,34

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

`itertools.count`。

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

list / set comprehensions with `range` `xrange` for python <3.0

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

`itertools.islice()`

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

“”

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

for。 break

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

-

```
def integers_starting_from(n):
    while True:
        yield n
        n += 1

natural_numbers = integers_starting_from(1)
```

Python 3.x 3.3

iterable`yield from`

```
def foob(x):
    yield from range(x * 2)
    yield from range(2)

list(foob(5))  # [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1]
```

。

```
def foob(x):
    yield from range(x * 2)
    yield from range(2)

list(foob(5))  # [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1]
```

```
# create and advance generator to the first yield
def coroutine(func):
    def start(*args,**kwargs):
        cr = func(*args,**kwargs)
        next(cr)
        return cr
    return start
```

```
# example coroutine
@coroutine
def adder(sum = 0):
    while True:
        x = yield sum
        sum += x

# example use
s = adder()
s.send(1) # 1
s.send(2) # 3
```

。 。

```
from os import listdir
from os.path import isfile, join, exists
```


```
from os import listdir
from os.path import isfile, join, exists
```

```
from os import listdir
from os.path import isfile, join, exists
```

```
from os import listdir
from os.path import isfile, join, exists
```


`yield from`

```
from os import listdir
from os.path import isfile, join, exists
```

`zip`

```
for x, y in zip(a,b):
    print(x,y)
```


```
for x, y in zip(a,b):
    print(x,y)
```


python 2`itertools.izip`。 `zip`。

iterableszip。 `itertools.zip_longest()` 。

```
def create():
    result = []
    # logic here...
    result.append(value) # possibly in several places
    # more logic...
    return result # possibly in several places

values = create()
```

```
def create():
    result = []
    # logic here...
    result.append(value) # possibly in several places
    # more logic...
    return result # possibly in several places

values = create()
```

yield from“”

```
def create():
    result = []
    # logic here...
    result.append(value) # possibly in several places
    # more logic...
    return result # possibly in several places

values = create()
```

next。 next。

```
def find_and_transform(sequence, predicate, func):
    for element in sequence:
        if predicate(element):
            return func(element)
    raise ValueError

item = find_and_transform(my_sequence, my_predicate, my_func)
```

```
def find_and_transform(sequence, predicate, func):
    for element in sequence:
        if predicate(element):
            return func(element)
    raise ValueError

item = find_and_transform(my_sequence, my_predicate, my_func)
```

first = next

```
def find_and_transform(sequence, predicate, func):
    for element in sequence:
        if predicate(element):
            return func(element)
    raise ValueError

item = find_and_transform(my_sequence, my_predicate, my_func)
```

https://riptutorial.com/zh-TW/python/topic/292/

162: urllib

Examples

HTTP GET

Python 2.x 2.7

Python 2

```
import urllib
response = urllib.urlopen('http://stackoverflow.com/documentation/')
```

`urllib.urlopen()`。

```
import urllib
response = urllib.urlopen('http://stackoverflow.com/documentation/')
```

`response.code`http。 200404NotFound。

```
import urllib
response = urllib.urlopen('http://stackoverflow.com/documentation/')
```

`response.read()` `response.readlines()`html。 `file.read*`

Python 3.x 3.0

Python 3

```
import urllib
response = urllib.urlopen('http://stackoverflow.com/documentation/')
```

Python 3.x。 `urllib.request.urlopen`。

HTTP POST

POSTurlopen

Python 2.x 2.7

Python 2

```
import urllib
query_parms = {'username':'stackoverflow', 'password':'me.me'}
```

```
encoded_parms = urllib.urlencode(query_parms)
response = urllib.urlopen("https://stackoverflow.com/users/login", encoded_parms)
response.code
# Output: 200
response.read()
# Output: '<!DOCTYPE html>\r\n<html>\r\n<head>\r\n\r\n<title>Log In - Stack Overflow'
```

Python 3.x 3.0

Python 3

```
import urllib
query_parms = {'username':'stackoverflow', 'password':'me.me'}
encoded_parms = urllib.urlencode(query_parms)
response = urllib.urlopen("https://stackoverflow.com/users/login", encoded_parms)
response.code
# Output: 200
response.read()
# Output: '<!DOCTYPE html>\r\n<html>\r\n<head>\r\n\r\n<title>Log In - Stack Overflow'
```

Python 3.x 3.0

```
import urllib.request

response = urllib.request.urlopen("http://stackoverflow.com/")
data = response.read()

encoding = response.info().get_content_charset()
html = data.decode(encoding)
```

Python 2.x 2.7

```
import urllib.request

response = urllib.request.urlopen("http://stackoverflow.com/")
data = response.read()

encoding = response.info().get_content_charset()
html = data.decode(encoding)
```

urllib https://riptutorial.com/zh-TW/python/topic/2645/urllib

163: WebSockets

Examples

aiohttp

`aiohttp`websockets。

Python 3.x 3.5

```
import asyncio
from aiohttp import ClientSession

with ClientSession() as session:
    async def hello_world():

        websocket = await session.ws_connect("wss://echo.websocket.org")

        websocket.send_str("Hello, world!")

        print("Received:", (await websocket.receive()).data)

        await websocket.close()

    loop = asyncio.get_event_loop()
    loop.run_until_complete(hello_world())
```

aiohttp

`aiohttp.ClientSession`WebSocket。

Python 3.x 3.5

```
import asyncio
from aiohttp import ClientSession

class EchoWebSocket(ClientSession):

    URL = "wss://echo.websocket.org"

    def __init__(self):
        super().__init__()
        self.websocket = None

    async def connect(self):
        """Connect to the WebSocket."""
        self.websocket = await self.ws_connect(self.URL)

    async def send(self, message):
        """Send a message to the WebSocket."""
        assert self.websocket is not None, "You must connect first!"
        self.websocket.send_str(message)
        print("Sent:", message)
```

```
    async def receive(self):
        """Receive one message from the WebSocket."""
        assert self.websocket is not None, "You must connect first!"
        return (await self.websocket.receive()).data

    async def read(self):
        """Read messages from the WebSocket."""
        assert self.websocket is not None, "You must connect first!"

        while self.websocket.receive():
            message = await self.receive()
            print("Received:", message)
            if message == "Echo 9!":
                break

async def send(websocket):
    for n in range(10):
        await websocket.send("Echo {}!".format(n))
        await asyncio.sleep(1)

loop = asyncio.get_event_loop()

with EchoWebSocket() as websocket:

    loop.run_until_complete(websocket.connect())

    tasks = (
        send(websocket),
        websocket.read()
    )

    loop.run_until_complete(asyncio.wait(tasks))

    loop.close()
```

Websocket

AutobahnPython Web。

Python Autobahn

terminal

Linux

```
sudo pip install autobahn
```

Windows

```
sudo pip install autobahn
```

Pythonecho

```
sudo pip install autobahn
```

9000localhost127.0.0.1。IP。LAN。WAN IP9000WebSocketWAN IP。

9000。

WebSockets https://riptutorial.com/zh-TW/python/topic/4751/websockets

164:

Python。 Python。

numbers

	numbers.Number	numbers.Integral	numbers.Rational	numbers.Real	nur
	✓	✓	✓	✓	✓
INT	✓	✓	✓	✓	✓
fractions.Fraction	✓	-	✓	✓	✓
	✓	-	-	✓	✓
	✓	-	-	-	✓
decimal.Decimal	✓	-	-	-	-

Examples

```
a, b = 1, 2

# Using the "+" operator:
a + b                   # = 3

# Using the "in-place" "+=" operator to add and assign:
a += b                  # a = 3 (equivalent to a = a + b)

import operator         # contains 2 argument arithmetic functions for the examples

operator.add(a, b)      # = 5  since a is set to 3 right before this line

# The "+=" operator is equivalent to:
a = operator.iadd(a, b)     # a = 5 since a is set to 3 right before this line
```

- intint int
- intfloat float
- intcomplex complex
- floatfloat float
- floatcomplex complex
- complexcomplex complex

+

```
a, b = 1, 2

# Using the "+" operator:
a + b                   # = 3

# Using the "in-place" "+=" operator to add and assign:
a += b                  # a = 3 (equivalent to a = a + b)

import operator         # contains 2 argument arithmetic functions for the examples

operator.add(a, b)      # = 5  since a is set to 3 right before this line

# The "+=" operator is equivalent to:
a = operator.iadd(a, b)    # a = 5 since a is set to 3 right before this line
```

```
a, b = 1, 2

# Using the "-" operator:
b - a                   # = 1


import operator         # contains 2 argument arithmetic functions
operator.sub(b, a)      # = 1
```

- intint int
- intfloat float
- intcomplex complex
- floatfloat float
- floatcomplex complex
- complexcomplex complex

```
a, b = 2, 3

a * b                   # = 6

import operator
operator.mul(a, b)      # = 6
```

- intint int
- intfloat float
- intcomplex complex
- floatfloat float
- floatcomplex complex
- complexcomplex complex

---

*

```
a, b = 2, 3

a * b                   # = 6
```

```
import operator
operator.mul(a, b)     # = 6
```

Python。 PythonPython 2.x3.x 。

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

Python 2.x 2.7

Python 2'/'。

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

`abint int` 。

。

`ca / cfloat` 。

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

Python 2.x 2.2

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

TypeError

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

Python 2.x 2.2

Python 2'//'。

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

Python 3.x 3.0

Python 3 `/`'true'。 `//`。

```
a, b, c, d, e = 3, 2, 2.0, -3, 10
```

- `int`int Python 2`int` Python 3`float`
- `int`float float
- `int`complex complex
- `float`float float

- floatcomplex complex
- complexcomplex complex

PEP 238 。

```
a, b = 2, 3

(a ** b)                # = 8
pow(a, b)               # = 8

import math
math.pow(a, b)          # = 8.0 (always float; does not allow complex results)

import operator
operator.pow(a, b)      # = 8
```

powmath.powpow


```
a, b = 2, 3

(a ** b)                # = 8
pow(a, b)               # = 8

import math
math.pow(a, b)          # = 8.0 (always float; does not allow complex results)

import operator
operator.pow(a, b)      # = 8
```


math.sqrt(x)x 。


```
a, b = 2, 3

(a ** b)                # = 8
pow(a, b)               # = 8

import math
math.pow(a, b)          # = 8.0 (always float; does not allow complex results)

import operator
operator.pow(a, b)      # = 8
```


。 。


```
a, b = 2, 3

(a ** b)                # = 8
pow(a, b)               # = 8

import math
math.pow(a, b)          # = 8.0 (always float; does not allow complex results)

import operator
operator.pow(a, b)      # = 8
```

math.exp(x)e ** x。

```
a, b = 2, 3

(a ** b)                # = 8
pow(a, b)               # = 8

import math
math.pow(a, b)          # = 8.0 (always float; does not allow complex results)

import operator
operator.pow(a, b)      # = 8
```

math.expm1(x)e ** x - 1。 xmath.exp(x) - 1。

```
a, b = 2, 3

(a ** b)                # = 8
pow(a, b)               # = 8

import math
math.pow(a, b)          # = 8.0 (always float; does not allow complex results)

import operator
operator.pow(a, b)      # = 8
```

math.log e。 base。

```
import math
import cmath

math.log(5)          # = 1.6094379124341003
# optional base argument. Default is math.e
math.log(5, math.e) # = 1.6094379124341003
cmath.log(5)         # = (1.6094379124341003+0j)
math.log(1000, 10)    # 3.0 (always returns float)
cmath.log(1000, 10)   # (3+0j)
```

math.log。

```
import math
import cmath

math.log(5)          # = 1.6094379124341003
# optional base argument. Default is math.e
math.log(5, math.e) # = 1.6094379124341003
cmath.log(5)         # = (1.6094379124341003+0j)
math.log(1000, 10)    # 3.0 (always returns float)
cmath.log(1000, 10)   # (3+0j)
```

```
a = a + 1
```

```
a = a + 1
```

```
a = a + 1
```

'='

- `-=`
- `+=`
- `*=`
- `/=`
- `//=` floorPython 3
- `%=`
- `**=`

`^` `|` etc

```
a, b = 1, 2

import math

math.sin(a)  # returns the sine of 'a' in radians
# Out: 0.8414709848078965

math.cosh(b)  # returns the inverse hyperbolic cosine of 'b' in radians
# Out: 3.7621956910836314

math.atan(math.pi)  # returns the arc tangent of 'pi' in radians
# Out: 1.2626272556789115

math.hypot(a, b) # returns the Euclidean norm, same as math.sqrt(a*a + b*b)
# Out: 2.23606797749979
```

`math.hypot(x, y)(0, 0)(x, y)`。

`(x1, y1)(x2, y2)(x2, y2)math.hypot`

```
a, b = 1, 2

import math

math.sin(a)  # returns the sine of 'a' in radians
# Out: 0.8414709848078965

math.cosh(b)  # returns the inverse hyperbolic cosine of 'b' in radians
# Out: 3.7621956910836314

math.atan(math.pi)  # returns the arc tangent of 'pi' in radians
# Out: 1.2626272556789115

math.hypot(a, b) # returns the Euclidean norm, same as math.sqrt(a*a + b*b)
# Out: 2.23606797749979
```

- > - >`math.degreesmath.radians`

```
a, b = 1, 2

import math
```

```
math.sin(a)  # returns the sine of 'a' in radians
# Out: 0.8414709848078965

math.cosh(b)  # returns the inverse hyperbolic cosine of 'b' in radians
# Out: 3.7621956910836314

math.atan(math.pi)  # returns the arc tangent of 'pi' in radians
# Out: 1.2626272556789115

math.hypot(a, b) # returns the Euclidean norm, same as math.sqrt(a*a + b*b)
# Out: 2.23606797749979
```

Python%。

```
3 % 4     # 3
10 % 2    # 0
6 % 4     # 2
```

operator

```
3 % 4     # 3
10 % 2    # 0
6 % 4     # 2
```

。

```
3 % 4     # 3
10 % 2    # 0
6 % 4     # 2
```

divmod

```
3 % 4     # 3
10 % 2    # 0
6 % 4     # 2
```

https://riptutorial.com/zh-TW/python/topic/298/

165:

- OBJ []
-
- []

Paramer	
obj	“”
start	objPythonobj0。 0。
stop	obj。 len(obj)。
step	step。 1。

unicode1。

[start, end)。 len(x[:n]) = nlen(x) > = n x[n:n+1]x[n:n+1] = [x[n]]xlen(x) >= n。

Examples

Python。

```
iterable_name[start:stop:step]
```

- start。 0
- stop。 len
- step

```
iterable_name[start:stop:step]
```

```
iterable_name[start:stop:step]
```

```
iterable_name[start:stop:step]
```

```
iterable_name[start:stop:step]
```

```
iterable_name[start:stop:step]
```

end_indexNone http://stackoverflow.com/a/12521981

```
iterable_name[start:stop:step]
```

```
arr[:]
```

◦ [:]start endslice ◦ 0 len(arr)1 arrarr◦

```
arr[:]
```

arr.append('d')darr copy

arr.copy()◦

str listtuple step◦

```
s = 'reverse me!'
s[::-1]    # '!em esrever'
```

◦ [::-1]startend -1◦

__ getitem ____ setitem____delitem__

```
class MultiIndexingList:
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return repr(self.value)

    def __getitem__(self, item):
        if isinstance(item, (int, slice)):
            return self.__class__(self.value[item])
        return [self.value[i] for i in item]

    def __setitem__(self, item, value):
        if isinstance(item, int):
            self.value[item] = value
        elif isinstance(item, slice):
            raise ValueError('Cannot interpret slice with multiindexing')
        else:
            for i in item:
                if isinstance(i, slice):
                    raise ValueError('Cannot interpret slice with multiindexing')
                self.value[i] = value

    def __delitem__(self, item):
        if isinstance(item, int):
            del self.value[item]
        elif isinstance(item, slice):
            del self.value[item]
        else:
            if any(isinstance(elem, slice) for elem in item):
                raise ValueError('Cannot interpret slice with multiindexing')
            item = sorted(item, reverse=True)
            for elem in item:
                del self.value[elem]
```

```
class MultiIndexingList:
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return repr(self.value)

    def __getitem__(self, item):
        if isinstance(item, (int, slice)):
            return self.__class__(self.value[item])
        return [self.value[i] for i in item]

    def __setitem__(self, item, value):
        if isinstance(item, int):
            self.value[item] = value
        elif isinstance(item, slice):
            raise ValueError('Cannot interpret slice with multiindexing')
        else:
            for i in item:
                if isinstance(i, slice):
                    raise ValueError('Cannot interpret slice with multiindexing')
                self.value[i] = value

    def __delitem__(self, item):
        if isinstance(item, int):
            del self.value[item]
        elif isinstance(item, slice):
            del self.value[item]
        else:
            if any(isinstance(elem, slice) for elem in item):
                raise ValueError('Cannot interpret slice with multiindexing')
            item = sorted(item, reverse=True)
            for elem in item:
                del self.value[elem]
```

```
class MultiIndexingList:
    def __init__(self, value):
        self.value = value

    def __repr__(self):
        return repr(self.value)

    def __getitem__(self, item):
        if isinstance(item, (int, slice)):
            return self.__class__(self.value[item])
        return [self.value[i] for i in item]

    def __setitem__(self, item, value):
        if isinstance(item, int):
            self.value[item] = value
        elif isinstance(item, slice):
            raise ValueError('Cannot interpret slice with multiindexing')
        else:
            for i in item:
                if isinstance(i, slice):
                    raise ValueError('Cannot interpret slice with multiindexing')
                self.value[i] = value

    def __delitem__(self, item):
        if isinstance(item, int):
```

```
            del self.value[item]
        elif isinstance(item, slice):
            del self.value[item]
        else:
            if any(isinstance(elem, slice) for elem in item):
                raise ValueError('Cannot interpret slice with multiindexing')
            item = sorted(item, reverse=True)
            for elem in item:
                del self.value[elem]
```

- Python

```
lst = [1, 2, 3]
lst[1:3] = [4, 5]
print(lst) # Out: [1, 4, 5]
```


```
lst = [1, 2, 3]
lst[1:3] = [4, 5]
print(lst) # Out: [1, 4, 5]
```

```
lst = [1, 2, 3]
lst[1:3] = [4, 5]
print(lst) # Out: [1, 4, 5]
```

```
lst = [1, 2, 3]
lst[1:3] = [4, 5]
print(lst) # Out: [1, 4, 5]
```


`slice()`

```
>>> programmer_1 = [ 1956, 'Guido', 'van Rossum', 'Python', 'Netherlands']
>>> programmer_2 = [ 1815, 'Ada', 'Lovelace', 'Analytical Engine', 'England']
>>> name_columns = slice(1, 3)
>>> programmer_1[name_columns]
['Guido', 'van Rossum']
>>> programmer_2[name_columns]
['Ada', 'Lovelace']
```

Python0 `0`

```
arr = ['a', 'b', 'c', 'd']
print(arr[0])
>> 'a'
```

`12`


```
arr = ['a', 'b', 'c', 'd']
print(arr[0])
>> 'a'
```


- index `-1`index `-2`

```
arr = ['a', 'b', 'c', 'd']
print(arr[0])
>> 'a'
```

IndexError

```
arr = ['a', 'b', 'c', 'd']
print(arr[0])
>> 'a'
```

https://riptutorial.com/zh-TW/python/topic/289/

166:

- empty_set = set
- literal_set = {'foo''bar''baz'}3
- set_from_list = set['foo''bar''baz']set
- set_from_iter = setxx30
- set_from_iter = {x for x in [random.randint0,10for i in range10]}alternative notation

O1。 。 。 。

。 frozenset 。

。 __hash____eq__。 listset。 dictimmutable。

Examples

- 。 。

```
restaurants = ["McDonald's", "Burger King", "McDonald's", "Chicken Chicken"]
unique_restaurants = set(restaurants)
print(unique_restaurants)
# prints {'Chicken Chicken', "McDonald's", 'Burger King'}
```

; dict。

Python listList

```
restaurants = ["McDonald's", "Burger King", "McDonald's", "Chicken Chicken"]
unique_restaurants = set(restaurants)
print(unique_restaurants)
# prints {'Chicken Chicken', "McDonald's", 'Burger King'}
```

```
restaurants = ["McDonald's", "Burger King", "McDonald's", "Chicken Chicken"]
unique_restaurants = set(restaurants)
print(unique_restaurants)
# prints {'Chicken Chicken', "McDonald's", 'Burger King'}
```

。

```
# Intersection
{1, 2, 3, 4, 5}.intersection({3, 4, 5, 6})  # {3, 4, 5}
{1, 2, 3, 4, 5} & {3, 4, 5, 6}              # {3, 4, 5}

# Union
{1, 2, 3, 4, 5}.union({3, 4, 5, 6})  # {1, 2, 3, 4, 5, 6}
{1, 2, 3, 4, 5} | {3, 4, 5, 6}       # {1, 2, 3, 4, 5, 6}

# Difference
{1, 2, 3, 4}.difference({2, 3, 5})  # {1, 4}
```

```
{1, 2, 3, 4} - {2, 3, 5}            # {1, 4}

# Symmetric difference with
{1, 2, 3, 4}.symmetric_difference({2, 3, 5})  # {1, 4, 5}
{1, 2, 3, 4} ^ {2, 3, 5}                      # {1, 4, 5}

# Superset check
{1, 2}.issuperset({1, 2, 3})  # False
{1, 2} >= {1, 2, 3}           # False

# Subset check
{1, 2}.issubset({1, 2, 3})  # True
{1, 2} <= {1, 2, 3}         # True

# Disjoint check
{1, 2}.isdisjoint({3, 4})  # True
{1, 2}.isdisjoint({1, 4})  # False
```

```
# Intersection
{1, 2, 3, 4, 5}.intersection({3, 4, 5, 6})  # {3, 4, 5}
{1, 2, 3, 4, 5} & {3, 4, 5, 6}              # {3, 4, 5}

# Union
{1, 2, 3, 4, 5}.union({3, 4, 5, 6})  # {1, 2, 3, 4, 5, 6}
{1, 2, 3, 4, 5} | {3, 4, 5, 6}       # {1, 2, 3, 4, 5, 6}

# Difference
{1, 2, 3, 4}.difference({2, 3, 5})  # {1, 4}
{1, 2, 3, 4} - {2, 3, 5}            # {1, 4}

# Symmetric difference with
{1, 2, 3, 4}.symmetric_difference({2, 3, 5})  # {1, 4, 5}
{1, 2, 3, 4} ^ {2, 3, 5}                      # {1, 4, 5}

# Superset check
{1, 2}.issuperset({1, 2, 3})  # False
{1, 2} >= {1, 2, 3}           # False

# Subset check
{1, 2}.issubset({1, 2, 3})  # True
{1, 2} <= {1, 2, 3}         # True

# Disjoint check
{1, 2}.isdisjoint({3, 4})  # True
{1, 2}.isdisjoint({1, 4})  # False
```

---

	s | = t	
	s= t	intersection_update
	s - = t	difference_update
symmetric_difference	s ^ = t	symmetric_difference_update

```
# Intersection
```

```
{1, 2, 3, 4, 5}.intersection({3, 4, 5, 6})  # {3, 4, 5}
{1, 2, 3, 4, 5} & {3, 4, 5, 6}              # {3, 4, 5}

# Union
{1, 2, 3, 4, 5}.union({3, 4, 5, 6})  # {1, 2, 3, 4, 5, 6}
{1, 2, 3, 4, 5} | {3, 4, 5, 6}       # {1, 2, 3, 4, 5, 6}

# Difference
{1, 2, 3, 4}.difference({2, 3, 5})  # {1, 4}
{1, 2, 3, 4} - {2, 3, 5}            # {1, 4}

# Symmetric difference with
{1, 2, 3, 4}.symmetric_difference({2, 3, 5})  # {1, 4, 5}
{1, 2, 3, 4} ^ {2, 3, 5}                      # {1, 4, 5}

# Superset check
{1, 2}.issuperset({1, 2, 3})  # False
{1, 2} >= {1, 2, 3}           # False

# Subset check
{1, 2}.issubset({1, 2, 3})  # True
{1, 2} <= {1, 2, 3}         # True

# Disjoint check
{1, 2}.isdisjoint({3, 4})  # True
{1, 2}.isdisjoint({1, 4})  # False
```

。 。

```
>>> setA = {'a','b','b','c'}
>>> setA
set(['a', 'c', 'b'])
```

'a' 'b' 'b' 'c''b'。

```
>>> setA = {'a','b','b','c'}
>>> setA
set(['a', 'c', 'b'])
```

。

Python`collectionsCounter`2.7

Python 2.x 2.7

```
>>> setA = {'a','b','b','c'}
>>> setA
set(['a', 'c', 'b'])
```

`Counter`。 。

MethodsBuiltins

ab

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

{1}{}dict ◦ set() ◦

---

a.intersection(b)ab

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

---

a.union(b)ab

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

---

a.difference(b)ab

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

---

a.symmetric_difference(b)ab

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

a.symmetric_difference(b) == b.symmetric_difference(a)

---

c.issubset(a)ca ◦

a.issuperset(c)ca ◦

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

a.intersection(b)	a & b
a.union(b)	a | b
a.difference(b)	a - b
a.symmetric_difference(b)	a ^ b
a.issubset(b)	a <= b
a.issuperset(b)	a >= b

`ad ad。`

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

`in`occurances

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

`len()`

```
>>> a = {1, 2, 2, 3, 4}
>>> b = {3, 3, 4, 4, 5}
```

```
{{1,2}, {3,4}}
```

```
{{1,2}, {3,4}}
```

`frozenset`

```
{{1,2}, {3,4}}
```

https://riptutorial.com/zh-TW/python/topic/497/

167:

Examples

unicode。 。

```
>>> u'□'.encode('utf-8')
'\xf0\x9f\x90\x8d'
```

unicode。 。

```
>>> u'□'.encode('utf-8')
'\xf0\x9f\x90\x8d'
```

https://riptutorial.com/zh-TW/python/topic/5618/

168:

Examples

◦ IndentationError◦ IndentationError

```
a = 7
if a > 5:
  print "foo"
else:
  print "bar"
 print "done"
```

IndentationError

```
a = 7
if a > 5:
  print "foo"
else:
  print "bar"
 print "done"
```

IndentationError

```
a = 7
if a > 5:
  print "foo"
else:
  print "bar"
 print "done"
```

◦ NoneFalse

```
a = 7
if a > 5:
  print "foo"
else:
  print "bar"
 print "done"
```

PythonGuido van Rossum◦ “Python”◦ :

```
class ExampleClass:
    #Every function belonging to a class must be indented equally
    def __init__(self):
        name = "example"

    def someFunction(self, a):
        #Notice everything belonging to a function must be indented
        if a > 5:
            return True
```

```
        else:
            return False

#If a function is not indented to the same level it will not be considers as part of the
parent class
def separateFunction(b):
    for i in b:
    #Loops are also indented and nested conditions start a new indentation
        if i == 1:
            return True
    return False

separateFunction([2,3,5,6,1])
```

4。 ***Python***Python 3Python 2。

。

。 0。 “INDENT”。 “INDENT” `IndentationError` 。

。 “DEDENT”。 “DEDENT”。

“INDENT”“DEDENT”。

“DEDENT”0。

```
if foo:
    if bar:
        x = 42
else:
    print foo
```

```
if foo:
    if bar:
        x = 42
else:
    print foo
```

“INDENT”“DEDENT”。

https://riptutorial.com/zh-TW/python/topic/2597/

169:

Stack Overflow `"ImportError: No module named '??????' SyntaxError: invalid syntaxNameError: name '???' is not defined` 。 。

Examples

IndentationErrors

PythonFORTRANMakefilesWhitespace。

IndentationError / SyntaxError

。

Python 2.x 2.0 2.7

```
print "This line is ok"
    print "This line isn't ok"
```

Python 3.x 3.0

```
print "This line is ok"
    print "This line isn't ok"
```

。

Python 2.x 2.0 2.7

```
print "This line is ok"
    print "This line isn't ok"
```

Python 3.x 3.0

```
print "This line is ok"
    print "This line isn't ok"
```

IndentationError / SyntaxErrorunindent

。

Python 2.x 2.0 2.7

```
print "This line is ok"
```

```
    print "This line isn't ok"
```

Python 3.x 3.0

```
print "This line is ok"
    print "This line isn't ok"
```

IndentationError

。 。

```
print "This line is ok"
    print "This line isn't ok"
```

`pass` `if` `else` `except` `class` `methoddefinition`/`except`

```
print "This line is ok"
    print "This line isn't ok"
```

IndentationError

```
print "This line is ok"
    print "This line isn't ok"
```

。 `PEP8`Python。

1. 4。
2. 4。
3. 8。

。

TypeErrors

TypeError[/]

。

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

`*args**kwargs` ◦ * args** kwargs

TypeError[]'???''???'

◦

`+`◦ + ing `'set1''tuple1'setset`◦

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

`intstring` +

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

`intfloat`

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

TypeError'???' object/

`IndexError` `__iter____iterator__` `__getitem__`◦

`bar`1

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

`forxamount[0]` amountint

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

TypeError'???'

```
def foo(a): return a
foo(a,b,c,d) #And a,b,c,d are defined
```

NameErrorname'???'

◦ ◦ import◦ ◦ ◦

```
foo   # This variable is not defined
bar() # This function is not defined
```

```
foo   # This variable is not defined
bar() # This function is not defined
```

`import`

```
foo   # This variable is not defined
bar() # This function is not defined
```

PythonLEGB

LEGBPython◦

```
foo   # This variable is not defined
bar() # This function is not defined
```

- **L** ocal◦
- **E** nclosing◦
- **G** lobal◦
- **B** uilt-in◦

```
foo   # This variable is not defined
bar() # This function is not defined
```

`dfor`

```
foo   # This variable is not defined
bar() # This function is not defined
```

Python`NameError: name 'd' is not defined`

AssertError

assert。

```
assert condition
```

```
assert condition
```

```
assert condition
```

。

debugTrueFalse-O。 。 。

KeyboardInterrupt

Ctrl + Cdel 。

ZeroDivisionError

1/0 。

Python 2.x 2.0 2.7

```
assert condition
```

Python 3.x 3.0

```
assert condition
```

ZeroDivisionErrorforx 。

Python 2.x 2.0 2.7

```
assert condition
```

Python 3.x 3.0

```
assert condition
```

SyntaxError

```
def my_print():
    x = (1 + 1
```

```
    print(x)
```

```
def my_print():
    x = (1 + 1
    print(x)
```

/。

Python 3

Python 3.x 3.0

```
def my_print():
    x = (1 + 1
    print(x)
```

`printprint()`

```
def my_print():
    x = (1 + 1
    print(x)
```

https://riptutorial.com/zh-TW/python/topic/9300/

170:

Python。 “Project X1.xProject Y4.x”。

。

。

1. root
2.
3. python

Examples

`virtualenv`Python。 Python。

virtualenv

。 `virtualenv`。 Debian`python-virtualenvpython3-virtualenv`。

pip`virtualenv`

```
$ pip install virtualenv
```

。

```
$ pip install virtualenv
```

`pythonfoo`。 。 `virtualenv`。 。

---

shellPython`foo`。 `activate`shell

```
$ pip install virtualenv
```

Windows

```
$ pip install virtualenv
```

`pythonpipfoo`。 `pip`。 。

```
$ pip install virtualenv
```

pipfreeze。 installPython。 requirements.txt

```
$ pip install virtualenv
```

freeze。 requirements.txt 。

---

shell

```
$ pip install virtualenv
```

$ source bin/activate virtualenvmod_wsgiAmazon API GatewayGoogle AppEngine。 virtualenv sys.path 。

virtualenvsys.pathsys.prefix

```
$ pip install virtualenv
```

。

virtualenvbin/activate_this.py lib/python2.7/site-packagessys.path

activate_this.pybinlib/python2.7/site-packages。

Python 3.x 3.3

Python 3.3 venv。 pyvenv

```
$ pip install virtualenv
```

```
$ pip install virtualenv
```

virtualenv 。 root。

virtualenv

```
(<Virtualenv Name) $ which python
/<Virtualenv Directory>/bin/python

(Virtualenv Name) $ which pip
/<Virtualenv Directory>/bin/pip
```

pipvirtualenv

```
(<Virtualenv Name) $ which python
```

```
/<Virtualenv Directory>/bin/python

(Virtualenv Name) $ which pip
/<Virtualenv Directory>/bin/pip
```

。

requirements.txt

```
(<Virtualenv Name) $ which python
/<Virtualenv Directory>/bin/python

(Virtualenv Name) $ which pip
/<Virtualenv Directory>/bin/pip
```

```
(<Virtualenv Name) $ which python
/<Virtualenv Directory>/bin/python

(Virtualenv Name) $ which pip
/<Virtualenv Directory>/bin/pip
```

2.10.0`requests`。

```
(<Virtualenv Name) $ which python
/<Virtualenv Directory>/bin/python

(Virtualenv Name) $ which pip
/<Virtualenv Directory>/bin/pip
```

。 pip。

```
(<Virtualenv Name) $ which python
/<Virtualenv Directory>/bin/python

(Virtualenv Name) $ which pip
/<Virtualenv Directory>/bin/pip
```

pipPIP 。

- 。

python

`pythonpython3` `python3`PythonPython 3

```
virtualenv -p python3 foo
```

```
virtualenv -p python3 foo
```

```
virtualenv -p python3 foo
```

```
virtualenv -p python3 foo
```

python。 /usr/bin//usr/local/bin/ Linux/Library/Frameworks/Python.framework/Versions/XX/bin/ python--python-p。

virtualenvwrapper

virtualenvwrapper/。

virtualenvwrapper~/.virtualenvs 。

virtualenvwrapper 。

Debian / Ubuntu

```
apt-get install virtualenvwrapper
```

Fedora/ CentOS/ RHEL

```
apt-get install virtualenvwrapper
```

Arch Linux

```
apt-get install virtualenvwrapper
```

pipPyPI

```
apt-get install virtualenvwrapper
```

Windowsvirtualenvwrapper-winvirtualenvwrapper-powershell。

mkvirtualenv。 virtualenv。

```
apt-get install virtualenvwrapper
```

```
apt-get install virtualenvwrapper
```

。 shellworkon

```
apt-get install virtualenvwrapper
```

workon. path/to/my-env/bin/activate workon;。

-asetvirtualenvproject。

```
apt-get install virtualenvwrapper
```

```
apt-get install virtualenvwrapper
```

`workoncdproject`。

virtualenvwrappervirtualenv`lsvirtualenv`。

virtualenv`rmvirtualenv`

```
apt-get install virtualenvwrapper
```

virtualenvwrappervirtualenv4bash `preactivate postactivate predeactivatepostdeactivate`。
virtualenvbash; `postactivate`virtualenv。 。 4.`virtualenvs/<virtualenv_name>/bin/`。

virtualenvwrapper。

Linux`bash`。

```
(my-project-env) user@hostname:~$ which python
/home/user/my-project-env/bin/python
```

Unix / Linuxpython

Linux shellpythonPythonshebang`#!#!`

```
#!/usr/bin/python
```

`python myscript.py`Python.`/myscript.py#!#!`。 Python

```
#!/usr/bin/python
```

shebang

```
#!/usr/bin/python
```

`./myscript.py` `python myscript.pypython3 myscript.py`。

virtualenv

`virtualenvvirtualenvwrapper`。 `virtualfish`。 virtualenvFish shell。

- virtualfish

  ```
  sudo pip install virtualfish
  ```

- fish shellpythonvirtualfish

```
sudo pip install virtualfish
```

- `fish_prompt$ funced fish_prompt --editor vim`vim

```
sudo pip install virtualfish
```

vim`$ funced fish_prompt --editor nano$ funced fish_prompt --editor gedit`

- `funcsave`

```
sudo pip install virtualfish
```

- `vf new`

```
sudo pip install virtualfish
```

- python3`-p`

```
sudo pip install virtualfish
```

- `vf deactivate vf activate another_env`

- https://github.com/adambrenecki/virtualfish
- http://virtualfish.readthedocs.io/en/latest/

Anaconda

`virtualenv`Anaconda `pip`。 Anaconda

```
conda create --name <envname> python=<version>
```

`<envname> <version>`Python。

```
conda create --name <envname> python=<version>
```

```
conda create --name <envname> python=<version>
```

```
conda create --name <envname> python=<version>
```

```
conda create --name <envname> python=<version>
```

conda。

shell。

python

```
import sys
sys.prefix
sys.real_prefix
```

- `sys.prefix`python`sys.real_prefix` 。
- `sys.prefix`python `sys.real_prefix`python。

venv`sys.real_prefix` 。 `sys.base_prefix`sys.prefix。

https://riptutorial.com/zh-TW/python/topic/868/

171: virtualenvwrapper

ABC.ABpython 3。 Cpython 2.7。

。

VirtualenvVirtualenvwrapperConda

virtualenvwrapper。

Examples

virtualenvwrapper

ABC.ABpython 3。 Cpython 2.7。

。

VirtualenvVirtualenvwrapperConda

virtualenvwrapper。

virtualenvwrapper。

```
$ pip install virtualenvwrapper

$ export WORKON_HOME=~/Envs
$ mkdir -p $WORKON_HOME
$ source /usr/local/bin/virtualenvwrapper.sh
$ printf '\n%s\n%s\n%s' '# virtualenv' 'export WORKON_HOME=~/virtualenvs' 'source
/home/salayhin/bin/virtualenvwrapper.sh' >> ~/.bashrc
$ source ~/.bashrc

$ mkvirtualenv python_3.5
Installing
setuptools.......................................
..................................................
..................................................
...............................done.
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/predeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postdeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/preactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postactivate New
python executable in python_3.5/bin/python

(python_3.5)$ ls $WORKON_HOME
python_3.5 hook.log
```

。

```
$ pip install virtualenvwrapper
```

```
$ export WORKON_HOME=~/Envs
$ mkdir -p $WORKON_HOME
$ source /usr/local/bin/virtualenvwrapper.sh
$ printf '\n%s\n%s\n%s' '# virtualenv' 'export WORKON_HOME=~/virtualenvs' 'source
/home/salayhin/bin/virtualenvwrapper.sh' >> ~/.bashrc
$ source ~/.bashrc

$ mkvirtualenv python_3.5
Installing
setuptools.........................................
...................................................
...................................................
................................done.
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/predeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postdeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/preactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postactivate New
python executable in python_3.5/bin/python

(python_3.5)$ ls $WORKON_HOME
python_3.5 hook.log
```

lssitepackages

```
$ pip install virtualenvwrapper

$ export WORKON_HOME=~/Envs
$ mkdir -p $WORKON_HOME
$ source /usr/local/bin/virtualenvwrapper.sh
$ printf '\n%s\n%s\n%s' '# virtualenv' 'export WORKON_HOME=~/virtualenvs' 'source
/home/salayhin/bin/virtualenvwrapper.sh' >> ~/.bashrc
$ source ~/.bashrc

$ mkvirtualenv python_3.5
Installing
setuptools.........................................
...................................................
...................................................
................................done.
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/predeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postdeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/preactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postactivate New
python executable in python_3.5/bin/python

(python_3.5)$ ls $WORKON_HOME
python_3.5 hook.log
```

。

workon

```
$ pip install virtualenvwrapper

$ export WORKON_HOME=~/Envs
$ mkdir -p $WORKON_HOME
$ source /usr/local/bin/virtualenvwrapper.sh
```

```
$ printf '\n%s\n%s\n%s' '# virtualenv' 'export WORKON_HOME=~/virtualenvs' 'source
/home/salayhin/bin/virtualenvwrapper.sh' >> ~/.bashrc
$ source ~/.bashrc

$ mkvirtualenv python_3.5
Installing
setuptools.........................................
...................................................
...................................................
...............................done.
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/predeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postdeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/preactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postactivate New
python executable in python_3.5/bin/python

(python_3.5)$ ls $WORKON_HOME
python_3.5 hook.log
```

virtualenv

```
$ pip install virtualenvwrapper

$ export WORKON_HOME=~/Envs
$ mkdir -p $WORKON_HOME
$ source /usr/local/bin/virtualenvwrapper.sh
$ printf '\n%s\n%s\n%s' '# virtualenv' 'export WORKON_HOME=~/virtualenvs' 'source
/home/salayhin/bin/virtualenvwrapper.sh' >> ~/.bashrc
$ source ~/.bashrc

$ mkvirtualenv python_3.5
Installing
setuptools.........................................
...................................................
...................................................
...............................done.
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/predeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postdeactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/preactivate
virtualenvwrapper.user_scripts Creating /Users/salayhin/Envs/python_3.5/bin/postactivate New
python executable in python_3.5/bin/python

(python_3.5)$ ls $WORKON_HOME
python_3.5 hook.log
```

virtualenvwrapper https://riptutorial.com/zh-TW/python/topic/9983/virtualenvwrapper

172:

。 。 。 Python。

- def decorator_functionfpassdecorator_function
- @decorator_function
  def ..._functiondecorator_function
- decorated_function = decorator_functiondecorated_function`@decorator_function`

F

Examples

。 。

```
# This simplest decorator does nothing to the function being decorated. Such
# minimal decorators can occasionally be used as a kind of code markers.
def super_secret_function(f):
    return f

@super_secret_function
def my_function():
    print("This is my secret function.")
```

`@` -notation

```
# This simplest decorator does nothing to the function being decorated. Such
# minimal decorators can occasionally be used as a kind of code markers.
def super_secret_function(f):
    return f

@super_secret_function
def my_function():
    print("This is my secret function.")
```

。 “unsugared”。

```
# This simplest decorator does nothing to the function being decorated. Such
# minimal decorators can occasionally be used as a kind of code markers.
def super_secret_function(f):
    return f

@super_secret_function
def my_function():
    print("This is my secret function.")
```

。 。 。

```
# This simplest decorator does nothing to the function being decorated. Such
# minimal decorators can occasionally be used as a kind of code markers.
def super_secret_function(f):
    return f

@super_secret_function
def my_function():
    print("This is my secret function.")
```

◦ `my_func = decorator(my_func)` ◦ decoratormy_funcdecorator◦ `__call__()`my_func

```
class Decorator(object):
    """Simple decorator class."""

    def __init__(self, func):
        self.func = func

    def __call__(self, *args, **kwargs):
        print('Before the function call.')
        res = self.func(*args, **kwargs)
        print('After the function call.')
        return res

@Decorator
def testfunc():
    print('Inside the function.')

testfunc()
# Before the function call.
# Inside the function.
# After the function call.
```

“”

```
class Decorator(object):
    """Simple decorator class."""

    def __init__(self, func):
        self.func = func

    def __call__(self, *args, **kwargs):
        print('Before the function call.')
        res = self.func(*args, **kwargs)
        print('After the function call.')
        return res

@Decorator
def testfunc():
    print('Inside the function.')

testfunc()
# Before the function call.
# Inside the function.
# After the function call.
```

`__get__` -method

```
class Decorator(object):
    """Simple decorator class."""

    def __init__(self, func):
        self.func = func

    def __call__(self, *args, **kwargs):
        print('Before the function call.')
        res = self.func(*args, **kwargs)
        print('After the function call.')
        return res

@Decorator
def testfunc():
    print('Inside the function.')

testfunc()
# Before the function call.
# Inside the function.
# After the function call.
```

。

```
class Decorator(object):
    """Simple decorator class."""

    def __init__(self, func):
        self.func = func

    def __call__(self, *args, **kwargs):
        print('Before the function call.')
        res = self.func(*args, **kwargs)
        print('After the function call.')
        return res

@Decorator
def testfunc():
    print('Inside the function.')

testfunc()
# Before the function call.
# Inside the function.
# After the function call.
```

。 。 。 functools.wraps。

```
from functools import wraps
```

。 。

```
from functools import wraps
```

"

```
from functools import wraps
```

'。'

。 。

。 。

```
def decoratorfactory(message):
    def decorator(func):
        def wrapped_func(*args, **kwargs):
            print('The decorator wants to tell you: {}'.format(message))
            return func(*args, **kwargs)
        return wrapped_func
    return decorator

@decoratorfactory('Hello World')
def test():
    pass

test()
```

Hello World

```
def decoratorfactory(message):
    def decorator(func):
        def wrapped_func(*args, **kwargs):
            print('The decorator wants to tell you: {}'.format(message))
            return func(*args, **kwargs)
        return wrapped_func
    return decorator

@decoratorfactory('Hello World')
def test():
    pass

test()
```

TypeErrordecorator1'func'

```
def decoratorfactory(message):
    def decorator(func):
        def wrapped_func(*args, **kwargs):
            print('The decorator wants to tell you: {}'.format(message))
            return func(*args, **kwargs)
        return wrapped_func
    return decorator

@decoratorfactory('Hello World')
```

```
def test():
    pass

test()
```

10

/。 。

```
def singleton(cls):
    instance = [None]
    def wrapper(*args, **kwargs):
        if instance[0] is None:
            instance[0] = cls(*args, **kwargs)
        return instance[0]

    return wrapper
```

。 。

```
def singleton(cls):
    instance = [None]
    def wrapper(*args, **kwargs):
        if instance[0] is None:
            instance[0] = cls(*args, **kwargs)
        return instance[0]

    return wrapper
```

“”。

```
import time
def timer(func):
    def inner(*args, **kwargs):
        t1 = time.time()
        f = func(*args, **kwargs)
        t2 = time.time()
        print 'Runtime took {0} seconds'.format(t2-t1)
        return f
    return inner

@timer
def example_function():
    #do stuff


example_function()
```

https://riptutorial.com/zh-TW/python/topic/229/

173:

Examples

。

```
>>> import copy
>>> c = [[1,2]]
>>> d = copy.copy(c)
>>> c is d
False
>>> c[0] is d[0]
True
```

。 。

```
>>> import copy
>>> c = [[1,2]]
>>> d = copy.deepcopy(c)
>>> c is d
False
>>> c[0] is d[0]
False
```

。

```
>>> l1 = [1,2,3]
>>> l2 = l1[:]      # Perform the shallow copy.
>>> l2
[1,2,3]
>>> l1 is l2
False
```

copy 。 。

```
>>> d1 = {1:[]}
>>> d2 = d1.copy()
>>> d1 is d2
False
>>> d1[1] is d2[1]
True
```

copy。 。

```
>>> s1 = {()}
>>> s2 = s1.copy()
>>> s1 is s2
False
>>> s2.add(3)
>>> s1
{[]}
```

```
>>> s2
{3,[]}
```

https://riptutorial.com/zh-TW/python/topic/920/

174:

- cmath.rectAbsoluteValuePhase

Examples

cmath。

```
import cmath
```

```
import cmath
```

```
import cmath
```

- log log10

  ```
  import cmath
  ```

- ```
  import cmath
  ```

- ```
  import cmath
  ```

- ```
  import cmath
  ```

Python。 j

```
z = 2+3j # A complex number
w = 1-7j # Another complex number
```

```
z = 2+3j # A complex number
w = 1-7j # Another complex number
```

Python

```
z = 2+3j # A complex number
w = 1-7j # Another complex number
```

https://riptutorial.com/zh-TW/python/topic/1142/
```

175:

tarballZIPgzipPythontarfilezipfilegzip。 Pythontarfile`TarFile.extractall(path=".", members=None)` tarball。 Pythonzipfile`ZipFile.extractall([path[, members[, pwd]]])`ZIP。 PythongzipGzipFile。

Examples

Python ZipFile.extractallZIP

```
file_unzip = 'filename.zip'
unzip = zipfile.ZipFile(file_unzip, 'r')
unzip.extractall()
unzip.close()
```

Python TarFile.extractalltarball

```
file_untar = 'filename.tar.gz'
untar = tarfile.TarFile(file_untar)
untar.extractall()
untar.close()
```

https://riptutorial.com/zh-TW/python/topic/9505/

176:

。 。 Python。 Python。

Examples

argparseHello

。 。

```
import argparse

parser = argparse.ArgumentParser()

parser.add_argument('name',
    help='name of user'
)

parser.add_argument('-g', '--greeting',
    default='Hello',
    help='optional alternate greeting'
)

args = parser.parse_args()

print("{greeting}, {name}!".format(
       greeting=args.greeting,
       name=args.name)
)
```

```
import argparse

parser = argparse.ArgumentParser()

parser.add_argument('name',
    help='name of user'
)

parser.add_argument('-g', '--greeting',
    default='Hello',
    help='optional alternate greeting'
)

args = parser.parse_args()

print("{greeting}, {name}!".format(
       greeting=args.greeting,
       name=args.name)
)
```

```
import argparse

parser = argparse.ArgumentParser()
```

```
parser.add_argument('name',
    help='name of user'
)

parser.add_argument('-g', '--greeting',
    default='Hello',
    help='optional alternate greeting'
)

args = parser.parse_args()

print("{greeting}, {name}!".format(
       greeting=args.greeting,
       name=args.name)
)
```

argparse 。

docopt

docopt。 docopt。

```
"""
Usage:
    script_name.py [-a] [-b] <path>

Options:
    -a            Print all the things.
    -b            Get more bees into the path.
"""
from docopt import docopt


if __name__ == "__main__":
    args = docopt(__doc__)
    import pprint; pprint.pprint(args)
```

```
"""
Usage:
    script_name.py [-a] [-b] <path>

Options:
    -a            Print all the things.
    -b            Get more bees into the path.
"""
from docopt import docopt


if __name__ == "__main__":
    args = docopt(__doc__)
    import pprint; pprint.pprint(args)
```

argparse

- `argparse.ArgumentParser.add_mutually_exclusive_group()` 。 foobar。

```
import argparse

parser = argparse.ArgumentParser()
group = parser.add_mutually_exclusive_group()
group.add_argument("-f", "--foo")
group.add_argument("-b", "--bar")
args = parser.parse_args()
print "foo = ", args.foo
print "bar = ", args.bar
```

`--foo--bar`。

```
error: argument -b/--bar: not allowed with argument -f/--foo
```

argv

Python 。 `sys.argv`“argv”“ **arg** ument **v** ector”。

`sys.argv`Python。

```
# cli.py
import sys
print(sys.argv)

$ python cli.py
=> ['cli.py']

$ python cli.py fizz
=> ['cli.py', 'fizz']

$ python cli.py fizz buzz
=> ['cli.py', 'fizz', 'buzz']
```

`argv`。 sys.argv。 。

```
# cli.py
import sys
print(sys.argv)

$ python cli.py
=> ['cli.py']

$ python cli.py fizz
=> ['cli.py', 'fizz']

$ python cli.py fizz buzz
=> ['cli.py', 'fizz', 'buzz']
```

“”`sys.argv`。

```
# cli.py
import sys
print(sys.argv)

$ python cli.py
```

```
=> ['cli.py']

$ python cli.py fizz
=> ['cli.py', 'fizz']

$ python cli.py fizz buzz
=> ['cli.py', 'fizz', 'buzz']
```

argparse

◦ `argparse.ArgumentParser.error`◦ `--foo--barstderr`◦

```
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("-f", "--foo")
parser.add_argument("-b", "--bar")
args = parser.parse_args()
if args.foo and args.bar is None:
    parser.error("--foo requires --bar. You did not specify bar.")

print "foo =", args.foo
print "bar =", args.bar
```

sample.py `python sample.py --foo ds_in_fridge`

```
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("-f", "--foo")
parser.add_argument("-b", "--bar")
args = parser.parse_args()
if args.foo and args.bar is None:
    parser.error("--foo requires --bar. You did not specify bar.")

print "foo =", args.foo
print "bar =", args.bar
```

argparse.add_argument_group

argparse ArgumentParser'-h'◦ example.py`python example.py -h`◦

```
import argparse

parser = argparse.ArgumentParser(description='Simple example')
parser.add_argument('name', help='Who to greet', default='World')
parser.add_argument('--bar_this')
parser.add_argument('--bar_that')
parser.add_argument('--foo_this')
parser.add_argument('--foo_that')
args = parser.parse_args()
```

```
import argparse

parser = argparse.ArgumentParser(description='Simple example')
```

```
parser.add_argument('name', help='Who to greet', default='World')
parser.add_argument('--bar_this')
parser.add_argument('--bar_that')
parser.add_argument('--foo_this')
parser.add_argument('--foo_that')
args = parser.parse_args()
```

。 。 `--foo_*` args`--bar_*` args。

```
import argparse

parser = argparse.ArgumentParser(description='Simple example')
parser.add_argument('name', help='Who to greet', default='World')
parser.add_argument('--bar_this')
parser.add_argument('--bar_that')
parser.add_argument('--foo_this')
parser.add_argument('--foo_that')
args = parser.parse_args()
```

`python example.py -h`

```
import argparse

parser = argparse.ArgumentParser(description='Simple example')
parser.add_argument('name', help='Who to greet', default='World')
parser.add_argument('--bar_this')
parser.add_argument('--bar_that')
parser.add_argument('--foo_this')
parser.add_argument('--foo_that')
args = parser.parse_args()
```

docoptdocopt_dispatch

docopt[docopt_dispatch]`__doc__`--help 。 doc`dispatch` 。

if / else。

`dispatch.on`。

```
"""Run something in development or production mode.

Usage: run.py --development <host> <port>
       run.py --production <host> <port>
       run.py items add <item>
       run.py items delete <item>

"""
from docopt_dispatch import dispatch

@dispatch.on('--development')
def development(host, port, **kwargs):
    print('in *development* mode')

@dispatch.on('--production')
def development(host, port, **kwargs):
    print('in *production* mode')
```

```
@dispatch.on('items', 'add')
def items_add(item, **kwargs):
    print('adding item...')

@dispatch.on('items', 'delete')
def items_delete(item, **kwargs):
    print('deleting item...')

if __name__ == '__main__':
    dispatch(__doc__)
```

https://riptutorial.com/zh-TW/python/topic/1382/

177:

Examples

Python

。

APIPython。

```
import logging

logger = logging.getLogger()
handler = logging.StreamHandler()
formatter = logging.Formatter(
        '%(asctime)s %(name)-12s %(levelname)-8s %(message)s')
handler.setFormatter(formatter)
logger.addHandler(handler)
logger.setLevel(logging.DEBUG)

logger.debug('this is a %s test', 'debug')
```


```
import logging

logger = logging.getLogger()
handler = logging.StreamHandler()
formatter = logging.Formatter(
        '%(asctime)s %(name)-12s %(levelname)-8s %(message)s')
handler.setFormatter(formatter)
logger.addHandler(handler)
logger.setLevel(logging.DEBUG)

logger.debug('this is a %s test', 'debug')
```


INI

logging_config.ini。 。


```
import logging

logger = logging.getLogger()
handler = logging.StreamHandler()
formatter = logging.Formatter(
        '%(asctime)s %(name)-12s %(levelname)-8s %(message)s')
handler.setFormatter(formatter)
logger.addHandler(handler)
logger.setLevel(logging.DEBUG)

logger.debug('this is a %s test', 'debug')
```


`logging.config.fileConfig()`

```
import logging

logger = logging.getLogger()
handler = logging.StreamHandler()
formatter = logging.Formatter(
        '%(asctime)s %(name)-12s %(levelname)-8s %(message)s')
handler.setFormatter(formatter)
logger.addHandler(handler)
logger.setLevel(logging.DEBUG)

logger.debug('this is a %s test', 'debug')
```

Python 2.7。 PEP 391。

```
import logging

logger = logging.getLogger()
handler = logging.StreamHandler()
formatter = logging.Formatter(
        '%(asctime)s %(name)-12s %(levelname)-8s %(message)s')
handler.setFormatter(formatter)
logger.addHandler(handler)
logger.setLevel(logging.DEBUG)

logger.debug('this is a %s test', 'debug')
```

`logging.exception(msg)`

```
>>> import logging
>>> logging.basicConfig()
>>> try:
...     raise Exception('foo')
... except:
...     logging.exception('bar')
...
ERROR:root:bar
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
Exception: foo
```

`logging.exception(msg)` `msg` arg

```
>>> import logging
>>> logging.basicConfig()
>>> try:
...     raise Exception('foo')
... except:
...     logging.exception('bar')
...
ERROR:root:bar
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
Exception: foo
```

```
>>> import logging
>>> logging.basicConfig()
```

```
>>> try:
...     raise Exception('foo')
... except:
...     logging.exception('bar')
...
ERROR:root:bar
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
Exception: foo
```

unicode。 `logging.exception(e)`。

。 。

python

```
>>> import logging
>>> logging.basicConfig()
>>> try:
...     raise Exception('foo')
... except:
...     logging.exception('bar')
...
ERROR:root:bar
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
Exception: foo
```

e `logging.exception(...)`。

ERROR

ERROR`exc_info`

```
>>> import logging
>>> logging.basicConfig()
>>> try:
...     raise Exception('foo')
... except:
...     logging.exception('bar')
...
ERROR:root:bar
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
Exception: foo
```

unicodeutf-8。 `repr(e)%r`

```
>>> import logging
>>> logging.basicConfig()
>>> try:
...     raise Exception('foo')
... except:
...     logging.exception('bar')
...
```

```
ERROR:root:bar
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
Exception: foo
```

https://riptutorial.com/zh-TW/python/topic/4081/

178: Python

Examples

PythonPythonpython

Python

```
import dis

def fib(n):
    if n <= 2: return 1
    return fib(n-1) + fib(n-2)

# Display the disassembled bytecode of the function.
dis.dis(fib)
```

dis`dis.dis`。

CPython。

`__code__` `co_code` 。

```
def fib(n):
    if n <= 2: return 1
    return fib(n-1) + fib(n-2)
dir(fib.__code__)

def fib(n):
    if n <= 2: return 1
    return fib(n-1) + fib(n-2)
dir(fib.__code__)
```

Python`inspect` 。 。 。

`randomrandint`

```
import random
import inspect

print(inspect.getsource(random.randint))
# Output:
#    def randint(self, a, b):
#        """Return random integer in range [a, b], including both end points.
#        """
#
#        return self.randrange(a, b+1)
```

```
import random
import inspect

print(inspect.getsource(random.randint))
```

```
# Output:
#    def randint(self, a, b):
#        """Return random integer in range [a, b], including both end points.
#        """
#
#        return self.randrange(a, b+1)
```

random.randint

```
import random
import inspect

print(inspect.getsource(random.randint))
# Output:
#    def randint(self, a, b):
#        """Return random integer in range [a, b], including both end points.
#        """
#
#        return self.randrange(a, b+1)
```

inspectdill.source.getsource

```
import random
import inspect

print(inspect.getsource(random.randint))
# Output:
#    def randint(self, a, b):
#        """Return random integer in range [a, b], including both end points.
#        """
#
#        return self.randrange(a, b+1)
```

Python**c**PythonMercurialhttps://www.python.org/downloads/source/ 。

```
import random
import inspect

print(inspect.getsource(random.randint))
# Output:
#    def randint(self, a, b):
#        """Return random integer in range [a, b], including both end points.
#        """
#
#        return self.randrange(a, b+1)
```

Python https://riptutorial.com/zh-TW/python/topic/4351/python

179:

- Python

Examples

。 。 。

“”。

```
from types import MethodType


class Animal(object):

    def __init__(self, *args, **kwargs):
        self.name = kwargs.pop('name', None) or 'Animal'
        if kwargs.get('walk', None):
            self.walk = MethodType(kwargs.pop('walk'), self)

    def walk(self):
        """
        Cause animal instance to walk

        Walking funcionallity is a strategy, and is intended to
        be implemented separately by different types of animals.
        """
        message = '{} should implement a walk method'.format(
            self.__class__.__name__)
        raise NotImplementedError(message)


# Here are some different walking algorithms that can be used with Animal
def snake_walk(self):
    print('I am slithering side to side because I am a {}.'.format(self.name))

def four_legged_animal_walk(self):
    print('I am using all four of my legs to walk because I am a(n) {}.'.format(
        self.name))

def two_legged_animal_walk(self):
    print('I am standing up on my two legs to walk because I am a {}.'.format(
        self.name))
```

```
from types import MethodType


class Animal(object):

    def __init__(self, *args, **kwargs):
        self.name = kwargs.pop('name', None) or 'Animal'
        if kwargs.get('walk', None):
            self.walk = MethodType(kwargs.pop('walk'), self)

    def walk(self):
```

```
        """
        Cause animal instance to walk

        Walking funcionallity is a strategy, and is intended to
        be implemented separately by different types of animals.
        """
        message = '{} should implement a walk method'.format(
            self.__class__.__name__)
        raise NotImplementedError(message)


# Here are some different walking algorithms that can be used with Animal
def snake_walk(self):
    print('I am slithering side to side because I am a {}.'.format(self.name))

def four_legged_animal_walk(self):
    print('I am using all four of my legs to walk because I am a(n) {}.'.format(
        self.name))

def two_legged_animal_walk(self):
    print('I am standing up on my two legs to walk because I am a {}.'.format(
        self.name))
```

C ++Java。 Python`types.MethodType`。

Singleton

`commonly occurring problems。 GoF(Gang of Four)。`

1. `The pattern name。`
2. `The problem。`
3. `The solution。`
4. `The consequences。`

1. 。
2.
3.
4.

1.
2.
3.

`Creational Pattern -。`

`Structural Pattern -。`

`Behavioral Pattern -。`

`creational pattern。`

Singleton。

SingletonPythonSingleton Pattern。

```
class Singleton(object):
    def __new__(cls):
        # hasattr method checks if the class object an instance property or not.
        if not hasattr(cls, 'instance'):
            cls.instance = super(Singleton, cls).__new__(cls)
        return cls.instance

s = Singleton()
print ("Object created", s)

s1 = Singleton()
print ("Object2 created", s1)
```


```
class Singleton(object):
    def __new__(cls):
        # hasattr method checks if the class object an instance property or not.
        if not hasattr(cls, 'instance'):
            cls.instance = super(Singleton, cls).__new__(cls)
        return cls.instance

s = Singleton()
print ("Object created", s)

s1 = Singleton()
print ("Object2 created", s1)
```


C ++Java。 。 Python。

Creational pattern。 “ factory。 。 。


```
class Singleton(object):
    def __new__(cls):
        # hasattr method checks if the class object an instance property or not.
        if not hasattr(cls, 'instance'):
            cls.instance = super(Singleton, cls).__new__(cls)
        return cls.instance

s = Singleton()
print ("Object created", s)

s1 = Singleton()
print ("Object2 created", s1)
```



```
class Singleton(object):
    def __new__(cls):
        # hasattr method checks if the class object an instance property or not.
        if not hasattr(cls, 'instance'):
            cls.instance = super(Singleton, cls).__new__(cls)
        return cls.instance

s = Singleton()
print ("Object created", s)

s1 = Singleton()
print ("Object2 created", s1)
```


MusicFactoryMp3Ogg。

。

。

:(consumer_service

```
from datetime import date
from operator import attrgetter

class Proxy:
    def __init__(self, current_user, reservation_service):
        self.current_user = current_user
        self.reservation_service = reservation_service

    def highest_total_price_reservations(self, date_from, date_to, reservations_count):
        if self.current_user.can_see_reservations:
            return self.reservation_service.highest_total_price_reservations(
                date_from,
                date_to,
                reservations_count
              )
        else:
            return []

#Models and ReservationService:

class Reservation:
    def __init__(self, date, total_price):
        self.date = date
        self.total_price = total_price

class ReservationService:
    def highest_total_price_reservations(self, date_from, date_to, reservations_count):
        # normally it would be read from database/external service
        reservations = [
            Reservation(date(2014, 5, 15), 100),
            Reservation(date(2017, 5, 15), 10),
            Reservation(date(2017, 1, 15), 50)
        ]

        filtered_reservations = [r for r in reservations if (date_from <= r.date <= date_to)]

        sorted_reservations = sorted(filtered_reservations, key=attrgetter('total_price'),
reverse=True)

        return sorted_reservations[0:reservations_count]


class User:
    def __init__(self, can_see_reservations, name):
        self.can_see_reservations = can_see_reservations
        self.name = name

#Consumer service:

class StatsService:
    def __init__(self, reservation_service):
        self.reservation_service = reservation_service

    def year_top_100_reservations_average_total_price(self, year):
```

```
        reservations = self.reservation_service.highest_total_price_reservations(
            date(year, 1, 1),
            date(year, 12, 31),
            1
        )

        if len(reservations) > 0:
            total = sum(r.total_price for r in reservations)

            return total / len(reservations)
        else:
            return 0

#Test:
def test(user, year):
    reservations_service = Proxy(user, ReservationService())
    stats_service = StatsService(reservations_service)
    average_price = stats_service.year_top_100_reservations_average_total_price(year)
    print("{0} will see: {1}".format(user.name, average_price))

test(User(True, "John the Admin"), 2017)
test(User(False, "Guest"),         2017)
```

- **ReservationService**。
- **date_from date_to reservations_count** 。
- **StatsService**

CAVEATS

- 。

https://riptutorial.com/zh-TW/python/topic/8056/

180:

-
- print“”
- “”“
  “”“

PEP257 - Docstring Conventions。 Google Sphinx 。

Examples

。

Python。

`#` 。

-

```
# This is a single line comment in Python
```

-

```
# This is a single line comment in Python
```

- `"""'''` 。

```
# This is a single line comment in Python
```

- - 。

```
def func():
    """This is a function that does nothing at all"""
    return
```

`__doc__`docstring

```
def func():
    """This is a function that does nothing at all"""
    return
```

```
def func():
    """This is a function that does nothing at all"""
    return
```

`__main__func`

func()

function.__doc__docstringhelpdocstring。

```
def func():
    """This is a function that does nothing at all"""
    return
```

```
def func():
    """This is a function that does nothing at all"""
    return
```

__main__greet

greet(name, greeting='Hello')

name

greeting。

。

```
def func():
    """This is a function that does nothing at all"""
    return
```

```
def func():
    """This is a function that does nothing at all"""
    return
```

```
def func():
    """This is a function that does nothing at all"""
    return
```

main

greet(name, greeting='Hello')

docstrings

docstring。 。

```
def hello(name):
    """Greet someone.

    Print a greeting ("Hello") for the person with the given name.
    """

    print("Hello "+name)
```

```
def hello(name):
    """Greet someone.
```

```
    Print a greeting ("Hello") for the person with the given name.
    """

    print("Hello "+name)
```

docstring - - `help`。

PEP 257

PEP 257。

- Docstrings

PEP 257。

```
def hello(name):
    """Greet someone.

    Print a greeting ("Hello") for the person with the given name.
    """

    print("Hello "+name)
```

。

-

。

```
def hello(name):
    """Greet someone.

    Print a greeting ("Hello") for the person with the given name.
    """

    print("Hello "+name)
```

。

PEP 257docstring 。 。

SphinxdocstringspythonHTML。 reStructuredText 。 pythonhosted.org 。 pycharm IDESphinx。

Sphinx / reStructuredText

```
def hello(name):
    """Greet someone.

    Print a greeting ("Hello") for the person with the given name.
    """

    print("Hello "+name)
```

Google Python

GoogleGoogle Python Python。 Sphinx / reSTGoogle。

pythonhosted.orgGoogle Style Guide。

NapoleonSphinxGoogle Style Guide。

Google

```
def hello(name):
    """Greet someone.

    Print a greeting ("Hello") for the person with the given name.
    """

    print("Hello "+name)
```

https://riptutorial.com/zh-TW/python/topic/4144/

181: PythonJavaScript

。 。

Examples

`in`

```
2 in [2, 3]
```

PythonTrueJavaScriptfalse。 Python2[2,3]。 JavaScript。 JavaScript[2,3]

```
2 in [2, 3]
```

“2”。 2'2'。

PythonJavaScript https://riptutorial.com/zh-TW/python/topic/10766/--pythonjavascript

182:

Python 2[https://docs.python.org/2/library/locale.html#locale.currency] [1]

Examples

```
import locale

locale.setlocale(locale.LC_ALL, '')
Out[2]: 'English_United States.1252'

locale.currency(762559748.49)
Out[3]: '$762559748.49'

locale.currency(762559748.49, grouping=True)
Out[4]: '$762,559,748.49'
```

https://riptutorial.com/zh-TW/python/topic/1783/

183:

Examples

Python_pdb_

Python*pdb*。 *pdb*“”。

```
python -m pdb <my_file.py>
```

。

。 pdb*set_trace*。

```
python -m pdb <my_file.py>
```

。

```
python -m pdb <my_file.py>
```

```
python -m pdb <my_file.py>
```

Pdb。 python。 *p*python*print* 。

```
python -m pdb <my_file.py>
```

```
python -m pdb <my_file.py>
```

```
python -m pdb <my_file.py>
```

python

```
python -m pdb <my_file.py>
```

' 'var。 “ **c** ”。 ' **c** '。 ' **c** '

```
python -m pdb <my_file.py>
```

IPythonipdb

IPython Jupyter

```
import ipdb
```

```
ipdb.set_trace()
```

```
import ipdb
ipdb.set_trace()
```

。

```
import ipdb
ipdb.set_trace()
```

。

python `rpdb`。

rpdbpdbstdinstdout。 4444

```
# In the Python file you want to debug.
import rpdb
rpdb.set_trace()
```

。

```
# In the Python file you want to debug.
import rpdb
rpdb.set_trace()
```

pdb promt

```
# In the Python file you want to debug.
import rpdb
rpdb.set_trace()
```

https://riptutorial.com/zh-TW/python/topic/2077/

184: CSV

Examples

TSV

```
import csv

with open('/tmp/output.tsv', 'wt') as out_file:
    tsv_writer = csv.writer(out_file, delimiter='\t')
    tsv_writer.writerow(['name', 'field'])
    tsv_writer.writerow(['Dijkstra', 'Computer Science'])
    tsv_writer.writerow(['Shelah', 'Math'])
    tsv_writer.writerow(['Aumann', 'Economic Sciences'])
```

```
import csv

with open('/tmp/output.tsv', 'wt') as out_file:
    tsv_writer = csv.writer(out_file, delimiter='\t')
    tsv_writer.writerow(['name', 'field'])
    tsv_writer.writerow(['Dijkstra', 'Computer Science'])
    tsv_writer.writerow(['Shelah', 'Math'])
    tsv_writer.writerow(['Aumann', 'Economic Sciences'])
```

`dictDataFrame`CSV。

```
import pandas as pd

d = {'a': (1, 101), 'b': (2, 202), 'c': (3, 303)}
pd.DataFrame.from_dict(d, orient="index")
df.to_csv("data.csv")
```

CSV`DataFramedict`

```
import pandas as pd

d = {'a': (1, 101), 'b': (2, 202), 'c': (3, 303)}
pd.DataFrame.from_dict(d, orient="index")
df.to_csv("data.csv")
```

CSV https://riptutorial.com/zh-TW/python/topic/2116/csv

185:

Examples

/ Dunder

PythonMagicdunder。 。 。

。

```
import math

class Vector(object):
    # instantiation
    def __init__(self, x, y):
        self.x = x
        self.y = y

    # unary negation (-v)
    def __neg__(self):
        return Vector(-self.x, -self.y)

    # addition (v + u)
    def __add__(self, other):
        return Vector(self.x + other.x, self.y + other.y)

    # subtraction (v - u)
    def __sub__(self, other):
        return self + (-other)

    # equality (v == u)
    def __eq__(self, other):
        return self.x == other.x and self.y == other.y

    # abs(v)
    def __abs__(self):
        return math.hypot(self.x, self.y)

    # str(v)
    def __str__(self):
        return '<{0.x}, {0.y}>'.format(self)

    # repr(v)
    def __repr__(self):
        return 'Vector({0.x}, {0.y})'.format(self)
```

Vector。

```
import math

class Vector(object):
    # instantiation
    def __init__(self, x, y):
        self.x = x
```

```
        self.y = y

    # unary negation (-v)
    def __neg__(self):
        return Vector(-self.x, -self.y)

    # addition (v + u)
    def __add__(self, other):
        return Vector(self.x + other.x, self.y + other.y)

    # subtraction (v - u)
    def __sub__(self, other):
        return self + (-other)

    # equality (v == u)
    def __eq__(self, other):
        return self.x == other.x and self.y == other.y

    # abs(v)
    def __abs__(self):
        return math.hypot(self.x, self.y)

    # str(v)
    def __str__(self):
        return '<{0.x}, {0.y}>'.format(self)

    # repr(v)
    def __repr__(self):
        return 'Vector({0.x}, {0.y})'.format(self)
```

。

。

```
class sparselist(object):
    def __init__(self, size):
        self.size = size
        self.data = {}

    # l[index]
    def __getitem__(self, index):
        if index < 0:
            index += self.size
        if index >= self.size:
            raise IndexError(index)
        try:
            return self.data[index]
        except KeyError:
            return 0.0

    # l[index] = value
    def __setitem__(self, index, value):
        self.data[index] = value

    # del l[index]
    def __delitem__(self, index):
        if index in self.data:
            del self.data[index]
```

```
    # value in l
    def __contains__(self, value):
        return value == 0.0 or value in self.data.values()

    # len(l)
    def __len__(self):
        return self.size

    # for value in l: ...
    def __iter__(self):
        return (self[i] for i in range(self.size)) # use xrange for python2
```

listsparselist ◦

```
class sparselist(object):
    def __init__(self, size):
        self.size = size
        self.data = {}

    # l[index]
    def __getitem__(self, index):
        if index < 0:
            index += self.size
        if index >= self.size:
            raise IndexError(index)
        try:
            return self.data[index]
        except KeyError:
            return 0.0

    # l[index] = value
    def __setitem__(self, index, value):
        self.data[index] = value

    # del l[index]
    def __delitem__(self, index):
        if index in self.data:
            del self.data[index]

    # value in l
    def __contains__(self, value):
        return value == 0.0 or value in self.data.values()

    # len(l)
    def __len__(self):
        return self.size

    # for value in l: ...
    def __iter__(self):
        return (self[i] for i in range(self.size)) # use xrange for python2
```

```
class adder(object):
    def __init__(self, first):
        self.first = first

    # a(...)
    def __call__(self, second):
        return self.first + second
```

```
add2 = adder(2)
add2(1)  # 3
add2(2)  # 4
```

`return NotImplemented` `NotImplementedError` 。 Python

`NotImplemented`。 `NotImplemented` 。

`x + y` `x.__add__(y)` `y.__radd__(x)` 。

```
class NotAddable(object):

    def __init__(self, value):
        self.value = value

    def __add__(self, other):
        return NotImplemented


class Addable(NotAddable):

    def __add__(self, other):
        return Addable(self.value + other.value)

    __radd__ = __add__
```

`__add__` `__radd__`;。

```
class NotAddable(object):

    def __init__(self, value):
        self.value = value

    def __add__(self, other):
        return NotImplemented


class Addable(NotAddable):

    def __add__(self, other):
        return Addable(self.value + other.value)

    __radd__ = __add__
```

。

NB`other`。

`+`	`__add__(self, other)`	`a1 + a2`
`-`	`__sub__(self, other)`	`a1 - a2`
`*`	`__mul__(self, other)`	`a1 * a2`
`@`	`__matmul__(self, other)`	`a1 @ a2` *Python 3.5*

/	__div__(self, other)	a1 / a2 *Python 2*
/	__truediv__(self, other)	a1 / a2 *Python 3*
//	__floordiv__(self, other)	a1 // a2
% Modulo / Remainder	__mod__(self, other)	a1 % a2
**	__pow__(self, other[, modulo])	a1 ** a2
<<	__lshift__(self, other)	a1 << a2
>>	__rshift__(self, other)	a1 >> a2
&AND	__and__(self, other)	a1 & a2
^	__xor__(self, other)	a1 ^ a2
| OR	__or__(self, other)	a1 | a2
-	__neg__(self)	-a1
+	__pos__(self)	+a1
~	__invert__(self)	~a1
<	__lt__(self, other)	a1 < a2
<=	__le__(self, other)	a1 <= a2
==	__eq__(self, other)	a1 == a2
!=	__ne__(self, other)	a1 != a2
>	__gt__(self, other)	a1 > a2
>=	__ge__(self, other)	a1 >= a2
[index]	__getitem__(self, index)	a1[index]
in In	__contains__(self, other)	a2 in a1
(*args, ...)	__call__(self, *args, **kwargs)	a1(*args, **kwargs)

__pow__modulopow[o]

“right”__r __radd__

```
class A:
    def __init__(self, a):
        self.a = a
    def __add__(self, other):
        return self.a + other
    def __radd__(self, other):
        print("radd")
        return other + self.a
```

```
A(1) + 2  # Out:  3
2 + A(1)  # prints radd. Out: 3
```

__i__i

```
class A:
    def __init__(self, a):
        self.a = a
    def __add__(self, other):
        return self.a + other
    def __radd__(self, other):
        print("radd")
        return other + self.a

A(1) + 2  # Out:  3
2 + A(1)  # prints radd. Out: 3
```

◦ str()__str__◦ ◦

int	__int__(self)	int(a1)
	__abs__(self)	abs(a1)
str	__str__(self)	str(a1)
unicode	__unicode__(self)	unicode(a1) Python 2
	__repr__(self)	repr(a1)
bool	__nonzero__(self)	bool(a1)
	__format__(self, formatstr)	"Hi {:abc}".format(a1)
	__hash__(self)	hash(a1)
	__len__(self)	len(a1)
	__reversed__(self)	reversed(a1)
	__floor__(self)	math.floor(a1)
	__ceil__(self)	math.ceil(a1)

__enter____exit__ ◦

https://riptutorial.com/zh-TW/python/topic/2063/

186:

- typing.Callable [[intstr]None] - > def funcaintbstr - > None
- typing.Mapping [strint] - > {“a”1“b”2“c”3}
- typing.List [int] - > [1,2,3]
- typing.Set [int] - > {1,2,3}
- typing.Optional [int] - > Noneint
- typing.Sequence [int] - > [1,2,3]1,2,3
- 。 - >
- typing.Union [intstr] - > 1“1”
- T = typing.TypeVar'T' - >

PEP 484Python。 typingPython。

Examples

typing.TypeVar。 ///

```
import typing

T = typing.TypeVar("T")

def get_first_element(l: typing.Sequence[T]) -> T:
    """Gets the first element of a sequence."""
    return l[0]
```

```
def two_sum(a, b):
    return a + b
```

two_sum。 int

```
def two_sum(a, b):
    return a + b
```

```
def two_sum(a, b):
    return a + b
```

list S tuple S。

python。

。

int

```
def two_sum(a, b):
    return a + b
```

:◦

str

```
def two_sum(a, b):
    return a + b
```

◦ :->

```
def two_sum(a, b):
    return a + b
```

two_sumint◦ str float list set◦

IDE◦ __annotations__

```
def two_sum(a, b):
    return a + b
```

```
class A:
    x = None  # type: float
    def __init__(self, x: float) -> None:
        """
        self should not be annotated
        init should be annotated to return None
        """
        self.x = x

    @classmethod
    def from_int(cls, x: int) -> 'A':
        """
        cls should not be annotated
        Use forward reference to refer to current class with string literal 'A'
        """
        return cls(float(x))
```

◦ ◦

```
x = 3  # type: int
x = negate(x)
x = 'a type-checker might catch this error'
```

Python 3.x 3.6

Python 3.6 ◦

```
x = 3  # type: int
x = negate(x)
x = 'a type-checker might catch this error'
```

```
x = 3  # type: int
x = negate(x)
```

```
x = 'a type-checker might catch this error'
```

`typing.get_type_hints(class_or_module)`

```
x = 3  # type: int
x = negate(x)
x = 'a type-checker might catch this error'
```

`__annotations__`

```
x = 3  # type: int
x = negate(x)
x = 'a type-checker might catch this error'
```

NamedTuple

namedtuple`typing`NamedTuple

```
import typing
Point = typing.NamedTuple('Point', [('x', int), ('y', int)])
```

。

```
def hello_world(greeting: str = 'Hello'):
    print(greeting + ' world!')
```

。

https://riptutorial.com/zh-TW/python/topic/1766/

187: ...

Python `itertools.groupby()`。

- itertools.groupbyiterablekey = Nonesome function

	python

groupby

。

Examples

1

```
s = 'AAAABBBCCDAABBB'
```

'A"B"C'

```
s = 'AAAABBBCCDAABBB'
```

```
s = 'AAAABBBCCDAABBB'
```

。 groupby

```
s = 'AAAABBBCCDAABBB'
```

```
s = 'AAAABBBCCDAABBB'
```

group by'A"A'。 c

```
s = 'AAAABBBCCDAABBB'
```

```
s = 'AAAABBBCCDAABBB'
```

'A'。

2

```
c = groupby(['goat', 'dog', 'cow', 1, 1, 2, 3, 11, 10, ('persons', 'man', 'woman')])
```

```
dic = {}
for k, v in c:
    dic[k] = list(v)
dic
```

```
c = groupby(['goat', 'dog', 'cow', 1, 1, 2, 3, 11, 10, ('persons', 'man', 'woman')])
dic = {}
for k, v in c:
    dic[k] = list(v)
dic
```

3

mulatocamel。 。 c。 。

```
list_things = ['goat', 'dog', 'donkey', 'mulato', 'cow', 'cat', ('persons', 'man', 'woman'), \
               'wombat', 'mongoose', 'malloo', 'camel']
c = groupby(list_things, key=lambda x: x[0])
dic = {}
for k, v in c:
    dic[k] = list(v)
dic
```

```
list_things = ['goat', 'dog', 'donkey', 'mulato', 'cow', 'cat', ('persons', 'man', 'woman'), \
               'wombat', 'mongoose', 'malloo', 'camel']
c = groupby(list_things, key=lambda x: x[0])
dic = {}
for k, v in c:
    dic[k] = list(v)
dic
```

```
list_things = ['goat', 'dog', 'donkey', 'mulato', 'cow', 'cat', ('persons', 'man', 'woman'), \
               'wombat', 'mongoose', 'malloo', 'camel']
c = groupby(list_things, key=lambda x: x[0])
dic = {}
for k, v in c:
    dic[k] = list(v)
dic
```

```
list_things = ['goat', 'dog', 'donkey', 'mulato', 'cow', 'cat', ('persons', 'man', 'woman'), \
               'wombat', 'mongoose', 'malloo', 'camel']
c = groupby(list_things, key=lambda x: x[0])
dic = {}
for k, v in c:
    dic[k] = list(v)
dic
```

4

iterable。

```
things = [("animal", "bear"), ("animal", "duck"), ("plant", "cactus"), ("vehicle", "harley"),
\
```

```
          ("vehicle", "speed boat"), ("vehicle", "school bus")]
dic = {}
f = lambda x: x[0]
for key, group in groupby(sorted(things, key=f), f):
    dic[key] = list(group)
dic
```

```
things = [("animal", "bear"), ("animal", "duck"), ("plant", "cactus"), ("vehicle", "harley"),
\
          ("vehicle", "speed boat"), ("vehicle", "school bus")]
dic = {}
f = lambda x: x[0]
for key, group in groupby(sorted(things, key=f), f):
    dic[key] = list(group)
dic
```

。 。

```
things = [("animal", "bear"), ("animal", "duck"), ("plant", "cactus"), ("vehicle", "harley"),
\
          ("vehicle", "speed boat"), ("vehicle", "school bus")]
dic = {}
f = lambda x: x[0]
for key, group in groupby(sorted(things, key=f), f):
    dic[key] = list(group)
dic
```

```
things = [("animal", "bear"), ("animal", "duck"), ("plant", "cactus"), ("vehicle", "harley"),
\
          ("vehicle", "speed boat"), ("vehicle", "school bus")]
dic = {}
f = lambda x: x[0]
for key, group in groupby(sorted(things, key=f), f):
    dic[key] = list(group)
dic
```

... https://riptutorial.com/zh-TW/python/topic/8690/-----

188:

-

Python`if except def class`。

。 `class`。 `except`“”。 `for`。

`pass``IndentationError`。 `Ellipsis`...`pass`。 `pass` - 。

- `Exception` `xml`

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

`pandas``KeyboardInterrupt``SystemExit` `HardwareIsOnFireError` - 。

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

`except Error:``except OSError:`。 python10`except ...: pass`python。

- `scipy`

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

`__init__`。 `pebl`

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

- `mpmath`

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

- docstring。 docstring `pass` “`pebl`。 ”`pebl`

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

- `pass`“// if-block / .....”`Ellipsis...` python-3“no-op”。

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

```
try:
    self.version = "Expat %d.%d.%d" % expat.version_info
except AttributeError:
    pass # unknown
```

`update_agent`。 `raise NotImplementedError`。 “” “”“

Examples

```
try:
    metadata = metadata['properties']
except KeyError:
    pass
```

```
class CompileError(Exception):
    pass
```

https://riptutorial.com/zh-TW/python/topic/6891/

189:

。 。 。 Python。

Examples

Global Interpreter Lock Python。 PythonCPU。

```
import threading
import time


def process():
    time.sleep(2)


start = time.time()
process()
print("One run took %.2fs" % (time.time() - start))


start = time.time()
threads = [threading.Thread(target=process) for _ in range(4)]
for t in threads:
    t.start()
for t in threads:
    t.join()
print("Four runs took %.2fs" % (time.time() - start))

# Out: One run took 2.00s
# Out: Four runs took 2.00s
```

`process`22。

Python

```
import threading
import time


def process():
    time.sleep(2)


start = time.time()
process()
print("One run took %.2fs" % (time.time() - start))


start = time.time()
threads = [threading.Thread(target=process) for _ in range(4)]
for t in threads:
    t.start()
for t in threads:
    t.join()
```

```
print("Four runs took %.2fs" % (time.time() - start))

# Out: One run took 2.00s
# Out: Four runs took 2.00s
```

```
import threading
import time


def process():
    time.sleep(2)


start = time.time()
process()
print("One run took %.2fs" % (time.time() - start))


start = time.time()
threads = [threading.Thread(target=process) for _ in range(4)]
for t in threads:
    t.start()
for t in threads:
    t.join()
print("Four runs took %.2fs" % (time.time() - start))

# Out: One run took 2.00s
# Out: Four runs took 2.00s
```

threading.Thread。

```
import threading
import os

def process():
    print("Pid is %s, thread id is %s" % (os.getpid(), threading.current_thread().name))

threads = [threading.Thread(target=process) for _ in range(4)]
for t in threads:
    t.start()
for t in threads:
    t.join()

# Out: Pid is 11240, thread id is Thread-1
# Out: Pid is 11240, thread id is Thread-2
# Out: Pid is 11240, thread id is Thread-3
# Out: Pid is 11240, thread id is Thread-4
```

multiprocessing.Process。 threading.Thread

```
import multiprocessing
import os

def process():
    print("Pid is %s" % (os.getpid(),))

processes = [multiprocessing.Process(target=process) for _ in range(4)]
```

```
for p in processes:
    p.start()
for p in processes:
    p.join()

# Out: Pid is 11206
# Out: Pid is 11207
# Out: Pid is 11208
# Out: Pid is 11209
```

◦

◦

```
import threading

obj = {}
obj_lock = threading.Lock()

def objify(key, val):
    print("Obj has %d values" % len(obj))
    with obj_lock:
        obj[key] = val
    print("Obj now has %d values" % len(obj))

ts = [threading.Thread(target=objify, args=(str(n), n)) for n in range(4)]
for t in ts:
    t.start()
for t in ts:
    t.join()
print("Obj final result:")
import pprint; pprint.pprint(obj)

# Out: Obj has 0 values
# Out:  Obj has 0 values
# Out: Obj now has 1 values
# Out: Obj now has 2 valuesObj has 2 values
# Out: Obj now has 3 values
# Out:
# Out:  Obj has 3 values
# Out: Obj now has 4 values
# Out: Obj final result:
# Out: {'0': 0, '1': 1, '2': 2, '3': 3}
```

◦ multiprocessing◦

```
import multiprocessing

plain_num = 0
shared_num = multiprocessing.Value('d', 0)
lock = multiprocessing.Lock()

def increment():
    global plain_num
    with lock:
        # ordinary variable modifications are not visible across processes
        plain_num += 1
        # multiprocessing.Value modifications are
```

```
        shared_num.value += 1

ps = [multiprocessing.Process(target=increment) for n in range(4)]
for p in ps:
    p.start()
for p in ps:
    p.join()

print("plain_num is %d, shared_num is %d" % (plain_num, shared_num.value))

# Out: plain_num is 0, shared_num is 4
```

https://riptutorial.com/zh-TW/python/topic/4110/

190:

Python 。 3 * 2 + 732713.*+。

。

Python

Python。 。 。 。

	Lambda
	OR
	AND
x	NOT
innot inisnot not<<=>> =<>===	
|	OR
^	
	AND
<<>>	
+ -	
*///	[8]
+ x-x~x	NOT
**	[9]
x [index]x [indexindex]xarguments ...x.attribute	
.....[...]{...}.....	

Examples

python。

PythonPEMDAS。 PEMDAS。

```
>>> a, b, c, d = 2, 3, 5, 7
>>> a ** (b + c)  # parentheses
256
>>> a * b ** c  # exponent: same as `a * (b ** c)`
7776
>>> a + b * c / d  # multiplication / division: same as `a + (b * c / d)`
4.142857142857142
```


```
>>> a, b, c, d = 2, 3, 5, 7
>>> a ** (b + c)  # parentheses
256
>>> a * b ** c  # exponent: same as `a * (b ** c)`
7776
>>> a + b * c / d  # multiplication / division: same as `a + (b * c / d)`
4.142857142857142
```


https://riptutorial.com/zh-TW/python/topic/5040/

191:

-
- itertools.ifilter
- future_builtins.filterfunctioniterable
- itertools.ifilterfalse
- itertools.filterfalse

	None
	iterable

`filter()` `ifilter()` ◦

Examples

`filter`

```
names = ['Fred', 'Wilma', 'Barney']

def long_name(name):
    return len(name) > 5
```

Python 2.x 2.0

```
names = ['Fred', 'Wilma', 'Barney']

def long_name(name):
    return len(name) > 5
```

Python 2.x 2.6

```
names = ['Fred', 'Wilma', 'Barney']

def long_name(name):
    return len(name) > 5
```

Python 3.x 3.0

```
names = ['Fred', 'Wilma', 'Barney']

def long_name(name):
    return len(name) > 5
```

`None`

```
list(filter(None, [1, 0, 2, [], '', 'a']))  # discards 0, [] and ''
# Out: [1, 2, 'a']
```

Python 2.x 2.0.1

```
list(filter(None, [1, 0, 2, [], '', 'a']))  # discards 0, [] and ''
# Out: [1, 2, 'a']
```

Python 3.x 3.0.0

```
list(filter(None, [1, 0, 2, [], '', 'a']))  # discards 0, [] and ''
# Out: [1, 2, 'a']
```

`filter` 3.X`ifilter` 2.x`or`and

Python 2.x 2.0.1

```
 # not recommended in real use but keeps the example short:
from itertools import ifilter as filter
```

Python 2.x 2.6.1

```
 # not recommended in real use but keeps the example short:
from itertools import ifilter as filter
```

100

```
 # not recommended in real use but keeps the example short:
from itertools import ifilter as filter
```

`next`。

filterfalseifilterfalse

`itertools` -module`filter`

Python 2.x 2.0.1

```
 # not recommended in real use but keeps the example valid for python 2.x and python 3.x
from itertools import ifilterfalse as filterfalse
```

Python 3.x 3.0.0

```
 # not recommended in real use but keeps the example valid for python 2.x and python 3.x
from itertools import ifilterfalse as filterfalse
```

`filterFalse`

```
 # not recommended in real use but keeps the example valid for python 2.x and python 3.x
from itertools import ifilterfalse as filterfalse
```

```
 # not recommended in real use but keeps the example valid for python 2.x and python 3.x
from itertools import ifilterfalse as filterfalse
```

```
 # not recommended in real use but keeps the example valid for python 2.x and python 3.x
from itertools import ifilterfalse as filterfalse
```

```
 # not recommended in real use but keeps the example valid for python 2.x and python 3.x
from itertools import ifilterfalse as filterfalse
```


https://riptutorial.com/zh-TW/python/topic/201/

192:

stopCondition。

。

Examples

1n

1nn1 + 2 + 3 + 4 + ... + (several hours later) + n 。 for

```
n = 0
for i in range (1, n+1):
    n += i
```


```
n = 0
for i in range (1, n+1):
    n += i
```


。 1 + 2 + 313.recursion(4)

4 - > 4 + 3 - > 4 + 3 + 2 - > 4 + 3 + 2 + 1 - > 10

for1 - > 1 + 2 - > 1 + 2 + 3 - > 1 + 2 + 3 + 4 - > 10。 。 。

。 x! 。

- 01。
- 1。

Python

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

。 factorial(3) 。 。 n factorial print。 locals()。 n = 3 。 locals(){'n': 3} 。 n == 3 n * factorial(n - 1) 。

。 n。 factorial(n - 1) 。 n - 12 。 2factorialn。 n 。 *A* *B*。 *A*{'n': 3} *B* {'n': 2} 。 n * factorial(n - 1) 。 。 *B* n *B*n*A*n n - 1 。 n * ((n - 1) * factorial((n - 1) - 1)) 。 。

factorial((n - 1) - 1))。 *A*n == 3 1factorial 。 {'n': 1}*C*.n * factorial(n - 1) 。 “”factorial((n - 1) - 1)) 。 “”n * ((n - 1) * ((n - 2) * factorial((n - 2) - 1))) 。

。 `factorial((n - 2) - 1)` 。 `0` 。 `1` 。 。 “”`n * ((n - 1) * ((n - 2) * 1))` 。 A`3 * ((3 - 1) * ((3 - 2) * 1))` 。 6.`3! == 3 * 2 * 1 == 6` 。 。

`if n == 0: return 1`。 。 。 。 。 `factorial`

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

。

“”。 Fibonacci

- 00。
- 11。
- 。

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

`factorial(3)``fib(5)`

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

`(1 + (0 + 1)) + ((0 + 1) + (1 + (0 + 1)))` `5` 。

- 。 `return foo(n - 1)``return foo(n - 1) + 1`。
- TCO。
- TCE。 TCETCO。

- 。 。
- 。

PythonTCO。 。 。 `factorial``fib`

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

。

Pythonlru_cache。

Python“”。 。 。 。 Python。

SchemeHaskell。 。

Fibonaccipythonlru2。 `n` 。

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

。 。

```
root
- A
  - AA
  - AB
- B
  - BA
  - BB
    - BBA
```

for。 `get_name()` `get_children()` `get_root()` 。

```
root
- A
  - AA
  - AB
- B
  - BA
  - BB
    - BBA
```

.....。

```
root
- A
  - AA
  - AB
- B
  - BA
  - BB
    - BBA
```

。 。

```
root
- A
```

```
  - AA
  - AB
- B
  - BA
  - BB
    - BBA
```

Python。 RuntimeError

```
RuntimeError: Maximum Recursion Depth Exceeded
```

```
RuntimeError: Maximum Recursion Depth Exceeded
```

```
RuntimeError: Maximum Recursion Depth Exceeded
```

```
RuntimeError: Maximum Recursion Depth Exceeded
```

```
RuntimeError: Maximum Recursion Depth Exceeded
```

Python 3.5RecursionErrorRuntimeError。

-

。

```
def countdown(n):
    if n == 0:
        print "Blastoff!"
    else:
        print n
        countdown(n-1)
```

。 find_max

```
def countdown(n):
    if n == 0:
        print "Blastoff!"
    else:
        print n
        countdown(n-1)
```

PythonPython。 。

Python1000。 `sys.setrecursionlimit(15000)`。 Tail Recursion。

```
#!/usr/bin/env python2.4
# This program shows off a python decorator which implements tail call optimization. It
# does this by throwing an exception if it is it's own grandparent, and catching such
# exceptions to recall the stack.
```

```
import sys

class TailRecurseException:
    def __init__(self, args, kwargs):
        self.args = args
        self.kwargs = kwargs

def tail_call_optimized(g):
"""
This function decorates a function with tail call
optimization. It does this by throwing an exception
if it is it's own grandparent, and catching such
exceptions to fake the tail call optimization.

This function fails if the decorated
function recurses in a non-tail context.
"""

    def func(*args, **kwargs):
        f = sys._getframe()
        if f.f_back and f.f_back.f_back and f.f_back.f_back.f_code == f.f_code:
            raise TailRecurseException(args, kwargs)
        else:
            while 1:
                try:
                    return g(*args, **kwargs)
                except TailRecurseException, e:
                    args = e.args
                    kwargs = e.kwargs
    func.__doc__ = g.__doc__
    return func
```

@tail_call_optimized。

```
#!/usr/bin/env python2.4
# This program shows off a python decorator which implements tail call optimization. It
# does this by throwing an exception if it is it's own grandparent, and catching such
# exceptions to recall the stack.

import sys

class TailRecurseException:
    def __init__(self, args, kwargs):
        self.args = args
        self.kwargs = kwargs

def tail_call_optimized(g):
"""
This function decorates a function with tail call
optimization. It does this by throwing an exception
if it is it's own grandparent, and catching such
exceptions to fake the tail call optimization.

This function fails if the decorated
function recurses in a non-tail context.
"""

    def func(*args, **kwargs):
        f = sys._getframe()
```

```
        if f.f_back and f.f_back.f_back and f.f_back.f_back.f_code == f.f_code:
            raise TailRecurseException(args, kwargs)
        else:
            while 1:
                try:
                    return g(*args, **kwargs)
                except TailRecurseException, e:
                    args = e.args
                    kwargs = e.kwargs
    func.__doc__ = g.__doc__
    return func
```

```
#!/usr/bin/env python2.4
# This program shows off a python decorator which implements tail call optimization. It
# does this by throwing an exception if it is it's own grandparent, and catching such
# exceptions to recall the stack.

import sys

class TailRecurseException:
    def __init__(self, args, kwargs):
        self.args = args
        self.kwargs = kwargs

def tail_call_optimized(g):
"""
This function decorates a function with tail call
optimization. It does this by throwing an exception
if it is it's own grandparent, and catching such
exceptions to fake the tail call optimization.

This function fails if the decorated
function recurses in a non-tail context.
"""

    def func(*args, **kwargs):
        f = sys._getframe()
        if f.f_back and f.f_back.f_back and f.f_back.f_back.f_code == f.f_code:
            raise TailRecurseException(args, kwargs)
        else:
            while 1:
                try:
                    return g(*args, **kwargs)
                except TailRecurseException, e:
                    args = e.args
                    kwargs = e.kwargs
    func.__doc__ = g.__doc__
    return func
```

https://riptutorial.com/zh-TW/python/topic/1716/

193:

OOP。

。

- ** params_you_want_fix

X	
ÿ	

Python doc *functools.partial*

argskeyword argumentsfunc。 args。 。

。

Examples

xy 。

```
def raise_power(x, y):
    return x**y
```

y

y[3,4,5]。 *y*

```
def raise_power(x, y):
    return x**y
```

。

```
def raise_power(x, y):
    return x**y
```

y。

。

https://riptutorial.com/zh-TW/python/topic/9383/

194:

Examples

Conda

Anaconda BinstarAnaconda 1.6+binstar

```
$ conda install binstar
$ conda update binstar
```

AnacondaBinstarpypi

```
$ conda install binstar
$ conda update binstar
```

```
$ conda install binstar
$ conda update binstar
```

whoami

```
$ conda install binstar
$ conda update binstar
```

“hello world”。 Github

```
$ conda install binstar
$ conda update binstar
```

```
$ conda install binstar
$ conda update binstar
```

`Setup.py`python `hello.py`hello_world。

`bld.bat` `build.shmeta.yaml`Conda。 Conda。

```
$ conda install binstar
$ conda update binstar
```

Conda。

conda build test_package /binstar。

```
$ conda install binstar
$ conda update binstar
```

Web。

*done*Binstar。

https://riptutorial.com/zh-TW/python/topic/4064/

195:

。 。 。

Examples

。

```
class Node:
    def __init__(self, val):
        self.data = val
        self.next = None

    def getData(self):
        return self.data

    def getNext(self):
        return self.next

    def setData(self, val):
        self.data = val

    def setNext(self, val):
        self.next = val

class LinkedList:
    def __init__(self):
        self.head = None

    def isEmpty(self):
        """Check if the list is empty"""
        return self.head is None

    def add(self, item):
        """Add the item to the list"""
        new_node = Node(item)
        new_node.setNext(self.head)
        self.head = new_node

    def size(self):
        """Return the length/size of the list"""
        count = 0
        current = self.head
        while current is not None:
            count += 1
            current = current.getNext()
        return count

    def search(self, item):
        """Search for item in list. If found, return True. If not found, return False"""
        current = self.head
        found = False
        while current is not None and not found:
            if current.getData() is item:
                found = True
            else:
```

```
            current = current.getNext()
        return found

    def remove(self, item):
        """Remove item from list. If item is not found in list, raise ValueError"""
        current = self.head
        previous = None
        found = False
        while current is not None and not found:
            if current.getData() is item:
                found = True
            else:
                previous = current
                current = current.getNext()
        if found:
            if previous is None:
                self.head = current.getNext()
            else:
                previous.setNext(current.getNext())
        else:
            raise ValueError
            print 'Value not found.'

    def insert(self, position, item):
        """
        Insert item at position specified. If position specified is
        out of bounds, raise IndexError
        """
        if position > self.size() - 1:
            raise IndexError
            print "Index out of bounds."
        current = self.head
        previous = None
        pos = 0
        if position is 0:
            self.add(item)
        else:
            new_node = Node(item)
            while pos < position:
                pos += 1
                previous = current
                current = current.getNext()
            previous.setNext(new_node)
            new_node.setNext(current)

    def index(self, item):
        """
        Return the index where item is found.
        If item is not found, return None.
        """
        current = self.head
        pos = 0
        found = False
        while current is not None and not found:
            if current.getData() is item:
                found = True
            else:
                current = current.getNext()
                pos += 1
        if found:
            pass
```

```
        else:
            pos = None
        return pos

    def pop(self, position = None):
        """
        If no argument is provided, return and remove the item at the head.
        If position is provided, return and remove the item at that position.
        If index is out of bounds, raise IndexError
        """
        if position > self.size():
            print 'Index out of bounds'
            raise IndexError

        current = self.head
        if position is None:
            ret = current.getData()
            self.head = current.getNext()
        else:
            pos = 0
            previous = None
            while pos < position:
                previous = current
                current = current.getNext()
                pos += 1
                ret = current.getData()
            previous.setNext(current.getNext())
        print ret
        return ret

    def append(self, item):
        """Append item to the end of the list"""
        current = self.head
        previous = None
        pos = 0
        length = self.size()
        while pos < length:
            previous = current
            current = current.getNext()
            pos += 1
        new_node = Node(item)
        if previous is None:
            new_node.setNext(current)
            self.head = new_node
        else:
            previous.setNext(new_node)

    def printList(self):
        """Print the list"""
        current = self.head
        while current is not None:
            print current.getData()
            current = current.getNext()
```

。

```
class Node:
    def __init__(self, val):
        self.data = val
        self.next = None
```

```
    def getData(self):
        return self.data

    def getNext(self):
        return self.next

    def setData(self, val):
        self.data = val

    def setNext(self, val):
        self.next = val

class LinkedList:
    def __init__(self):
        self.head = None

    def isEmpty(self):
        """Check if the list is empty"""
        return self.head is None

    def add(self, item):
        """Add the item to the list"""
        new_node = Node(item)
        new_node.setNext(self.head)
        self.head = new_node

    def size(self):
        """Return the length/size of the list"""
        count = 0
        current = self.head
        while current is not None:
            count += 1
            current = current.getNext()
        return count

    def search(self, item):
        """Search for item in list. If found, return True. If not found, return False"""
        current = self.head
        found = False
        while current is not None and not found:
            if current.getData() is item:
                found = True
            else:
                current = current.getNext()
        return found

    def remove(self, item):
        """Remove item from list. If item is not found in list, raise ValueError"""
        current = self.head
        previous = None
        found = False
        while current is not None and not found:
            if current.getData() is item:
                found = True
            else:
                previous = current
                current = current.getNext()
        if found:
            if previous is None:
                self.head = current.getNext()
```

```
            else:
                previous.setNext(current.getNext())
        else:
            raise ValueError
            print 'Value not found.'

    def insert(self, position, item):
        """
        Insert item at position specified. If position specified is
        out of bounds, raise IndexError
        """
        if position > self.size() - 1:
            raise IndexError
            print "Index out of bounds."
        current = self.head
        previous = None
        pos = 0
        if position is 0:
            self.add(item)
        else:
            new_node = Node(item)
            while pos < position:
                pos += 1
                previous = current
                current = current.getNext()
            previous.setNext(new_node)
            new_node.setNext(current)

    def index(self, item):
        """
        Return the index where item is found.
        If item is not found, return None.
        """
        current = self.head
        pos = 0
        found = False
        while current is not None and not found:
            if current.getData() is item:
                found = True
            else:
                current = current.getNext()
                pos += 1
        if found:
            pass
        else:
            pos = None
        return pos

    def pop(self, position = None):
        """
        If no argument is provided, return and remove the item at the head.
        If position is provided, return and remove the item at that position.
        If index is out of bounds, raise IndexError
        """
        if position > self.size():
            print 'Index out of bounds'
            raise IndexError

        current = self.head
        if position is None:
            ret = current.getData()
```

```
            self.head = current.getNext()
        else:
            pos = 0
            previous = None
            while pos < position:
                previous = current
                current = current.getNext()
                pos += 1
                ret = current.getData()
            previous.setNext(current.getNext())
        print ret
        return ret

    def append(self, item):
        """Append item to the end of the list"""
        current = self.head
        previous = None
        pos = 0
        length = self.size()
        while pos < length:
            previous = current
            current = current.getNext()
            pos += 1
        new_node = Node(item)
        if previous is None:
            new_node.setNext(current)
            self.head = new_node
        else:
            previous.setNext(new_node)

    def printList(self):
        """Print the list"""
        current = self.head
        while current is not None:
            print current.getData()
            current = current.getNext()
```

https://riptutorial.com/zh-TW/python/topic/9299/

196:

Examples

python

- None
- 。

```
#! /usr/bin/env python

class Node:
      def __init__(self, cargo=None, next=None):
          self.car = cargo
          self.cdr = next
      def __str__(self):
          return str(self.car)


      def display(lst):
          if lst:
             w("%s " % lst)
             display(lst.cdr)
          else:
             w("nil\n")
```

https://riptutorial.com/zh-TW/python/topic/6916/

197: GZip

GNUgzipgunzip。

zlib。

gzipGzipFilePython。 GzipFilegzip。

Examples

GNU zip

```
import gzip
import os

outfilename = 'example.txt.gz'
output = gzip.open(outfilename, 'wb')
try:
    output.write('Contents of the example file go here.\n')
finally:
    output.close()

print outfilename, 'contains', os.stat(outfilename).st_size, 'bytes of compressed data'
os.system('file -b --mime %s' % outfilename)
```

1gzip_write.py1.Run。

```
import gzip
import os

outfilename = 'example.txt.gz'
output = gzip.open(outfilename, 'wb')
try:
    output.write('Contents of the example file go here.\n')
finally:
    output.close()

print outfilename, 'contains', os.stat(outfilename).st_size, 'bytes of compressed data'
os.system('file -b --mime %s' % outfilename)
```

GZip https://riptutorial.com/zh-TW/python/topic/8993/gzip

198:

- reducefunctioniterable [initializer]

iterable。	
	。
。	

`reduce`。

- `sum()`

```
sum([1,2,3])                                    # = 6
```

- `str.join`

```
sum([1,2,3])                                    # = 6
```

- `reduce next`

```
sum([1,2,3])                                    # = 6
```

Examples

```
# No import needed


# No import required...
from functools import reduce # ... but it can be loaded from the functools module


from functools import reduce # mandatory
```

`reduceiterable`。

```
# No import needed


# No import required...
from functools import reduce # ... but it can be loaded from the functools module


from functools import reduce # mandatory
```

`add`。 Python`operator`

```
# No import needed


# No import required...
from functools import reduce # ... but it can be loaded from the functools module


from functools import reduce # mandatory
```

reduce

```
# No import needed


# No import required...
from functools import reduce # ... but it can be loaded from the functools module


from functools import reduce # mandatory
```

reduce

```
def multiply(s1, s2):
    print('{arg1} * {arg2} = {res}'.format(arg1=s1,
                                           arg2=s2,
                                           res=s1*s2))
    return s1 * s2

asequence = [1, 2, 3]
```

initializer

```
def multiply(s1, s2):
    print('{arg1} * {arg2} = {res}'.format(arg1=s1,
                                           arg2=s2,
                                           res=s1*s2))
    return s1 * s2

asequence = [1, 2, 3]
```

initializerreduce

```
def multiply(s1, s2):
    print('{arg1} * {arg2} = {res}'.format(arg1=s1,
                                           arg2=s2,
                                           res=s1*s2))
    return s1 * s2

asequence = [1, 2, 3]
```

```
import operator
reduce(operator.mul, [10, 5, -3])
# Out: -150
```

/

`reduceiterableany()all()`

```
import operator
# non short-circuit "all"
reduce(operator.and_, [False, True, True, True]) # = False

# non short-circuit "any"
reduce(operator.or_, [True, False, False, False]) # = True
```

truthy / falsy

```
# First falsy element or last element if all are truthy:
reduce(lambda i, j: i and j, [100, [], 20, 10])    # = []
reduce(lambda i, j: i and j, [100, 50, 20, 10])    # = 10

# First truthy element or last element if all falsy:
reduce(lambda i, j: i or j, [100, [], 20, 0])     # = 100
reduce(lambda i, j: i or j, ['', {}, [], None])   # = None
```

`lambda`

```
# First falsy element or last element if all are truthy:
reduce(lambda i, j: i and j, [100, [], 20, 10])    # = []
reduce(lambda i, j: i and j, [100, 50, 20, 10])    # = 10

# First truthy element or last element if all falsy:
reduce(lambda i, j: i or j, [100, [], 20, 0])     # = 100
reduce(lambda i, j: i or j, ['', {}, [], None])   # = None
```

https://riptutorial.com/zh-TW/python/topic/328/

199:

Queue。 。 1。 2. LifoQueue 3.PriorityQueue1。 2。

Examples

```
from Queue import Queue

question_queue = Queue()

for x in range(1,10):
    temp_dict = ('key', x)
    question_queue.put(temp_dict)

while(not question_queue.empty()):
    item = question_queue.get()
    print(str(item))
```

```
from Queue import Queue

question_queue = Queue()

for x in range(1,10):
    temp_dict = ('key', x)
    question_queue.put(temp_dict)

while(not question_queue.empty()):
    item = question_queue.get()
    print(str(item))
```

https://riptutorial.com/zh-TW/python/topic/8339/

200:

- random.seeda == 2python 3.x
- random.getstate
- random.setstate
- random.randintab
- random.randrange
- random.randrangestartstopstep = 1
- random.choiceSEQ
- random.shufflexrandom = random.random
- random.samplepopulationk

Examples

```
import random
```

`random.shuffle()`/。 `list`

```
import random
```

```
import random
```

`choice`

```
import random
```

```
import random
```

ValueError

randintrandrangerandomuniform

```
import random
```

randint

`xy`

```
import random
```

18

```
import random
```

randrange

random.randrangerangerandom.randint

```
import random
```

01

```
import random
```

xy

```
import random
```

```
random.seed(5)                  # Create a fixed state
print(random.randrange(0, 10))   # Get a random integer between 0 and 9
# Out: 9
print(random.randrange(0, 10))
# Out: 4
```

“”

```
random.seed(5)                  # Create a fixed state
print(random.randrange(0, 10))   # Get a random integer between 0 and 9
# Out: 9
print(random.randrange(0, 10))
# Out: 4
```

94 ◦ getstatesetstate

```
random.seed(5)                  # Create a fixed state
print(random.randrange(0, 10))   # Get a random integer between 0 and 9
# Out: 9
print(random.randrange(0, 10))
# Out: 4
```

None seed

```
random.seed(5)                  # Create a fixed state
print(random.randrange(0, 10))   # Get a random integer between 0 and 9
# Out: 9
print(random.randrange(0, 10))
# Out: 4
```

seed

```
random.seed(5)                  # Create a fixed state
print(random.randrange(0, 10))   # Get a random integer between 0 and 9
# Out: 9
print(random.randrange(0, 10))
# Out: 4
```

PythonMersenne Twister PRNG◦

SystemRandomos.urandom CPRNG ◦

SystemRandom◦ ◦

```
from random import SystemRandom
secure_rand_gen = SystemRandom()
```

[0, 20] randrange() [0, 20]10intrandrange()

```
from random import SystemRandom
secure_rand_gen = SystemRandom()
```

randint

```
from random import SystemRandom
secure_rand_gen = SystemRandom()
```

。 。

os.urandom。

string。 punctuationascii_lettersdigits

```
from string import punctuation, ascii_letters, digits
```

symbolssymbols

```
from string import punctuation, ascii_letters, digits
```

。

random.SystemRandom。 10

```
from string import punctuation, ascii_letters, digits
```

random - random.choice random.randint - 。

Mersenne Twister PRNG CSPRNG。 。 SystemRandom。

Python 3.x 3.6

Python 3.6 secrets。

“”

```
from string import punctuation, ascii_letters, digits
```

```
import random

probability = 0.3

if random.random() < probability:
    print("Decision with probability 0.3")
else:
    print("Decision with probability 0.7")
```

https://riptutorial.com/zh-TW/python/topic/239/

201:

Examples

Python。 。 `__action__`。 Python 。

。 Python。 Python。 `Vector`。 。

```
class Vector(object):
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __add__(self, v):
        # Addition with another vector.
        return Vector(self.x + v.x, self.y + v.y)

    def __sub__(self, v):
        # Subtraction with another vector.
        return Vector(self.x - v.x, self.y - v.y)

    def __mul__(self, s):
        # Multiplication with a scalar.
        return Vector(self.x * s, self.y * s)

    def __div__(self, s):
        # Division with a scalar.
        float_s = float(s)
        return Vector(self.x / float_s, self.y / float_s)

    def __floordiv__(self, s):
        # Division with a scalar (value floored).
        return Vector(self.x // s, self.y // s)

    def __repr__(self):
        # Print friendly representation of Vector class. Else, it would
        # show up like, <__main__.Vector instance at 0x01DDDDC8>.
        return '<Vector (%f, %f)>' % (self.x, self.y, )

a = Vector(3, 5)
b = Vector(2, 7)

print a + b # Output: <Vector (5.000000, 12.000000)>
print b - a # Output: <Vector (-1.000000, 2.000000)>
print b * 1.3 # Output: <Vector (2.600000, 9.100000)>
print a // 17 # Output: <Vector (0.000000, 0.000000)>
print a / 17 # Output: <Vector (0.176471, 0.294118)>
```

。 。

https://riptutorial.com/zh-TW/python/topic/946/

202: Python

Examples

IronPython

C.NETMonoApache License 2.0。 DLR。 2.73。

CPython

- .NET Framework。
- Unicode。
- CCPython。
- Global Interpreter Lock。
- 。

```
print "Hello World!"
```

.NET

```
print "Hello World!"
```

-
- GitHub

Jython

JavaJVMPython Software Foundation License。 2.73。

CPython

- JVM。
- Unicode。
- CCPython。
- Global Interpreter Lock。
- 。

```
print "Hello World!"
```

Java

```
print "Hello World!"
```

-
- Mercurial

Transcrypt

TranscryptPythonJavascript。

- PythonOOCPython
- WebJavaScriptPython
- URLPyPi
- PythonJavaScript
-
- KBMB
- JavaScriptmemoization
-

---

650 kBPythonJavaScript。 JavaScript。

HTML

```
<script src="__javascript__/hello.js"></script>
<h2>Hello demo</h2>

<p>
<div id = "greet">...</div>
<button onclick="hello.solarSystem.greet ()">Click me repeatedly!</button>

<p>
<div id = "explain">...</div>
<button onclick="hello.solarSystem.explain ()">And click me repeatedly too!</button>
```

JavaScriptDOM

```
<script src="__javascript__/hello.js"></script>
<h2>Hello demo</h2>

<p>
<div id = "greet">...</div>
<button onclick="hello.solarSystem.greet ()">Click me repeatedly!</button>

<p>
<div id = "explain">...</div>
<button onclick="hello.solarSystem.explain ()">And click me repeatedly too!</button>
```

JavaScript

TranscryptJavaScript。 ao react.jsriot.jsfabric.jsnode.js。

PythonJavaScript

```
<script src="__javascript__/hello.js"></script>
<h2>Hello demo</h2>

<p>
<div id = "greet">...</div>
<button onclick="hello.solarSystem.greet ()">Click me repeatedly!</button>

<p>
<div id = "explain">...</div>
<button onclick="hello.solarSystem.explain ()">And click me repeatedly too!</button>
```

JavaScript

```
<script src="__javascript__/hello.js"></script>
<h2>Hello demo</h2>

<p>
<div id = "greet">...</div>
<button onclick="hello.solarSystem.greet ()">Click me repeatedly!</button>

<p>
<div id = "explain">...</div>
<button onclick="hello.solarSystem.explain ()">And click me repeatedly too!</button>
```

- http //www.transcrypt.org/
- https //github.com/JdeH/Transcrypt

Python https://riptutorial.com/zh-TW/python/topic/5225/python

203:

Examples

Pyglet

```
import pyglet
audio = pyglet.media.load("audio.wav")
audio.play()
```

pyglet

WAV

- Windows

```
import winsound
winsound.PlaySound("path_to_wav_file.wav", winsound.SND_FILENAME)
```

- /
- /

```
import winsound
winsound.PlaySound("path_to_wav_file.wav", winsound.SND_FILENAME)
```

pythonffmpeg

```
from subprocess import check_call

ok = check_call(['ffmpeg','-i','input.mp3','output.wav'])
if ok:
    with open('output.wav', 'rb') as f:
        wav_file = f.read()
```

- http://superuser.com/questions/507386/why-would-i-choose-libav-over-ffmpeg-or-is-there-even-a-difference
- ffmpeglibavavconv

Windows

Windows `winsound`。

```
import winsound
freq = 2500 # Set frequency To 2500 Hertz
```

```
dur = 1000 # Set duration To 1000 ms == 1 second
winsound.Beep(freq, dur)
```

https://riptutorial.com/zh-TW/python/topic/8189/

204:

Python。 。 。

。

Examples

PythonJavaC ++。 。

```
class BaseClass(object):
    pass

class DerivedClass(BaseClass):
    pass
```

`BaseClass` `DerivedClassBaseClass` 。 Python 2.2`objectobject`。

`Rectangleobject`

```
class BaseClass(object):
    pass

class DerivedClass(BaseClass):
    pass
```

`RectangleSquare`square。

```
class BaseClass(object):
    pass

class DerivedClass(BaseClass):
    pass
```

`SquareRectangle`。 `super()Rectangle__init__()`。 Python 3 `super()`。

```
class BaseClass(object):
    pass

class DerivedClass(BaseClass):
    pass
```

`issubclass(DerivedClass, BaseClass)` `DerivedClassBaseClassTrue`

`isinstance(s, Class)``True`S`ClassClass`

```
class BaseClass(object):
    pass
```

```
class DerivedClass(BaseClass):
    pass
```

。

```
class C:
    x = 2  # class variable

    def __init__(self, y):
        self.y = y  # instance variable

C.x
# 2
C.y
# AttributeError: type object 'C' has no attribute 'y'

c1 = C(3)
c1.x
# 2
c1.y
# 3

c2 = C(4)
c2.x
# 2
c2.y
# 4
```

class

```
class C:
    x = 2  # class variable

    def __init__(self, y):
        self.y = y  # instance variable

C.x
# 2
C.y
# AttributeError: type object 'C' has no attribute 'y'

c1 = C(3)
c1.x
# 2
c1.y
# 3

c2 = C(4)
c2.x
# 2
c2.y
# 4
```

。

```
class C:
```

```
    x = 2  # class variable

    def __init__(self, y):
        self.y = y  # instance variable

C.x
# 2
C.y
# AttributeError: type object 'C' has no attribute 'y'

c1 = C(3)
c1.x
# 2
c1.y
# 3

c2 = C(4)
c2.x
# 2
c2.y
# 4
```

Python 3。 Python 3def。 。 Af Af

Python 3.x 3.0

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

Python 2instancemethod。 .__func__。

Python 2.x 2.3

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

- Python 3Python 2。

Python 3.x 3.0

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

Python 2.x 2.3

```
class A(object):
    def f(self, x):
```

```
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

Python/AAf。

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

aAAafAf “”a - Pythona。

af__getattribute__a af Ammethod Afm.__func__ am.__self__。 m(...) => m.__func__(m.__self__, ...)。。 Python 23。

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

Python -。。 m.__self__ = type(a)。 a。。

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

```
class A(object):
    def f(self, x):
        return 2 * x
A.f
# <function A.f at ...>  (in Python 3.x)
```

staticmethods__get__ __set__ __del__。 classmethodsstaticmethods

- Python@staticmethod@classmethod
- @classmethod@staticmethod

Python 2.x 2.2.0

Python 2.2。 object。 。

```
# new-style class
class New(object):
    pass

# new-style instance
```

```
new = New()

new.__class__
# <class '__main__.New'>
type(new)
# <class '__main__.New'>
issubclass(New, object)
# True
```

`object`。 `instance`。

```
# new-style class
class New(object):
    pass

# new-style instance
new = New()

new.__class__
# <class '__main__.New'>
type(new)
# <class '__main__.New'>
issubclass(New, object)
# True
```

Python 3.x 3.0.0

Python 3。

Python 3`object` `MyClass(object)` 。

```
# new-style class
class New(object):
    pass

# new-style instance
new = New()

new.__class__
# <class '__main__.New'>
type(new)
# <class '__main__.New'>
issubclass(New, object)
# True
```

```
class Rectangle(object):
    def __init__(self, width, height, color='blue'):
        self.width = width
        self.height = height
        self.color = color

    def area(self):
        return self.width  * self.height

# Create some instances of the class
default_rectangle = Rectangle(2, 3)
print(default_rectangle.color) # blue
```

```
red_rectangle = Rectangle(2, 3, 'red')
print(red_rectangle.color) # red
```

。

```
class Rectangle(object):
    def __init__(self, width, height, color='blue'):
        self.width = width
        self.height = height
        self.color = color

    def area(self):
        return self.width  * self.height

# Create some instances of the class
default_rectangle = Rectangle(2, 3)
print(default_rectangle.color) # blue

red_rectangle = Rectangle(2, 3, 'red')
print(red_rectangle.color) # red
```

Python。


```
class Rectangle(object):
    def __init__(self, width, height, color='blue'):
        self.width = width
        self.height = height
        self.color = color

    def area(self):
        return self.width  * self.height

# Create some instances of the class
default_rectangle = Rectangle(2, 3)
print(default_rectangle.color) # blue

red_rectangle = Rectangle(2, 3, 'red')
print(red_rectangle.color) # red
```


Mutable Default Arguments“Least Astonishment”Mutable Default Argument 。

PythonC3。 MRO。

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

FooBarfooFoo

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

FooBarMRO

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

PythonMRO

1. `FooBarFoo`
2. `object Bar`
3. 。

FooBarBarBarFooBar。

Python 。

`super` 。 super。

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'
```

```
class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

initinitineritanceinit。

Foo**initBar**init

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

Bar。 **FooBarBarFoo**。

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

```
class Foo(object):
    foo = 'attr foo of Foo'


class Bar(object):
    foo = 'attr foo of Bar' # we won't see this.
    bar = 'attr bar of Bar'

class FooBar(Foo, Bar):
    foobar = 'attr foobar of FooBar'
```

`__get__` `__set__`、`__delete__`。

__set____delete__

staticmethod classmethodproperty ◦ barFoofoo - foo.bar

1. bar Foo ◦ ◦ property◦
2. __dict__bar ◦ ◦ bar ◦
3. Foo bar ◦ ◦ classmethodstaticmethod◦ AttributeError

◦ ◦

Person

```
class Person(object):

    def __init__(self, first_name, last_name, age):
        self.first_name = first_name
        self.last_name = last_name
        self.age = age
        self.full_name = first_name + " " + last_name

    def greet(self):
        print("Hello, my name is " + self.full_name + ".")
```

◦ last_name

```
class Person(object):

    def __init__(self, first_name, last_name, age):
        self.first_name = first_name
        self.last_name = last_name
        self.age = age
        self.full_name = first_name + " " + last_name

    def greet(self):
        print("Hello, my name is " + self.full_name + ".")
```

1. first_namelast_namefirst_name◦ /if / elif / else◦
2. last_nameNone first_name/.....

◦ from_full_namefrom_full_name classmethod◦

```
class Person(object):

    def __init__(self, first_name, last_name, age):
        self.first_name = first_name
        self.last_name = last_name
        self.age = age
        self.full_name = first_name + " " + last_name

    def greet(self):
        print("Hello, my name is " + self.full_name + ".")
```

clsselffrom_full_name。 self。 clsPersonfrom_full_namePerson(first_name, last_name, age) Person __init__Person。 Employee of Person from_full_nameEmployee。

Person__init__

```
class Person(object):

    def __init__(self, first_name, last_name, age):
        self.first_name = first_name
        self.last_name = last_name
        self.age = age
        self.full_name = first_name + " " + last_name

    def greet(self):
        print("Hello, my name is " + self.full_name + ".")
```

- Python @classmethod@staticmethod
- https://docs.python.org/2/library/functions.html#classmethod
- https://docs.python.org/3.5/library/functions.html#classmethod

。 /。

```
class Country(object):
    def __init__(self):
        self.cities=[]

    def addCity(self,city):
        self.cities.append(city)


class City(object):
    def __init__(self, numPeople):
        self.people = []
        self.numPeople = numPeople


    def addPerson(self, person):
        self.people.append(person)

    def join_country(self,country):
        self.country = country
        country.addCity(self)

        for i in range(self.numPeople):
                person(i).join_city(self)


class Person(object):
    def __init__(self, ID):
        self.ID=ID

    def join_city(self, city):
        self.city = city
        city.addPerson(self)

    def people_in_my_country(self):
```

```
        x= sum([len(c.people) for c in self.city.country.cities])
        return x

US=Country()
NYC=City(10).join_country(US)
SF=City(5).join_country(US)

print(US.cities[0].people[0].people_in_my_country())

# 15
```

“”。 A

```
class A(object):
    def __init__(self, num):
        self.num = num

    def __add__(self, other):
        return A(self.num + other.num)
```

。 。

```
class A(object):
    def __init__(self, num):
        self.num = num

    def __add__(self, other):
        return A(self.num + other.num)
```

AA。

```
class A(object):
    def __init__(self, num):
        self.num = num

    def __add__(self, other):
        return A(self.num + other.num)
```

。

get_numA

。

```
class A(object):
    def __init__(self, num):
        self.num = num

    def __add__(self, other):
        return A(self.num + other.num)
```

。

dir()

```
dir(Class)
```

```
dir(Class)
```

“”。 __

```
dir(Class)
```

`__dir__()`。 `dir()``__dir__()` Python。 dir。 python

dir dict。 **getattr** 。

dir。 。

。

```
class Person(object):
     """A simple class."""                            # docstring
     species = "Homo Sapiens"                         # class attribute

     def __init__(self, name):                        # special method
         """This is the initializer. It's a special
         method (see below).
         """
         self.name = name                             # instance attribute

     def __str__(self):                               # special method
         """This method is run when Python tries
         to cast the object to a string. Return
         this string when using print(), etc.
         """
         return self.name

     def rename(self, renamed):                       # regular method
         """Reassign and print the name attribute."""
         self.name = renamed
         print("Now my name is {}".format(self.name))
```

。

1. 。
2. 。
3. docstring `__init__()` 。 。
4. 。
5. 。 `__init__()`;`__init__()`。
6. 。 `self` `self`self PythonPython。
7. 。 Python`private`C ++ / Java `public`。 “”。
8. `__functionname__(self, other_stuff)` 。 “”Python。 Python。 。

```
Person
```

```
class Person(object):
     """A simple class."""                            # docstring
     species = "Homo Sapiens"                         # class attribute

     def __init__(self, name):                        # special method
         """This is the initializer. It's a special
         method (see below).
         """
         self.name = name                             # instance attribute

     def __str__(self):                               # special method
         """This method is run when Python tries
         to cast the object to a string. Return
         this string when using print(), etc.
         """
         return self.name

     def rename(self, renamed):                       # regular method
         """Reassign and print the name attribute."""
         self.name = renamed
         print("Now my name is {}".format(self.name))
```

Person kelly josephjohn_doe ◦

.

```
class Person(object):
     """A simple class."""                            # docstring
     species = "Homo Sapiens"                         # class attribute

     def __init__(self, name):                        # special method
         """This is the initializer. It's a special
         method (see below).
         """
         self.name = name                             # instance attribute

     def __str__(self):                               # special method
         """This method is run when Python tries
         to cast the object to a string. Return
         this string when using print(), etc.
         """
         return self.name

     def rename(self, renamed):                       # regular method
         """Reassign and print the name attribute."""
         self.name = renamed
         print("Now my name is {}".format(self.name))
```

.

```
class Person(object):
     """A simple class."""                            # docstring
     species = "Homo Sapiens"                         # class attribute

     def __init__(self, name):                        # special method
         """This is the initializer. It's a special
         method (see below).
         """
         self.name = name                             # instance attribute
```

```
    def __str__(self):                              # special method
        """This method is run when Python tries
        to cast the object to a string. Return
        this string when using print(), etc.
        """
        return self.name

    def rename(self, renamed):                      # regular method
        """Reassign and print the name attribute."""
        self.name = renamed
        print("Now my name is {}".format(self.name))
```

Python 。

```
class MyClass(object):

    def __init__(self):
       self._my_string = ""

    @property
    def string(self):
        """A profoundly important string."""
        return self._my_string

    @string.setter
    def string(self, new_value):
        assert isinstance(new_value, str), \
               "Give me a string, not a %r!" % type(new_value)
        self._my_string = new_value

    @string.deleter
    def x(self):
        self._my_string = None
```

```
MyClass.string
```

```
class MyClass(object):

    def __init__(self):
       self._my_string = ""

    @property
    def string(self):
        """A profoundly important string."""
        return self._my_string

    @string.setter
    def string(self, new_value):
        assert isinstance(new_value, str), \
               "Give me a string, not a %r!" % type(new_value)
        self._my_string = new_value

    @string.deleter
    def x(self):
        self._my_string = None
```

。 API - 。

“” - 。

```
class MyClass(object):

    def __init__(self):
       self._my_string = ""

    @property
    def string(self):
        """A profoundly important string."""
        return self._my_string

    @string.setter
    def string(self, new_value):
        assert isinstance(new_value, str), \
               "Give me a string, not a %r!" % type(new_value)
        self._my_string = new_value

    @string.deleter
    def x(self):
        self._my_string = None
```

/。 python singleton 。

```
class Singleton:
    def __new__(cls):
        try:
            it = cls.__it__
        except AttributeError:
            it = cls.__it__ = object.__new__(cls)
        return it

    def __repr__(self):
        return '<{}>'.format(self.__class__.__name__.upper())

    def __eq__(self, other):
         return other is self
```

。 Singleton

```
class Singleton:
    def __new__(cls):
        try:
            it = cls.__it__
        except AttributeError:
            it = cls.__it__ = object.__new__(cls)
        return it

    def __repr__(self):
        return '<{}>'.format(self.__class__.__name__.upper())

    def __eq__(self, other):
         return other is self
```

Instance

```
class Singleton:
```

```
    def __new__(cls):
        try:
            it = cls.__it__
        except AttributeError:
            it = cls.__it__ = object.__new__(cls)
        return it

    def __repr__(self):
        return '<{}>'.format(self.__class__.__name__.upper())

    def __eq__(self, other):
         return other is self
```

https://riptutorial.com/zh-TW/python/topic/419/

205: __str__ __repr__

__str__“Ace of Spaces”__repr__eval。

repr()Python

eval。

__str__“Ace of Spaces” __repr__Card('Spades', 1)

“”eval

```
object -> string -> object
```


```
object -> string -> object
```


[1]。 cpython。

[2]str() / repr()。 str()str()。 repr()。

Examples

Python

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```


```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```


“”


```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```


```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```


。 。

。 print(ace_of_spades)Pythonace_of_spadesCard。 。

typeid。 print。 [1]

Python“”。


```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```


Card。

PythonCardCard。

str(ace_of_spades) Card。

1

Python。 __str__ “dunder”“”。

Python__str__。

Card


```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```


Card__str__ 。

“”。 str(ace_of_spades)str(ace_of_spades) 。 __str__。 。

__str__self additonal。

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

Card hand 。

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

PythonCard

2

Python。 Python__str__。

__repr__ “。 ”[2]

print

__str____repr__

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

__str____repr__(S)__str__(R)__repr____repr__ 。

__str__ __repr__。

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
```

```
        self.pips = pips
```

__str__Cardprint__repr__ print 。

str() repr() repr() 。

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

■■■■■■

Python str() repr() - 。

。 。 __repr__ __str__str() Python__repr____repr__。

Card

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

__repr__。 str()

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

repr()

```
class Card:
    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips
```

“”__repr__。 Raymond Hettinger__repr__Python。 ID。

__repr____str__。 。

eval-round-trip__repr __

```
class Card:
    special_names = {1:'Ace', 11:'Jack', 12:'Queen', 13:'King'}

    def __init__(self, suit, pips):
        self.suit = suit
        self.pips = pips

    # Called when instance is converted to a string via str()
    # Examples:
    #   print(card1)
    #   print(str(card1)
    def __str__(self):
        card_name = Card.special_names.get(self.pips, str(self.pips))
        return "%s of %s" % (card_name, self.suit)

    # Called when instance is converted to a string via repr()
    # Examples:
    #   print([card1, card2, card3])
    #   print(repr(card1))
    def __repr__(self):
        return "Card(%s, %d)" % (self.suit, self.pips)
```

__str____repr__ https://riptutorial.com/zh-TW/python/topic/4845/---str----repr--

206:

Examples

Ninja TwistTurtle Graphics

Turtle Graphics Ninja Twist

```
import turtle

ninja = turtle.Turtle()

ninja.speed(10)

for i in range(180):
    ninja.forward(100)
    ninja.right(30)
    ninja.forward(20)
    ninja.left(60)
    ninja.forward(50)
    ninja.right(30)

    ninja.penup()
    ninja.setposition(0, 0)
    ninja.pendown()

    ninja.right(2)

turtle.done()
```

https://riptutorial.com/zh-TW/python/topic/7915/

S. No		Contributors
1	Python	A. Raza, Aaron Critchley, Abhishek Jain, AER, afeique, Akshay Kathpal, alejosocorro, Alessandro Trinca Tornidor, Alex Logan, ALinuxLover, Andrea, Andrii Abramov, Andy, Andy Hayden, angussidney, Ani Menon, Anthony Pham, Antoine Bolvy, Aquib Javed Khan, Ares, Arpit Solanki, B8vrede, Baaing Cow, baranskistad, Brian C, Bryan P, BSL-5, BusyAnt, Cbeb24404, ceruleus, ChaoticTwist, Charlie H, Chris Midgley, Christian Ternus, Claudiu, Clíodhna, CodenameLambda, cLDs EED, Community, Conrad.Dean, Daksh Gupta, Dania, Daniel Minnaar, Darth Shadow, Dartmouth, deeenes, Delgan, depperm, DevD, dodell, Douglas Starnes, duckman_1991, Eamon Charles, edawine, Elazar, eli-bd, Enrico Maria De Angelis, Erica, Erica, ericdwang, Erik Godard, EsmaeelE, Filip Haglund, Firix, fox, Franck Dernoncourt, Fred Barclay, Freddy, Gerard Roche, glS, GoatsWearHats, GThamizh, H. Pauwelyn, hardmooth, hayalci, hichris123, Ian, IanAuld, icesin, Igor Raush, Ilyas Mimouni, itsthejoker, J F, Jabba, jalanb, James, James Taylor, Jean-Francois T., jedwards, Jeffrey Lin, jfunez, JGreenwell, Jim Fasarakis Hilliard, jim opleydulven, jimsug, jmunsch, Johan Lundberg, John Donner, John Slegers, john400, jonrsharpe, Joseph True, JRodDynamite, jtbandes, Juan T, Kamran Mackey, Karan Chudasama, KerDam, Kevin Brown, Kiran Vemuri, kisanme, Lafexlos, Leon, Leszek Kicior, LostAvatar, Majid, manu, MANU, Mark Miller, Martijn Pieters, Mathias711, matsjoyce, Matt, Mattew Whitt, mdegis, Mechanic, Media, mertyildiran, metahost, Mike Driscoll, MikJR, Miljen Mikic, mnoronha, Morgoth, moshemeirelles, MSD, MSeifert, msohng, msw, muddyfish, Mukund B, Muntasir Alam, Nathan Arthur, Nathaniel Ford, Ned Batchelder, Ni., niyasc, noɥʎd ʎzɐɹƆ, numbermaniac, orvi, Panda, Patrick Haugh, Pavan Nath, Peter Masiar, PSN, PsyKzz, pylang, pzp, Qchmqs, Quill, Rahul Nair, Rakitić, Ram Grandhi, rfkortekaas, rick112358, Robotski, rrao, Ryan Hilbert, Sam Krygsheld, Sangeeth Sudheer, SashaZd, Selcuk, Severiano Jaramillo Quintanar, Shiven, Shoe, Shog9, Sigitas Mockus, Simplans, Slayther, stark, StuxCrystal, SuperBiasedMan, Snađoшƒa, taylor swift, techydesigner, Tejus Prasad, TerryA, The_Curry_Man, TheGenie OfTruth, Timotheus.Kampik, tjohnson, Tom Barron, Tom de Geus, Tony Suffolk 66, tonyo, TPVasconcelos, user2314737, user2853437, user312016, Utsav T, vaichidrewar, vasili111, Vin, W.Wong, weewooquestionaire,

		Will, wintermute, Yogendra Sharma, Zach Janicki, Zags
2	* args** kwargs	cjds, Eric Zhang, ericmarkmartin, Geeklhem, J F, Jeff Hutchins , Jim Fasarakis Hilliard, JuanPablo, kdopen, loading..., Marlon Abeykoon, Mattew Whitt, Pasha, pcurry, PsyKzz, Scott Mermelstein, user2314737, Valentin Lorentz, Veedrac
3	__name__	Annonymous, BusyAnt, Christian Ternus, jonrsharpe, Lutz Prechelt, Steven Elliott
4	2to3	Alessandro Trinca Tornidor, Dartmouth, Firix, Kevin Brown, Naga2Raja, Stephen Leppik
5	ArcPy	Midavalo, PolyGeo, Zhanping Shi
6	Asyncio	2Cubed, Alessandro Trinca Tornidor, Cimbali, hiro protagonist, obust, pylang, RamenChef, Seth M. Larson, Simplans, Stephen Leppik, Udi
7	base64	Thomas Gerot
8	ChemPy - python	Biswa_9937
9	configparser	Chinmay Hegde, Dunatotatos
10	ctypes	Or East
11	Deque	Anthony Pham, BusyAnt, matsjoyce, ravigadila, Simplans, Thomas Ahle, user2314737
12	dis	muddyfish, user2314737
13	Django	code_geek, orvi
14	Functools	Alessandro Trinca Tornidor, enrico.bacis, flamenco, RamenChef, Shrey Gupta, Simplans, Stephen Leppik, StuxCrystal
15	hashlib	Mark Omo, xiaoyi
16	Heapq	ettanany
17	HTML	alecxe, talhasch
18	ijson	Prem Narain
19	IterablesIterators	4444, Conrad.Dean, demonplus, Ilia Barahovski, Pythonista
20	Itertools	ADITYA, Alessandro Trinca Tornidor, Andy Hayden, balki, bpachev, Ffisegydd, jackskis, Julien Spronck, Kevin Brown, machine yearning, nlsdfnbch, pylang, RahulHP, RamenChef,

		Simplans, Stephen Leppik, Symmitchry, Wickramaranga, wnnmaw
21	JSON	Indradhanush Gupta, Leo, Martijn Pieters, pzp, theheadofabroom, Underyx, Wolfgang
22	kivy - NUIPython	dhimanta
23	Neo4jCypherPy2Neo	Wingston Sharon
24	os.path	Claudiu, Fábio Perez, girish946, Jmills, Szabolcs Dombi, VJ Magar
25	os	Andy, Christian Ternus, JelmerS, JL Peyret, mnoronha, Vinzee
26	Pandas Transform	Dee
27	Pickle	J F, Majid, Or East, RahulHP, rfkortekaas, zvone
28	pipPyPI	Andy, Arpit Solanki, Community, InitializeSahib, JNat, Mahdi, Majid, Matt Giltaji, Nathaniel Ford, Rápli András, SerialDev, Simplans, Steve Barnes, StuxCrystal, tlo
29	PostgreSQL	Alessandro Trinca Tornidor, RamenChef, Stephen Leppik, user2027202827
30	py.test	Andy, Claudiu, Ffisegydd, Kinifwyne, Matt Giltaji
31	pyaudio	Biswa_9937
32	pyautogui	Damien, Rednivrug
33	pygame	Anthony Pham, Aryaman Arora, Pavan Nath
34	Pyglet	Comrade SparklePony, Stephen Leppik
35	PyInstaller - Python	ChaoticTwist, Eric, mnoronha
36	Python HTTP Server	Arpit Solanki, J F, jmunsch, Justin Chadwell, Mark, MervS, orvi , quantummind, Raghav, RamenChef, Sachin Kalkur, Simplans , techydesigner
37	Python Lex-Yacc	CLDSEED
38	Python	David Heyman, Faiz Halde, Iván Rodríguez Torres, J F, Thomas Moreau, Tyler Gubala
39	PythonMutable vs ImmutableHashable	Cilyan
40	Python	Imran Bughio, mvis89, Rednivrug

41	PythonShell	mnoronha, Shijo
42	python	4444, Alessandro Trinca Tornidor, Fred Barclay, RamenChef, Ricardo, Stephen Leppik
43	Pythonpyserial	Alessandro Trinca Tornidor, Ani Menon, girish946, mnoronha, Saranjith, user2314737
44	Python	Alessandro Trinca Tornidor, Annonymous, eenblam, Mahmoud Hashemi, RamenChef, Stephen Leppik
45	PythonExcel	bee-sting, Chinmay Hegde, GiantsLoveDeathMetal, hackvan, Majid, talhasch, user2314737, Will
46	Python	RamenChef, user2728397
47	Python	Gavin, lorenzofeliz, Pike D., Rednivrug
48	Python	Nick Humrich
49	PythonCurses	4444, Guy, kollery, Vinzee
50	Python	ADITYA, Antonio, Elodin, Neil A., Vinzee
51	Python	atayenel, ChaoticTwist, David, Geeklhem, mattgathu, mnoronha, thsecmaniac
52	Python - virtualenv	Vikash Kumar Jain
53	Python	Ken Y-N, RandomHash
54	setup.py	Adam Brenecki, amblina, JNat, ravigadila, strpeter, user2027202827, Y0da
55	Sqlite3	Chinmay Hegde, Simplans
56	SYS	blubberdiblub
57	tempfile NamedTemporaryFile	Alessandro Trinca Tornidor, amblina, Kevin Brown, Stephen Leppik
58	Tkinter	Dartmouth, rlee827, Thomas Gerot, TidB
59	Unicode	Claudiu, KeyWeeUsr
60	WebWSGI	David Heyman, Kevin Brown, Preston, techydesigner
61	Web	Thomas Gerot
62	“with”Statement	Abhijeet Kasurde, Alessandro Trinca Tornidor, Andy Hayden, Antoine Bolvy, carrdelling, Conrad.Dean, Dartmouth, David

		Marx, DeepSpace, Elazar, Kevin Brown, magu_, Majid, Martijn Pieters, Matthew, nlsdfnbch, Pasha, Peter Brittain, petrs, Shuo, Simplans, SuperBiasedMan, The_Cthulhu_Kid, Thomas Gerot, tyteen4a03, user312016, Valentin Lorentz, vaultah, λuser
63	Python 2Python 3	671620616, Abhishek Kumar, Akshit Soota, Alex Gaynor, Allan Burleson, Alleo, Amarpreet Singh, Andy Hayden, Ani Menon, Antoine Bolvy, AntsySysHack, Antti Haapala, Antwan, arekolek , Ares, asmeurer, B8vrede, Bakuriu, Bharel, Bhargav Rao, bignose, bitchaser, Bluethon, Cache Staheli, Cameron Gagnon , Charles, Charlie H, Chris Sprague, Claudiu, Clayton Wahlstrom, cLDSEED, Colin Yang, Cometsong, Community, Conrad.Dean, danidee, Daniel Stradowski, Darth Shadow, Dartmouth, Dave J, David Cullen, David Heyman, deeenes, DeepSpace, Delgan, DoHe, Duh-Wayne-101, Dunno, dwanderson, Ekeyme Mo, Elazar, enderland, enrico.bacis, erewok, ericdwang, ericmarkmartin, Ernir, ettanany, Everyone_Else, evuez, Franck Dernoncourt, Fred Barclay, garg10may, Gavin, geoffspear, ghostarbeiter, GoatsWearHats, H. Pauwelyn, Haohu Shen, holdenweb, iScrE4m, Iván C., J F, J. C. Leitão, James Elderfield, James Thiele, jarondl, jedwards, Jeffrey Lin, JGreenwell, Jim Fasarakis Hilliard, Jimmy Song, John Slegers, Jojodmo, jonrsharpe, Josh, Juan T, Justin, Justin M. Ucar, Kabie, kamalbanga, Karl Knechtel, Kevin Brown, King's jester, Kunal Marwaha, Lafexlos, lenz, linkdd, l'L'l, Mahdi, Martijn Pieters, Martin Thoma, masnun, Matt, Matt Dodge, Matt Rowland, Mattew Whitt, Max Feng, mgwilliams, Michael Recachinas, mkj, mnoronha, Moinuddin Quadri, muddyfish, Nathaniel Ford, niemmi, niyasc, noɥʎdʎzɐɹƆ, OrangeTux, Pasha, Paul Weaver, Paulo Freitas, pcurry, pktangyue, poppie, pylang, python273, Pythonista, RahulHP, Rakitić, RamenChef, Rauf, René G, rfkortekaas, rrao, Ryan, sblair, Scott Mermelstein, Selcuk, Serenity, Seth M. Larson, ShadowRanger, Simplans, Slayther, solarc, sricharan, Steven Hewitt, sth, SuperBiasedMan, Tadhg McDonald-Jensen, techydesigner, Thomas Gerot, Tim, tobias_k, Tyler, tyteen4a03 , user2314737, user312016, Valentin Lorentz, Veedrac, Ven, Vinayak, Vlad Shcherbina, VPfB, WeizhongTu, Wieland, wim, Wolf, Wombatz, xtreak, zarak, zcb, zopieux, zurfyx, zvezda
64	intfloatstrtuple frozensets	Alessandro Trinca Tornidor, FazeL, Ganesh K, RamenChef, Stephen Leppik
65		Akshat Mahajan, Dair, Franck Dernoncourt, J F, Mahdi, nlsdfnbch, Ryan Smith, Vinzee, Xavier Combelle
66		Eleftheria, evuez, mnoronha

67		Jeremy, Mohammed Salman
68	“pip”PyPI	Zydnar
69	AMQPStorm RabbitMQ	eandersson
70	Pandas	Mark Miller
71	PythonWindows	Simon Hibbs
72	PythonRaspberry PI	dhimanta
73	PythonWeb	alecxe, Amitay Stern, jmunsch, mrtuovinen, Ni., RamenChef, Saiful Azad, Saqib Shamsi, Simplans, Steven Maude, sth, sytech, talhasch, Thomas Gerot
74	Python	Aquib Javed Khan, Arun, ChaoticTwist, cledoux, Ffisegydd, ifma
75	ZIP	Chinmay Hegde, ghostarbeiter, Jeffrey Lin, SuperBiasedMan
76		zenlc2000
77		Adrian Antunez, Alessandro Trinca Tornidor, Alfe, Andy, Benjamin Hodgson, Brian Rodriguez, BusyAnt, Claudiu, driax, Elazar, flazzarini, ghostarbeiter, Ilia Barahovski, J F, Marco Pashkov, muddyfish, noɥʇʎdʎzɐɹƆ, Paul Weaver, Rahul Nair, RamenChef, Shawn Mehan, Shiven, Shkelqim Memolla, Simplans, Slickytail, Stephen Leppik, Sudip Bhandari, SuperBiasedMan, user2314737
78		Anthony Pham, Antoine Bolvy, BusyAnt, Community, Elazar, James, Jim Fasarakis Hilliard, Joab Mendes, Majid, Md.Sifatul Islam, Mechanic, mezzode, nlsdfnbch, noɥʇʎdʎzɐɹƆ, Selcuk, Simplans, textshell, tobias_k, Tony Suffolk 66, user2314737
79		2Cubed, Amir Rachum, Antoine Pinsard, Camsbury, Community, driax, Igor Raush, InitializeSahib, Marco Pashkov, Martijn Pieters, Mattew Whitt, OozeMeister, Pasha, Paulo Scardine, RamenChef, Rob Bednark, Simplans, sisanared, zvone
80		rassar
81	CLI	Alessandro Trinca Tornidor, anatoly techtonik, Darth Shadow
82		Anthony Pham, intboolstring, jtbandes, Luke Taylor, MSeifert, Pasha, supersam654

83		Alessandro Trinca Tornidor, JGreenwell, metahost, Pigman168 , RamenChef, Stephen Leppik
84		Greg, JakeD
85		3442, 4444, acdr, Ahsanul Haque, Akshay Anand, Akshit Soota, Alleo, Amir Rachum, André Laszlo, Andy Hayden, Ankit Kumar Singh, Antoine Bolvy, APerson, Ashwinee K Jha, B8vrede, bfontaine, Brian Cline, Brien, Casebash, Celeo, cfi, ChaoticTwist, Charles, Charlie H, Chong Tang, Community, Conrad.Dean, Dair, Daniel Stradowski, Darth Shadow, Dartmouth, David Heyman, Delgan, Dima Tisnek, eenblam, Elazar, Emma, enrico.bacis, EOL, ericdwang, ericmarkmartin, Esteis, Faiz Halde, Felk, Fermi paradox, Florian Bender, Franck Dernoncourt, Fred Barclay, freidrichen, G M, Gal Dreiman, garg10may, ghostarbeiter, GingerHead, griswolf, Hannele, Harry, Hurkyl, IanAuld, iankit, Infinity, intboolstring, J F, J0HN, James, JamesS, Jamie Rees, jedwards, Jeff Langemeier, JGreenwell, JHS, jjwatt, JKillian, JNat, joel3000, John Slegers, Jon, jonrsharpe, Josh Caswell, JRodDynamite, Julian, justhalf, Kamyar Ghasemlou, kdopen, Kevin Brown, KIDJourney, Kwarrtz, Lafexlos, lapis, Lee Netherton, Liteye, Locane, Lyndsy Simon, machine yearning, Mahdi, Marc, Markus Meskanen, Martijn Pieters, Matt, Matt Giltaji, Matt S, Mattew Whitt, Maximillian Laumeister, mbrig, Mirec Miskuf, Mitch Talmadge, Morgan Thrapp, MSeifert, muddyfish, n8henrie, Nathan Arthur, nehemiah, noɥʎɟʎzɐɹƆ, Or East, Ortomala Lokni, pabouk, Panda, Pasha, pktangyue, Preston, Pro Q, pylang, R Nar, Rahul Nair, rap-2-h, Riccardo Petraglia, rll, Rob Fagen, rrao, Ryan Hilbert, Ryan Smith, ryanyuyu, Samuel McKay, sarvajeetsuman, Sayakiss, Sebastian Kreft, Shoe, SHOWMEWHATYOUGOT, Simplans, Slayther, Slickytail, solidcell, StuxCrystal, sudo bangbang, Sunny Patel, SuperBiasedMan, syb0rg, Symmitchry, The_Curry_Man, theheadofabroom, Thomas Gerot, Tim McNamara, Tom Barron, user2314737, user2357112, Utsav T, Valentin Lorentz, Veedrac, viveksyngh, vog, W.P. McNeill, Will, Will, Wladimir Palant, Wolf, XCoder Real, yurib, Yury Fedorov, Zags, Zaz
86		J F, sth, zmo
87		J F, keiv.fly, SashaZd
88	Python	Claudiu, KeyWeeUsr, Marco Pashkov, pylang, SuperBiasedMan, Thtu
89		Adriano, Akshat Mahajan, AlexV, Andy, Andy Hayden, Anthony Pham, Arkady, B8vrede, Benjamin Hodgson, btel,

		CamelBackNotation, Camsbury, Chandan Purohit, ChaoticTwist, Charlie H, Chris Larson, Community, D. Alveno, danidee, DawnPaladin, Delgan, duan, duckman_1991, elegent, Elodin, Emma, EsmaeelE, Ffisegydd, Gal Dreiman, ghostarbeiter, Hurkyl, J F, James, Jeffrey Lin, JGreenwell, Jim Fasarakis Hilliard, jkitchen, Jossie Calderon, Justin, Kevin Brown, L3viathan, Lee Netherton, Martijn Pieters, Martin Thurau, Matt Giltaji, Mike - SMT, Mike Driscoll, MSeifert, muddyfish, Murphy4, nd., noɥʎdʎzɐɹƆ, Pasha, pylang, pzp, Rahul Nair, Severiano Jaramillo Quintanar, Simplans, Slayther, Steve Barnes, Steven Maude, SuperBiasedMan, textshell, then0rTh, Thomas Gerot, user2314737, user3333708, user405, Utsav T, vaultah, Veedrac, Will, Will, zxxz, λuser
90		Aaron Christiansen, David, Elazar, Peter Shinners, ppperry
91		Anthony Pham, davidism, Elazar, Esteis, Mike Driscoll, SuperBiasedMan, user2314737, zvone
92		Adriano, Alexander, Anthony Pham, Ares, Barry, blueenvelope, Bosoneando, BusyAnt, Çağatay Uslu, caped114, Chandan Purohit, ChaoticTwist, cizixs, Daniel Porteous, Darth Kotik, deeenes, Delgan, Elazar, Ellis, Emma, evuez, exhuma, Ffisegydd, Flickerlight, Gal Dreiman, ganesh gadila, ghostarbeiter, Igor Raush, intboolstring, J F, j3485, jalanb, James, James Elderfield, jani, jimsug, jkdev, JNat, jonrsharpe, KartikKannapur, Kevin Brown, Lafexlos, LDP, Leo Thumma, Luke Taylor, lukewrites, lxer, Majid, Mechanic, MrP01, MSeifert, muddyfish, n12312, noɥʎdʎzɐɹƆ, Oz Bar-Shalom, Pasha, Pavan Nath, poke, RamenChef, ravigadila, ronrest, Serenity, Severiano Jaramillo Quintanar, Shawn Mehan, Simplans, sirin, solarc, SuperBiasedMan, textshell, The_Cthulhu_Kid, user2314737, user6457549, Utsav T, Valentin Lorentz, vaultah, Will, wythagoras, Xavier Combelle
93		Alireza Savand, Ami Tavory, antimatter15, Arpit Solanki, bijancn, Claudiu, Dartmouth, engineercoding, Ffisegydd, J F, JGreenwell, jmunsch, joel3000, Kevin Brown, Kinifwyne, Mario Corchero, Matt Giltaji, Mattew Whitt, mgilson, muddyfish, pylang, strpeter
94	GIL	Scott Mermelstein
95		xiaoyi
96	windows virtualenvwrapper	Sirajus Salayhin

97		naren
98		APerson, cfi, Igor Raush, Jon Ericson, Karl Knechtel, Marco Pashkov, MSeifert, noɥʎɟzɐɹƆ, Parousia, Simplans, SuperBiasedMan, tlama, user2314737
99		bogdanciobanu, Claudiu, Conrad.Dean, Elazar, FazeL, J F, James Elderfield, lukess, muddyfish, Sam Whited, SiggyF, Stephen Leppik, SuperBiasedMan, Xavier Combelle
100		Doraemon, GoatsWearHats, J F, JNat, Marco Pashkov, Mark Miller, Martijn Pieters, Nathaniel Ford, Nicolás, pcurry, pzp, SashaZd, SuperBiasedMan, Vilmar
101		ADITYA, boboquack, Chromium, cjds, depperm, Hannes Karppila, JGreenwell, Jonatan, kdopen, OliPro007, orvi, SashaZd, Sнađошƒa, textshell, Thomas Ahle, user2314737
102		Alon Alexander, Nander Speerstra, unutbu, Vinzee, Will
103		Benedict Bunting, DeepSpace, depperm, Simplans, skrrgwasme, Vinzee
104		boboquack, Buzz, rrao
105		Alu, cLDSEED, juggernaut, Kevin Brown, Kristof, mattgathu, Nabeel Ahmed, nlsdfnbch, Rahul, Rahul Nair, Riccardo Petraglia, Thomas Gerot, Will, Yogendra Sharma
106		David Cullen, Dev, MattCorr, nlsdfnbch, Rob H, StuxCrystal, textshell, Thomas Gerot, Will
107		Adam Matan, Andrew Schade, Brendan Abel, jfs, jmunsch, Riccardo Petraglia
108		Amir Rachum, Anthony Pham, APerson, ArtOfCode, BoppreH, Burhan Khalid, Chris Mueller, cizixs, depperm, Ffisegydd, Gareth Latty, Guy, helpful, iBelieve, Igor Raush, Infinity, James , JGreenwell, jonrsharpe, Karsten 7., kdopen, machine yearning, Majid, mattgathu, Mechanic, MSeifert, muddyfish, Nathan, nlsdfnbch, noɥʎɟzɐɹƆ, ronrest, Roy Iacob, Shawn Mehan, Simplans, SuperBiasedMan, TehTris, Valentin Lorentz, viveksyngh, Xavier Combelle
109		Amitay Stern, Andy Hayden, Ares, Bhargav Rao, Brien, BusyAnt, Cache Staheli, caped114, ChaoticTwist, Charles, Dartmouth, David Heyman, depperm, Doug Henderson, Elazar , ganesh gadila, ghostarbeiter, GoatsWearHats, idjaw, Igor Raush, Ilia Barahovski, j__, Jim Fasarakis Hilliard, JL Peyret,

		Kevin Brown, krato, MarkyPython, Metasomatism, Mikail Land, MSeifert, mu , Nathaniel Ford, OliPro007, orvi, pzp, ronrest, Shrey Gupta, Simplans, SuperBiasedMan, theheadofabroom, user1349663, user2314737, Veedrac, WeizhongTu, wnnmaw
110		4444, Aaron Christiansen, Adam_92, ADITYA, Akshit Soota, aldanor, alecxe, Alessandro Trinca Tornidor, Andy Hayden, Ani Menon, B8vrede, Bahrom, Bhargav, Charles, Chris, Darth Shadow, Dartmouth, Dave J, Delgan, dreftymac, evuez, Franck Dernoncourt, Gal Dreiman, gerrit, Giannis Spiliopoulos, GiantsLoveDeathMetal, goyalankit, Harrison, James Elderfield, Jean-Francois T., Jeffrey Lin, jetpack_guy, JL Peyret, joel3000, Jonatan, JRodDynamite, Justin, Kevin Brown, knight, krato, Marco Pashkov, Mark, Matt, Matt Giltaji, mu , MYGz, Nander Speerstra, Nathan Arthur, Nour Chawich, orion_tvv, ragesz, SashaZd, Serenity, serv-inc, Simplans, Slayther, Sometowngeek, SuperBiasedMan, Thomas Gerot, tobias_k, Tony Suffolk 66, UloPe, user2314737, user312016, Vin, zondo
111		adeora, ArtOfCode, BSL-5, Kevin Brown, matsjoyce, SuperBiasedMan, Thomas Gerot, Wladimir Palant, wrwrwr
112	/	Mohammad Julfikar
113		Dartmouth, J F, mattgathu, Nathan Osman, techydesigner, ygram
114	PythonSQL Server	metmirr
115		angussidney, Anthony Pham, Antonis Kalou, Brett Cannon, BusyAnt, Casebash, Christian Ternus, Community, Conrad.Dean, Daniel, Dartmouth, Esteis, Ffisegydd, FMc, Gerard Roche, Gideon Buckwalter, J F, JGreenwell, Kinifwyne, languitar, Lex Scarisbrick, Matt Giltaji, MSeifert, niyasc, nlsdfnbch, Paulo Freitas, pylang, Rahul Nair, Saiful Azad, Serenity, Simplans, StardustGogeta, StuxCrystal, SuperBiasedMan, techydesigner, the_cat_lady, Thomas Gerot, Tony Meyer, Tushortz, user2683246, Valentin Lorentz, Valor Naram, vaultah, wnnmaw
116		Alessandro Trinca Tornidor, Darth Shadow, DhiaTN, J F, Jacques de Hooge, Leo, Martijn Pieters, mnoronha, Priya, RamenChef, Stephen Leppik
117		Elazar, SashaZd, SuperBiasedMan
118		boboquack, Brett Cannon, Dair, Ffisegydd, John Zwinck, Severiano Jaramillo Quintanar, Steven Maude

119		abukaj, ADITYA, Alec, Alessandro Trinca Tornidor, Alex, Antoine Bolvy, Baaing Cow, Bhargav Rao, Billy, bixel, Charles, Cheney, Christophe Roussy, Dartmouth, DeepSpace, DhiaTN, Dilettant, fox, Fred Barclay, Gerard Roche, greatwolf, hiro protagonist, Jeffrey Lin, JGreenwell, Jim Fasarakis Hilliard, Lafexlos, maazza, Malt, Mark, matsjoyce, Matt Dodge, MervS, MSeifert, ncmathsadist, omgimanerd, Patrick Haugh, pylang, RamenChef, Reut Sharabani, Rob Bednark, rrao, SashaZd, Shihab Shahriar, Simplans, SuperBiasedMan, Tim D, Tom Dunbavan, tyteen4a03, user2314737, Will Vousden, Wombatz
120	CPython	Julij Jegorov
121		davidism, J F, zmo, Валерий Павлов
122	CSV	Hriddhi Dey, Thomas Crowley
123		Adriano, Alex L, alfonso.kim, Alleo, Anthony Pham, Antti Haapala, Chris Hunt, Christian Ternus, Darth Kotik, DeepSpace, Delgan, DhiaTN, ebo, Elazar, Eric Finn, Felix D., Ffisegydd, Gal Dreiman, Generic Snake, ghostarbeiter, GoatsWearHats, Guy, Inbar Rose, intboolstring, J F, James, Jeffrey Lin, JGreenwell, Jim Fasarakis Hilliard, jrast, Karl Knechtel, machine yearning, Mahdi, manetsus, Martijn Pieters, Math, Mathias711, MSeifert, pnhgiol, rajah9, Rishabh Gupta, Ryan, sarvajeetsuman, sevenforce, SiggyF, Simplans, skrrgwasme, SuperBiasedMan, textshell, The_Curry_Man, Thomas Gerot, Tom, Tony Suffolk 66, user1349663, user2314737, Vinzee, Will
124		A. Ciclet, RamenChef, user2314737
125		Benjamin Hodgson, Elazar, Faiz Halde, J F, Lee Netherton, loading..., Mister Mister
126		Beall619, Frustrated, Justin, Leon Z., lukewrites, SuperBiasedMan, Valentin Lorentz
127	abc	Akshat Mahajan, Alessandro Trinca Tornidor, JGreenwell, Kevin Brown, Mattew Whitt, mkrieger1, SashaZd, Stephen Leppik
128		Teepeemm
129		Abhishek Jain, boboquack, Charles, Gal Dreiman, intboolstring , JakeD, JNat, Kevin Brown, Matías Brignone, nemesisfixx, poke, R Colmenares, Shawn Mehan, Simplans, Thomas Gerot, tmr232, Tony Suffolk 66, viveksyngh

130		Antti Haapala, APerson, GoatsWearHats, Mirec Miskuf, MSeifert, RamenChef, Simplans, Valentin Lorentz
131		bbayles, cizixs, Nemo, pylang, SuperBiasedMan
132	/	naren
133		2Cubed, proprefenetre, pylang, rrao, Simon Hibbs, Simplans
134		Dan Sanderson, Igor Raush, MSeifert
135		MSeifert
136	XML	4444, Brad Larson, Chinmay Hegde, Francisco Guimaraes, greuze, heyhey2k, Rob Murray
137		Biswa_9937
138		asmeurer, Community, Elazar, jmunsch, kon psych, Marco Pashkov, MSeifert, RamenChef, Shawn Mehan, Simplans, Steven Maude, Symmitchry, void, XCoder Real
139		Anthony Pham, ArtOfCode, asmeurer, Christofer Ohlsson, Ellis , fredley, ghostarbeiter, Igor Raush, intboolstring, J F, James Elderfield, JGreenwell, MSeifert, niyasc, RahulHP, rajah9, Simplans, StardustGogeta, SuperBiasedMan, yurib
140		Devesh Saini, Infinity, rfkortekaas
141		Alessandro Trinca Tornidor, Antonio, bee-sting, CLDSEED, D. Alveno, John Y, LostAvatar, mbsingh, Michel Touw, qwertyuip9 , RamenChef, rrawat, Stephen Leppik, Stephen Nyamweya, sumitroy, user2314737, valeas, zweiterlinde
142		Andy Hayden, MSeifert, Peter Mølgaard Pallesen, pylang
143		Andy, Pavan Nath, RamenChef, Vin
144	I / O.	Ajean, Anthony Pham, avb, Benjamin Hodgson, Bharel, Charles, crhodes, David Cullen, Dov, Esteis, ilse2005, isvforall, jfsturtz, Justin, Kevin Brown, mattgathu, MSeifert, nlsdfnbch, Ozair Kafray, PYPL, pzp, RamenChef, Ronen Ness, rrao, Serenity, Simplans, SuperBiasedMan, Tasdik Rahman, Thomas Gerot, Umibozu, user2314737, Will, WombatPM, xgord
145		DeepSpace, James
146		Ajean, alecxe, Andy, Antti Haapala, BusyAnt, Conrad.Dean, Elazar, ghostarbeiter, J F, Jeffrey Lin, jonrsharpe, Kevin Brown

		, Nicole White, nlsdfnbch, Ohad Eytan, Paul, paulmorriss, proprius, RahulHP, RamenChef, sagism, Simplans, Sirajus Salayhin, Suku, Will
147		surfthecity
148		Razik
149		Andy, Elazar, evuez, Martijn Pieters, techydesigner
150		Andy Hayden, BusyAnt, Chris Larson, deepakkt, Delgan, Elazar, evuez, Ffisegydd, Geeklhem, Hannes Karppila, James, Kevin Brown, krato, Max Feng, noɥʎɐʌɔ, rajah9, rrao, SashaZd, Simplans, Slayther, Soumendra Kumar Sahoo, Thomas Gerot, Trimax, Valentin Lorentz, Vinzee, wwii, xgord, Zack
151		DeepSpace, Simplans, tjohnson
152		Esteis, Marlon Abeykoon, mnoronha, PYPL
153		Aidan, alejosocorro, andandandand, Andy Hayden, ashes999, B8vrede, Claudiu, Darth Shadow, driax, Fermi paradox, ganesh gadila, goodmami, Jan, Jeffrey Lin, jonrsharpe, Julien Spronck, Kevin Brown, Md.Sifatul Islam, Michael M., mnoronha , Nander Speerstra, nrusch, Or East, orvi, regnarg, sarvajeetsuman, Simplans, SN Ravichandran KR, SuperBiasedMan, user2314737, zondo
154		Anthony Pham, Ares, Elazar, J F, MSeifert, Shawn Mehan, SuperBiasedMan, Will, Xavier Combelle
155		Doc, Rahul Nair, SashaZd
156		Stephen Leppik, Thomas Gerot
157	`exec``eval`	Antti Haapala, Ilja Everilä
158	Matplotlib	Arun, user2314737
159		Alessandro Trinca Tornidor, Beall619, mnoronha, RamenChef, Stephen Leppik, Sun Qingyao
160		2Cubed, Ahsanul Haque, Akshat Mahajan, Andy Hayden, Arthur Dent, ArtOfCode, Augustin, Barry, Chankey Pathak, Claudiu, CodenameLambda, Community, deeenes, Delgan, Devesh Saini, Elazar, ericmarkmartin, Ernir, ForceBru, Igor Raush, Ilia Barahovski, J0HN, jackskis, Jim Fasarakis Hilliard, Juan T, Julius Bullinger, Karl Knechtel, Kevin Brown, Kronen, Luc M, Lyndsy Simon, machine yearning, Martijn Pieters, Matt

		Giltaji, max, MSeifert, nlsdfnbch, Pasha, Pedro, PsyKzz, pzp, satsumas, sevenforce, Signal, Simplans, Slayther, StuxCrystal, tversteeg, Valentin Lorentz, Will, William Merrill, xtreak, Zaid Ajaj, zarak, λuser
161	urllib	Amitay Stern, ravigadila, sth, Will
162	WebSockets	2Cubed, Stephen Leppik, Tyler Gubala
163		amin, blueenvelope, Bryce Frank, Camsbury, David, DeepSpace, Elazar, J F, James, JGreenwell, Jon Ericson, Kevin Brown, Lafexlos, matsjoyce, Mechanic, Milo P, MSeifert, numbermaniac, sarvajeetsuman, Simplans, techydesigner, Tony Suffolk 66, Undo, user2314737, wythagoras, Zenadix
164		Alleo, amblina, Antoine Bolvy, Bonifacio2, Ffisegydd, Guy, Igor Raush, Jonatan, Martec, MSeifert, MUSR, pzp, RahulHP, Reut Sharabani, SashaZd, Sayed M Ahamad, SuperBiasedMan, theheadofabroom, user2314737, yurib
165		Andrzej Pronobis, Andy Hayden, Bahrom, Cimbali, Cody Piersall, Conrad.Dean, Elazar, evuez, J F, James, Or East, pylang, RahulHP, RamenChef, Simplans, user2314737
166		wim
167		Alessandro Trinca Tornidor, depperm, J F, JGreenwell, Matt Giltaji, Pasha, RamenChef, Stephen Leppik
168		Juan T, TemporalWolf
169		Adrian17, Artem Kolontay, ArtOfCode, Bhargav, brennan, Dair, Daniil Ryzhkov, Darkade, Darth Shadow, edwinksl, Fernando, ghostarbeiter, ha_1694, Hans Then, Iancnorden, J F, Majid, Marco Pashkov, Matt Giltaji, Mattew Whitt, nehemiah, Nuhil Mehdy, Ortomala Lokni, Preston, pylang, qwertyuip9, RamenChef, Régis B., Sebastian Schrader, Serenity, Shantanu Alshi, Shrey Gupta, Simon Fraser, Simplans, wrwrwr , ychaouche, zopieux, zvezda
170	virtualenvwrapper	Sirajus Salayhin
171		Alessandro Trinca Tornidor, ChaoticTwist, Community, Dair, doratheexplorer0911, Emolga, greut, iankit, JGreenwell, jonrsharpe, kefkius, Kevin Brown, Mattew Whitt, MSeifert, muddyfish, Mukunda Modell, Nearoo, Nemo, Nuno André, Pasha, Rob Bednark, seenu s, Shreyash S Sarnayak, Simplans, StuxCrystal, Suhas K, technusm1, Thomas Gerot, tyteen4a03, Wladimir Palant, zvone

172		hashcode55, StuxCrystal
173		Adeel Ansari, Bosoneando, bpachev
174		andrew
175		amblina, Braiam, Claudiu, cledoux, Elazar, Gerard Roche, krato, loading..., Marco Pashkov, Or Duan, Pasha, RamenChef , rfkortekaas, Simplans, Thomas Gerot, Topperfalkon, zmo, zondo
176		Gal Dreiman, Jörn Hees, sxnwlfkk
177	Python	muddyfish, StuxCrystal, user2314737
178		Charul, denvaar, djaszczurowski
179		Ani Menon, FunkySayu, MattCorr, SuperBiasedMan, TuringTux
180	PythonJavaScript	user2683246
181		Will, XonAether
182		Aldo, B8vrede, joel3000, Sardathrion, Sardorbek Imomaliev, Vlad Bezden
183	CSV	Adam Matan, Franck Dernoncourt, Martin Valgur, mnoronha, ravigadila, Setu
184		Andy Hayden, Darth Shadow, ericmarkmartin, Ffisegydd, Igor Raush, Jonas S, jonrsharpe, L3viathan, Majid, RamenChef, Simplans, Valentin Lorentz
185		alecxe, Annonymous, Antti Haapala, Elazar, Jim Fasarakis Hilliard, Jonatan, RamenChef, Seth M. Larson, Simplans, Stephen Leppik
186	...	Parousia, Thomas Gerot
187		Anaphory
188		Claudiu, Thomas Gerot
189		HoverHell, JGreenwell, MathSquared, SashaZd, Shreyash S Sarnayak
190		APerson, cfi, J Atkin, MSeifert, rajah9, SuperBiasedMan
191		Bastian, japborst, JGreenwell, Jossie Calderon, mbomb007, SashaZd, Tyler Crompton

192		FrankBr
193		Gal Dreiman, Iancnorden, Wayne Werner
194		Nemo
195		orvi
196	GZip	orvi
197		APerson, Igor Raush, Martijn Pieters, MSeifert
198		Prem Narain
199		Alex Gaynor, Andrzej Pronobis, Anthony Pham, Community, David Robinson, Delgan, giucal, Jim Fasarakis Hilliard, michaelrbock, MSeifert, Nobilis, ppperry, RamenChef, Simplans, SuperBiasedMan
200		Aaron Hall, Akshat Mahajan, Anthony Pham, Antti Haapala, Byte Commander, dermen, Elazar, Ellis, ericmarkmartin, Fermi paradox, Ffisegydd, japborst, Jim Fasarakis Hilliard, jonrsharpe , Justin, kramer65, Lafexlos, LDP, Morgan Thrapp, muddyfish, nico, OrangeTux, pcurry, Pythonista, Selcuk, Serenity, Tejas Jadhav, tobias_k, Vlad Shcherbina, Will
201	Python	Jacques de Hooge, Squidward
202		blueberryfields, Comrade SparklePony, frankyjuang, jmunsch, orvi, qwertyuip9, Stephen Leppik, Thomas Gerot
203		Aaron Hall, Ahsanul Haque, Akshat Mahajan, Andrzej Pronobis , Anthony Pham, Avantol13, Camsbury, cfi, Community, Conrad.Dean, Daksh Gupta, Darth Shadow, Dartmouth, depperm, Elazar, Ffisegydd, Haris, Igor Raush, InitializeSahib, J F, jkdev, jlarsch, John Militer, Jonas S, Jonathan, Kallz, KartikKannapur, Kevin Brown, Kinifwyne, Leo, Liteye, lmiguelvargasf, Mailerdaimon, Martijn Pieters, Massimiliano Kraus, Mattew Whitt, MrP01, Nathan Arthur, ojas mohril, Pasha, Peter Steele, pistache, Preston, pylang, Richard Fitzhugh, rohittk239, Rushy Panchal, Sempoo, Simplans, Soumendra Kumar Sahoo, SuperBiasedMan, techydesigner, then0rTh, Thomas Gerot, Tony Suffolk 66, tox123, UltraBob, user2314737, wrwrwr, Yogendra Sharma
204	__str____repr__	Alessandro Trinca Tornidor, jedwards, JelmerS, RamenChef, Stephen Leppik
205		Luca Van Oort, Stephen Leppik