Python Tutorial

3.7.2

Guido van Rossum
and the Python development team

12, 2019

Python Software Foundation
Email: docs@python.org

Contents

Python Python
Python Python https://www.python.org/ Python

Python C C++ C Python
Python Python
library-index reference-index C C++ extending-index c-api-index Python

Python Python Python library-index

CHAPTER 1

C/C++/Java

Python

Unix Windows shell GUI C/C++/Java Python Windows Mac OS X Unix

Python shell Python C “ ” Python Awk Perl

Python Python —— Python Tk

Python

Python Python C C++ Java

-
-
-

Python “ ” C Python Python C

Python “ ” BBC “Monty Python ” Monty Python

Python Python

Python

CHAPTER 2

Python

2.1

Python Python `/usr/local/bin/python3.7` ; `/usr/local/bin` Unix shell , :

```
python3.7
```

[1] Python `/usr/local/python`

Windows Python `C:\Python37` DOS :

```
set path=%path%;C:\python37
```

Unix `Control-D` Windows `Control-Z` 0 `quit()`

readline `Tab` `Control-P` “ ” `^P` `Backspace`

Unix tty

`python -c command [arg] ...` *command* `-c` Python *command*

Python `python -m module [arg] ...` *module*

`-i`

using-on-general

2.1.1

`sys` `argv` `import sys` `sys.argv[0]` `sys.argv[0]` `'-'` `-c` *command* `sys.argv[0]` `'-c'` `-m` *module* `sys.argv[0]` `-c` *command* `-m` *module* `sys.argv`

[1] Unix Python 3.x `python` Python 2.x

2.1.2

tty *interactive mode* *primary prompt* >>> ...

```
$ python3.7
Python 3.7 (default, Sep 16 2015, 09:25:04)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

if

```
>>> the_world_is_flat = True
>>> if the_world_is_flat:
...     print("Be careful not to fall off!")
...
Be careful not to fall off!
```

2.2

2.2.1

Python UTF-8 —— ASCII
UTF-8

```
# -*- coding: encoding -*-
```

encoding Python codecs

Windows-1252

```
# -*- coding: cp1252 -*-
```

UNIX "shebang"

```
#!/usr/bin/env python3
# -*- coding: cp1252 -*-
```

CHAPTER 3

Python

(>>> and ...)

Python # Python

:

```
# this is the first comment
spam = 1  # and this is the second comment
          # ... and now a third!
text = "# This is not a comment because it's inside quotes."
```

3.1 Python

Python >>>

3.1.1

“+“ - * /`` Pascal C ``() :

```
>>> 2 + 2
4
>>> 50 - 5*6
20
>>> (50 - 5*6) / 4
5.0
>>> 8 / 5  # division always returns a floating point number
1.6
```

“2“ 4 20 int “5.0“ 1.6 float

/ *floor division* // %

```
>>> 17 / 3  # classic division returns a float
5.666666666666667
>>>
>>> 17 // 3  # floor division discards the fractional part
5
>>> 17 % 3  # the % operator returns the remainder of the division
2
>>> 5 * 3 + 2  # result * divisor + remainder
17
```

Python ** [1]

```
>>> 5 ** 2  # 5 squared
25
>>> 2 ** 7  # 2 to the power of 7
128
```

= :

```
>>> width = 20
>>> height = 5 * 9
>>> width * height
900
```

( ) :

```
>>> n  # try to access an undefined variable
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'n' is not defined
```

Python :

```
>>> 4 * 3.75 - 1
14.0
```

_ Python :

```
>>> tax = 12.5 / 100
>>> price = 100.50
>>> price * tax
12.5625
>>> price + _
113.0625
>>> round(_, 2)
113.06
```

int float Python Decimal Fraction Python j J 3+5j

[1] ** - , -3**2 -(3**2) -9. 9, (-3)**2.

3.1.2

Python '.....' "....." [2] \ :

```
>>> 'spam eggs'  # single quotes
'spam eggs'
>>> 'doesn\'t'  # use \' to escape the single quote...
"doesn't"
>>> "doesn't"  # ...or use double quotes instead
"doesn't"
>>> '"Yes," they said.'
'"Yes," they said.'
>>> "\"Yes,\" they said."
'"Yes," they said.'
>>> '"Isn\'t," they said.'
'"Isn\'t," they said.'
```

print() :

```
>>> '"Isn\'t," they said.'
'"Isn\'t," they said.'
>>> print('"Isn\'t," they said.')
"Isn't," they said.
>>> s = 'First line.\nSecond line.'  # \n means newline
>>> s  # without print(), \n is included in the output
'First line.\nSecond line.'
>>> print(s)  # with print(), \n produces a new line
First line.
Second line.
```

\ r :

```
>>> print('C:\some\name')  # here \n means newline!
C:\some
ame
>>> print(r'C:\some\name')  # note the r before the quote
C:\some\name
```

"""...""" '''...''' \ :

```
print("""\
Usage: thingy [OPTIONS]
     -h                        Display this usage message
     -H hostname               Hostname to connect to
""")
```

:

```
Usage: thingy [OPTIONS]
     -h                        Display this usage message
     -H hostname               Hostname to connect to
```

+ * :

[2] , \n ('...') ("...") . " (\') .

```
>>> # 3 times 'un', followed by 'ium'
>>> 3 * 'un' + 'ium'
'ununium'
```

.

```
>>> 'Py' 'thon'
'Python'
```

:

```
>>> text = ('Put several strings within parentheses '
...         'to have them joined together.')
>>> text
'Put several strings within parentheses to have them joined together.'
```

:

```
>>> prefix = 'Py'
>>> prefix 'thon'  # can't concatenate a variable and a string literal
  File "<stdin>", line 1
    prefix 'thon'
           ^
SyntaxError: invalid syntax
>>> ('un' * 3) 'ium'
  File "<stdin>", line 1
    ('un' * 3) 'ium'
               ^
SyntaxError: invalid syntax
```

+ :

```
>>> prefix + 'thon'
'Python'
```

0 :

```
>>> word = 'Python'
>>> word[0]  # character in position 0
'P'
>>> word[5]  # character in position 5
'n'
```

:

```
>>> word[-1]  # last character
'n'
>>> word[-2]  # second-last character
'o'
>>> word[-6]
'P'
```

-0 0 -1

:

```
>>> word[0:2]  # characters from position 0 (included) to 2 (excluded)
'Py'
>>> word[2:5]  # characters from position 2 (included) to 5 (excluded)
'tho'
```

s[:i] + s[i:] s

```
>>> word[:2] + word[2:]
'Python'
>>> word[:4] + word[4:]
'Python'
```

0 :

```
>>> word[:2]   # character from the beginning to position 2 (excluded)
'Py'
>>> word[4:]   # characters from position 4 (included) to the end
'on'
>>> word[-2:]  # characters from the second-last (included) to the end
'on'
```

0 n n :

```
 +---+---+---+---+---+---+
 | P | y | t | h | o | n |
 +---+---+---+---+---+---+
 0   1   2   3   4   5   6
-6  -5  -4  -3  -2  -1
```

i j i j

word[1:3] 2.

:

```
>>> word[42]  # the word only has 6 characters
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: string index out of range
```

:

```
>>> word[4:42]
'on'
>>> word[42:]
''
```

Python *immutable* :

```
>>> word[0] = 'J'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
>>> word[2:] = 'py'
Traceback (most recent call last):
```

( )

( )

```
  File "<stdin>", line 1, in <module>
TypeError: 'str' object does not support item assignment
```

:

```
>>> 'J' + word[1:]
'Jython'
>>> word[:2] + 'py'
'Pypy'
```

len() :

```
>>> s = 'supercalifragilisticexpialidocious'
>>> len(s)
34
```

:

textseq

string-methods

f-strings

formatstrings str.format()

old-string-formatting %

3.1.3

Python :

```
>>> squares = [1, 4, 9, 16, 25]
>>> squares
[1, 4, 9, 16, 25]
```

sequence :

```
>>> squares[0]  # indexing returns the item
1
>>> squares[-1]
25
>>> squares[-3:]  # slicing returns a new list
[9, 16, 25]
```

( ) :

```
>>> squares[:]
[1, 4, 9, 16, 25]
```

:

```
>>> squares + [36, 49, 64, 81, 100]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

immutable , *mutable* :

```
>>> cubes = [1, 8, 27, 65, 125]  # something's wrong here
>>> 4 ** 3  # the cube of 4 is 64, not 65!
64
>>> cubes[3] = 64  # replace the wrong value
>>> cubes
[1, 8, 27, 64, 125]
```

append() ( ):

```
>>> cubes.append(216)  # add the cube of 6
>>> cubes.append(7 ** 3)  # and the cube of 7
>>> cubes
[1, 8, 27, 64, 125, 216, 343]
```

:

```
>>> letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g']
>>> letters
['a', 'b', 'c', 'd', 'e', 'f', 'g']
>>> # replace some values
>>> letters[2:5] = ['C', 'D', 'E']
>>> letters
['a', 'b', 'C', 'D', 'E', 'f', 'g']
>>> # now remove them
>>> letters[2:5] = []
>>> letters
['a', 'b', 'f', 'g']
>>> # clear the list by replacing all the elements with an empty list
>>> letters[:] = []
>>> letters
[]
```

len() :

```
>>> letters = ['a', 'b', 'c', 'd']
>>> len(letters)
4
```

( ), :

```
>>> a = ['a', 'b', 'c']
>>> n = [1, 2, 3]
>>> x = [a, n]
>>> x
[['a', 'b', 'c'], [1, 2, 3]]
>>> x[0]
['a', 'b', 'c']
>>> x[0][1]
'b'
```

3.2

Python ‘ <https://en.wikipedia.org/wiki/Fibonacci_number>‘_ :

```
>>> # Fibonacci series:
... # the sum of two elements defines the next
... a, b = 0, 1
>>> while a < 10:
...     print(a)
...     a, b = b, a+b
...
0
1
1
2
3
5
8
```

- : a b 0 1. ,
- while (: b < 10) Python C * * ,
  . C < ( ) > ( ) == ( ) <= ( ) >= ( ) != ( )
- Python Tab ,
- print() . ( ) print() , , , :

```
>>> i = 256*256
>>> print('The value of i is', i)
The value of i is 65536
```

end , :

```
>>> a, b = 0, 1
>>> while a < 1000:
...     print(a, end=',')
...     a, b = b, a+b
...
0,1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987,
```

CHAPTER 4

`while` Python

4.1 if

`if`

```
>>> x = int(input("Please enter an integer: "))
Please enter an integer: 42
>>> if x < 0:
...     x = 0
...     print('Negative changed to zero')
... elif x == 0:
...     print('Zero')
... elif x == 1:
...     print('Single')
... else:
...     print('More')
...
More
```

`elif` `else` 'elif' 'else if' `if ... elif ... elif ...` `switch` `case`

4.2 for

Python `for` C Pascal Python `for` Pascal
C :

```
>>> # Measure some strings:
... words = ['cat', 'window', 'defenestrate']
>>> for w in words:
...     print(w, len(w))
...
cat 3
window 6
defenestrate 12
```

```
>>> for w in words[:]:  # Loop over a slice copy of the entire list.
...     if len(w) > 6:
...         words.insert(0, w)
...
>>> words
['defenestrate', 'cat', 'window', 'defenestrate']
```

for w in words: defenestrate

4.3 range()

range() :

```
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
```

range(10) 10 10 range ' '

```
range(5, 10)
   5, 6, 7, 8, 9

range(0, 10, 3)
   0, 3, 6, 9

range(-10, -100, -30)
  -10, -40, -70
```

range() len() :

```
>>> a = ['Mary', 'had', 'a', 'little', 'lamb']
>>> for i in range(len(a)):
...     print(i, a[i])
...
0 Mary
1 had
```

( )

( )

```
2 a
3 little
4 lamb
```

enumerate()

range :

```
>>> print(range(10))
range(0, 10)
```

range()

for list()

```
>>> list(range(5))
[0, 1, 2, 3, 4]
```

4.4 break continue else

break C for while .

else (for) (while) break :

```
>>> for n in range(2, 10):
...     for x in range(2, n):
...         if n % x == 0:
...             print(n, 'equals', x, '*', n//x)
...             break
...     else:
...         # loop fell through without finding a factor
...         print(n, 'is a prime number')
...
2 is a prime number
3 is a prime number
4 equals 2 * 2
5 is a prime number
6 equals 2 * 3
7 is a prime number
8 equals 2 * 4
9 equals 3 * 3
```

else for if

else try else if : try else else break try

continue C :

```
>>> for num in range(2, 10):
...     if num % 2 == 0:
...         print("Found an even number", num)
```

( )

( )

```
...         continue
...     print("Found a number", num)
Found an even number 2
Found a number 3
Found an even number 4
Found a number 5
Found an even number 6
Found a number 7
Found an even number 8
Found a number 9
```

4.5 pass

pass :

```
>>> while True:
...     pass  # Busy-wait for keyboard interrupt (Ctrl+C)
...
```

:

```
>>> class MyEmptyClass:
...     pass
...
```

pass pass :

```
>>> def initlog(*args):
...     pass   # Remember to implement this!
...
```

4.6

Fibonacci :

```
>>> def fib(n):    # write Fibonacci series up to n
...     """Print a Fibonacci series up to n."""
...     a, b = 0, 1
...     while a < n:
...         print(a, end=' ')
...         a, b = b, a+b
...     print()
...
>>> # Now call the function we just defined:
... fib(2000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597
```

def

docstring

global

[1]

:

```
>>> fib
<function fib at 10042ed0>
>>> f = fib
>>> f(100)
0 1 1 2 3 5 8 13 21 34 55 89
```

fib return None None
print()

```
>>> fib(0)
>>> print(fib(0))
None
```

:

```
>>> def fib2(n):  # return Fibonacci series up to n
...     """Return a list containing the Fibonacci series up to n."""
...     result = []
...     a, b = 0, 1
...     while a < n:
...         result.append(a)    # see below
...         a, b = b, a+b
...     return result
...
>>> f100 = fib2(100)    # call it
>>> f100                # write the result
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

Python :

- return return None None
- result.append(a) result * ' ' obj.methodname obj methodname append() result = result + [a]

4.7

4.7.1

:

[1]

```
def ask_ok(prompt, retries=4, reminder='Please try again!'):
    while True:
        ok = input(prompt)
        if ok in ('y', 'ye', 'yes'):
            return True
        if ok in ('n', 'no', 'nop', 'nope'):
            return False
        retries = retries - 1
        if retries < 0:
            raise ValueError('invalid user response')
        print(reminder)
```

:

- ask_ok('Do you really want to quit?')
- ask_ok('OK to overwrite the file?', 2)
- ask_ok('OK to overwrite the file?', 2, 'Come on, only yes or no!')

in

```
i = 5

def f(arg=i):
    print(arg)

i = 6
f()
```

5

:

```
def f(a, L=[]):
    L.append(a)
    return L

print(f(1))
print(f(2))
print(f(3))
```

```
[1]
[1, 2]
[1, 2, 3]
```

:

```
def f(a, L=None):
    if L is None:
        L = []
    L.append(a)
    return L
```

4.7.2

kwarg=value :

```
def parrot(voltage, state='a stiff', action='voom', type='Norwegian Blue'):
    print("-- This parrot wouldn't", action, end=' ')
    print("if you put", voltage, "volts through it.")
    print("-- Lovely plumage, the", type)
    print("-- It's", state, "!")
```

voltage state, action type :

```
parrot(1000)                                          # 1 positional argument
parrot(voltage=1000)                                  # 1 keyword argument
parrot(voltage=1000000, action='VOOOOOM')             # 2 keyword arguments
parrot(action='VOOOOOM', voltage=1000000)             # 2 keyword arguments
parrot('a million', 'bereft of life', 'jump')         # 3 positional arguments
parrot('a thousand', state='pushing up the daisies')  # 1 positional, 1 keyword
```

:

```
parrot()                     # required argument missing
parrot(voltage=5.0, 'dead')  # non-keyword argument after a keyword argument
parrot(110, voltage=220)     # duplicate value for the same argument
parrot(actor='John Cleese')  # unknown keyword argument
```

actor parrot parrot(voltage=1000)
:

```
>>> def function(a):
...     pass
...
>>> function(0, a=0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: function() got multiple values for keyword argument 'a'
```

**name typesmapping *name *name
**name :

```
def cheeseshop(kind, *arguments, **keywords):
    print("-- Do you have any", kind, "?")
    print("-- I'm sorry, we're all out of", kind)
    for arg in arguments:
        print(arg)
    print("-" * 40)
    for kw in keywords:
        print(kw, ":", keywords[kw])
```

:

```
cheeseshop("Limburger", "It's very runny, sir.",
           "It's really very, VERY runny, sir.",
           shopkeeper="Michael Palin",
```

( )

( )

```
          client="John Cleese",
          sketch="Cheese Shop Sketch")
```

:

```
-- Do you have any Limburger ?
-- I'm sorry, we're all out of Limburger
It's very runny, sir.
It's really very, VERY runny, sir.
----------------------------------------
shopkeeper : Michael Palin
client : John Cleese
sketch : Cheese Shop Sketch
```

4.7.3

:

```
def write_multiple_items(file, separator, *args):
    file.write(separator.join(args))
```

*args ‘ ’ :

```
>>> def concat(*args, sep="/"):
...     return sep.join(args)
...
>>> concat("earth", "mars", "venus")
'earth/mars/venus'
>>> concat("earth", "mars", "venus", sep=".")
'earth.mars.venus'
```

4.7.4

range() *start* *stop* *- :

```
>>> list(range(3, 6))            # normal call with separate arguments
[3, 4, 5]
>>> args = [3, 6]
>>> list(range(*args))            # call with arguments unpacked from a list
[3, 4, 5]
```

** :

```
>>> def parrot(voltage, state='a stiff', action='voom'):
...     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 !
```

4.7.5 Lambda

lambda lambda a, b: a+b Lambda lambda :

```
>>> def make_incrementor(n):
...     return lambda x: x + n
...
>>> f = make_incrementor(42)
>>> f(0)
42
>>> f(1)
43
```

lambda :

```
>>> pairs = [(1, 'one'), (2, 'two'), (3, 'three'), (4, 'four')]
>>> pairs.sort(key=lambda pair: pair[1])
>>> pairs
[(4, 'four'), (1, 'one'), (3, 'three'), (2, 'two')]
```

4.7.6

Python Python
" " 8

:

```
>>> def my_function():
...     """Do nothing, but document it.
...
...     No, really, it doesn't do anything.
...     """
...     pass
...
>>> print(my_function.__doc__)
Do nothing, but document it.

    No, really, it doesn't do anything.
```

4.7.7

PEP 3107 PEP 484

__annotations__ -> def
:

```
>>> def f(ham: str, eggs: str = 'eggs') -> str:
...     print("Annotations:", f.__annotations__)
...     print("Arguments:", ham, eggs)
...     return ham + ' and ' + eggs
...
>>> f('spam')
Annotations: {'ham': <class 'str'>, 'return': <class 'str'>, 'eggs': <class 'str'>}
Arguments: spam eggs
'spam and eggs'
```

4.8

Python

Python PEP 8 Python

- 4

  4
- 79

- a = f(1, 2) + g(3, 4)
- CamelCase lower_case_with_underscores self
- Python UTF-8 ASCII
- ASCII

CHAPTER 5

5.1

`list.append(`*x*`)`
`a[len(a):] = [x]`

`list.extend(`*iterable*`)`
`a[len(a):] = iterable`

`list.insert(`*i*, *x*`)`
`a.insert(0, x)` `a.insert(len(a), x)` `a.append(x)`

`list.remove(`*x*`)`
x `ValueError`

`list.pop(`[*i*]`)`
`a.pop()` *i* Python)

`list.clear()`
`del a[:]`

`list.index(`*x*[, *start*[, *end*]]`)`
x `ValueError`

start *end* *start*

`list.count(`*x*`)`
x

`list.sort(`*key=None*, *reverse=False*`)`
`sorted()`

list.reverse()

list.copy()
a[:]

```
>>> fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
>>> fruits.count('apple')
2
>>> fruits.count('tangerine')
0
>>> fruits.index('banana')
3
>>> fruits.index('banana', 4)  # Find next banana starting a position 4
6
>>> fruits.reverse()
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'orange']
>>> fruits.append('grape')
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'orange', 'grape']
>>> fruits.sort()
>>> fruits
['apple', 'apple', 'banana', 'banana', 'grape', 'kiwi', 'orange', 'pear']
>>> fruits.pop()
'pear'
```

insert remove sort —— None [1] Python

5.1.1

“ ” append() pop()

```
>>> stack = [3, 4, 5]
>>> stack.append(6)
>>> stack.append(7)
>>> stack
[3, 4, 5, 6, 7]
>>> stack.pop()
7
>>> stack
[3, 4, 5, 6]
>>> stack.pop()
6
>>> stack.pop()
5
>>> stack
[3, 4]
```

[1] d->insert("a")->remove("b")->sort();

5.1.2

(“ ”) ( )

collections.deque

```
>>> from collections import deque
>>> queue = deque(["Eric", "John", "Michael"])
>>> queue.append("Terry")           # Terry arrives
>>> queue.append("Graham")          # Graham arrives
>>> queue.popleft()                 # The first to arrive now leaves
'Eric'
>>> queue.popleft()                 # The second to arrive now leaves
'John'
>>> queue                           # Remaining queue in order of arrival
deque(['Michael', 'Terry', 'Graham'])
```

5.1.3

```
>>> squares = []
>>> for x in range(10):
...     squares.append(x**2)
...
>>> squares
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

x for

```
squares = list(map(lambda x: x**2, range(10)))
```

```
squares = [x**2 for x in range(10)]
```

for for if for if :

```
>>> [(x, y) for x in [1,2,3] for y in [3,1,4] if x != y]
[(1, 3), (1, 4), (2, 3), (2, 1), (2, 4), (3, 1), (3, 4)]
```

```
>>> combs = []
>>> for x in [1,2,3]:
...     for y in [3,1,4]:
...         if x != y:
...             combs.append((x, y))
...
>>> combs
[(1, 3), (1, 4), (2, 3), (2, 1), (2, 4), (3, 1), (3, 4)]
```

for if

(x, y)

```
>>> vec = [-4, -2, 0, 2, 4]
>>> # create a new list with the values doubled
>>> [x*2 for x in vec]
[-8, -4, 0, 4, 8]
>>> # filter the list to exclude negative numbers
>>> [x for x in vec if x >= 0]
[0, 2, 4]
>>> # apply a function to all the elements
>>> [abs(x) for x in vec]
[4, 2, 0, 2, 4]
>>> # call a method on each element
>>> freshfruit = ['  banana', '  loganberry ', 'passion fruit  ']
>>> [weapon.strip() for weapon in freshfruit]
['banana', 'loganberry', 'passion fruit']
>>> # create a list of 2-tuples like (number, square)
>>> [(x, x**2) for x in range(6)]
[(0, 0), (1, 1), (2, 4), (3, 9), (4, 16), (5, 25)]
>>> # the tuple must be parenthesized, otherwise an error is raised
>>> [x, x**2 for x in range(6)]
  File "<stdin>", line 1, in <module>
    [x, x**2 for x in range(6)]
               ^
SyntaxError: invalid syntax
>>> # flatten a list using a listcomp with two 'for'
>>> vec = [[1,2,3], [4,5,6], [7,8,9]]
>>> [num for elem in vec for num in elem]
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
>>> from math import pi
>>> [str(round(pi, i)) for i in range(1, 6)]
['3.1', '3.14', '3.142', '3.1416', '3.14159']
```

5.1.4

3x4 3 4

```
>>> matrix = [
...     [1, 2, 3, 4],
...     [5, 6, 7, 8],
...     [9, 10, 11, 12],
... ]
```

```
>>> [[row[i] for row in matrix] for i in range(4)]
[[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]
```

for :

```
>>> transposed = []
>>> for i in range(4):
...     transposed.append([row[i] for row in matrix])
...
>>> transposed
[[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]
```

```
>>> transposed = []
>>> for i in range(4):
...     # the following 3 lines implement the nested listcomp
...     transposed_row = []
...     for row in matrix:
...         transposed_row.append(row[i])
...     transposed.append(transposed_row)
...
>>> transposed
[[1, 5, 9], [2, 6, 10], [3, 7, 11], [4, 8, 12]]
```

zip()

```
>>> list(zip(*matrix))
[(1, 5, 9), (2, 6, 10), (3, 7, 11), (4, 8, 12)]
```

5.2 del

: del pop() del :

```
>>> a = [-1, 1, 66.25, 333, 333, 1234.5]
>>> del a[0]
>>> a
[1, 66.25, 333, 333, 1234.5]
>>> del a[2:4]
>>> a
[1, 66.25, 1234.5]
>>> del a[:]
>>> a
[]
```

del

```
>>> del a
```

a del

5.3

typesseq Python :

```
>>> t = 12345, 54321, 'hello!'
>>> t[0]
12345
>>> t
(12345, 54321, 'hello!')
>>> # Tuples may be nested:
... u = t, (1, 2, 3, 4, 5)
>>> u
((12345, 54321, 'hello!'), (1, 2, 3, 4, 5))
>>> # Tuples are immutable:
... t[0] = 88888
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
>>> # but they can contain mutable objects:
... v = ([1, 2, 3], [3, 2, 1])
>>> v
([1, 2, 3], [3, 2, 1])
```

immutable namedtuples *mutable*

0 1

```
>>> empty = ()
>>> singleton = 'hello',    # <-- note trailing comma
>>> len(empty)
0
>>> len(singleton)
1
>>> singleton
('hello',)
```

t = 12345, 54321, 'hello!' 12345, 54321 'hello!'

```
>>> x, y, z = t
```

5.4

Python

set() set() {}

```
>>> basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}
>>> print(basket)                      # show that duplicates have been removed
{'orange', 'banana', 'pear', 'apple'}
>>> 'orange' in basket                 # fast membership testing
True
>>> 'crabgrass' in basket
False

>>> # Demonstrate set operations on unique letters from two words
...
>>> a = set('abracadabra')
>>> b = set('alacazam')
>>> a                                  # unique letters in a
{'a', 'r', 'b', 'c', 'd'}
>>> a - b                              # letters in a but not in b
{'r', 'd', 'b'}
>>> a | b                              # letters in a or b or both
{'a', 'c', 'r', 'd', 'b', 'm', 'z', 'l'}
>>> a & b                              # letters in both a and b
{'a', 'c'}
>>> a ^ b                              # letters in a or b but not both
{'r', 'd', 'b', 'm', 'z', 'l'}
```

```
>>> a = {x for x in 'abracadabra' if x not in 'abc'}
>>> a
{'r', 'd'}
```

5.5

Python (typesmapping)

append() extend()

: {}

del

list(d) (sorted(d)) in

```
>>> tel = {'jack': 4098, 'sape': 4139}
>>> tel['guido'] = 4127
>>> tel
{'jack': 4098, 'sape': 4139, 'guido': 4127}
>>> tel['jack']
4098
>>> del tel['sape']
>>> tel['irv'] = 4127
>>> tel
{'jack': 4098, 'guido': 4127, 'irv': 4127}
>>> list(tel)
```

( )

( )

```
['jack', 'guido', 'irv']
>>> sorted(tel)
['guido', 'irv', 'jack']
>>> 'guido' in tel
True
>>> 'jack' not in tel
False
```

dict()

```
>>> dict([('sape', 4139), ('guido', 4127), ('jack', 4098)])
{'sape': 4139, 'guido': 4127, 'jack': 4098}
```

```
>>> {x: x**2 for x in (2, 4, 6)}
{2: 4, 4: 16, 6: 36}
```

```
>>> dict(sape=4139, guido=4127, jack=4098)
{'sape': 4139, 'guido': 4127, 'jack': 4098}
```

5.6

items()

```
>>> knights = {'gallahad': 'the pure', 'robin': 'the brave'}
>>> for k, v in knights.items():
...     print(k, v)
...
gallahad the pure
robin the brave
```

enumerate()

```
>>> for i, v in enumerate(['tic', 'tac', 'toe']):
...     print(i, v)
...
0 tic
1 tac
2 toe
```

zip()

```
>>> questions = ['name', 'quest', 'favorite color']
>>> answers = ['lancelot', 'the holy grail', 'blue']
>>> for q, a in zip(questions, answers):
...     print('What is your {0}?  It is {1}.'.format(q, a))
...
What is your name?  It is lancelot.
```

( )

( )

```
What is your quest?  It is the holy grail.
What is your favorite color?  It is blue.
```

reversed()

```
>>> for i in reversed(range(1, 10, 2)):
...     print(i)
...
9
7
5
3
1
```

sorted()

```
>>> basket = ['apple', 'orange', 'apple', 'pear', 'orange', 'banana']
>>> for f in sorted(set(basket)):
...     print(f)
...
apple
banana
orange
pear
```

```
>>> import math
>>> raw_data = [56.2, float('NaN'), 51.7, 55.3, 52.5, float('NaN'), 47.8]
>>> filtered_data = []
>>> for value in raw_data:
...     if not math.isnan(value):
...         filtered_data.append(value)
...
>>> filtered_data
[56.2, 51.7, 55.3, 52.5, 47.8]
```

5.7

while if

in not in is is not

a < b == c a b b c

and or not not or A and not B or C (A and (not B)) or C

and or A C B A and B and C C

```
>>> string1, string2, string3 = '', 'Trondheim', 'Hammer Dance'
>>> non_null = string1 or string2 or string3
>>> non_null
'Trondheim'
```

Python C C C == =

5.8

Unicode

```
(1, 2, 3)              < (1, 2, 4)
[1, 2, 3]              < [1, 2, 4]
'ABC' < 'C' < 'Pascal' < 'Python'
(1, 2, 3, 4)           < (1, 2, 4)
(1, 2)                 < (1, 2, -1)
(1, 2, 3)             == (1.0, 2.0, 3.0)
(1, 2, ('aa', 'ab'))   < (1, 2, ('abc', 'a'), 4)
```

< > 0 0.0 TypeError

CHAPTER 6

Python

Python

Python .py __name__ fibo.py :

```
# Fibonacci numbers module

def fib(n):    # write Fibonacci series up to n
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()

def fib2(n):   # return Fibonacci series up to n
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)
        a, b = b, a+b
    return result
```

Python :

```
>>> import fibo
```

fibo fibo :

```
>>> fibo.fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
>>> fibo.fib2(100)
```

( )

( )

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
>>> fibo.__name__
'fibo'
```

:

```
>>> fib = fibo.fib
>>> fib(500)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

6.1

import [1] ( )

modname.itemname

import

import :

```
>>> from fibo import fib, fib2
>>> fib(500)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

fibo

:

```
>>> from fibo import *
>>> fib(500)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

_ Python

*

as as

```
>>> import fibo as fib
>>> fib.fib(500)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

import fibo fib

It can also be used when utilising from with similar effects:

```
>>> from fibo import fib as fibonacci
>>> fibonacci(500)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

: importlib.reload() import importlib; importlib.reload(modulename)

[1] “ ” “ ”

6.1.1

Python :

```
python fibo.py <arguments>
```

__name__ "__main__" :

```
if __name__ == "__main__":
    import sys
    fib(int(sys.argv[1]))
```

“main” :

```
$ python fibo.py 50
0 1 1 2 3 5 8 13 21 34
```

:

```
>>> import fibo
>>>
```

6.1.2

spam sys.path spam.py sys.path :

-
- PYTHONPATH shell PATH
-

:

Python sys.path

6.1.3 “ ”Python

Python __pycache__ module.*version*.pyc
Python CPython 3.3 spam.py __pycache__/spam.cpython-33.pyc Python

Python

Python

:

- Python -O -OO -O -OO ___doc___ “ ” opt-
- .pyc .py .pyc
- compileall .pyc

- PEP 3147

6.2

Python Python “ ” winreg
Windows sys Python sys.ps1 sys.ps2 :

```
>>> import sys
>>> sys.ps1
'>>> '
>>> sys.ps2
'... '
>>> sys.ps1 = 'C> '
C> print('Yuck!')
Yuck!
C>
```

sys.path PYTHONPATH PYTHONPATH :

```
>>> import sys
>>> sys.path.append('/ufs/guido/lib/python')
```

6.3 dir()

dir() :

```
>>> import fibo, sys
>>> dir(fibo)
['__name__', 'fib', 'fib2']
>>> dir(sys)
['__displayhook__', '__doc__', '__excepthook__', '__loader__', '__name__',
 '__package__', '__stderr__', '__stdin__', '__stdout__',
 '_clear_type_cache', '_current_frames', '_debugmallocstats', '_getframe',
 '_home', '_mercurial', '_xoptions', 'abiflags', 'api_version', 'argv',
 'base_exec_prefix', 'base_prefix', 'builtin_module_names', 'byteorder',
 'call_tracing', 'callstats', 'copyright', 'displayhook',
 'dont_write_bytecode', 'exc_info', 'excepthook', 'exec_prefix',
 'executable', 'exit', 'flags', 'float_info', 'float_repr_style',
 'getcheckinterval', 'getdefaultencoding', 'getdlopenflags',
 'getfilesystemencoding', 'getobjects', 'getprofile', 'getrecursionlimit',
 'getrefcount', 'getsizeof', 'getswitchinterval', 'gettotalrefcount',
 'gettrace', 'hash_info', 'hexversion', 'implementation', 'int_info',
 'intern', 'maxsize', 'maxunicode', 'meta_path', 'modules', 'path',
 'path_hooks', 'path_importer_cache', 'platform', 'prefix', 'ps1',
 'setcheckinterval', 'setdlopenflags', 'setprofile', 'setrecursionlimit',
 'setswitchinterval', 'settrace', 'stderr', 'stdin', 'stdout',
 'thread_info', 'version', 'version_info', 'warnoptions']
```

dir() :

```
>>> a = [1, 2, 3, 4, 5]
>>> import fibo
>>> fib = fibo.fib
>>> dir()
['__builtins__', '__name__', 'a', 'fib', 'fibo', 'sys']
```

dir() builtins :

```
>>> import builtins
>>> dir(builtins)
['ArithmeticError', 'AssertionError', 'AttributeError', 'BaseException',
 'BlockingIOError', 'BrokenPipeError', 'BufferError', 'BytesWarning',
 'ChildProcessError', 'ConnectionAbortedError', 'ConnectionError',
 'ConnectionRefusedError', 'ConnectionResetError', 'DeprecationWarning',
 'EOFError', 'Ellipsis', 'EnvironmentError', 'Exception', 'False',
 'FileExistsError', 'FileNotFoundError', 'FloatingPointError',
 'FutureWarning', 'GeneratorExit', 'IOError', 'ImportError',
 'ImportWarning', 'IndentationError', 'IndexError', 'InterruptedError',
 'IsADirectoryError', 'KeyError', 'KeyboardInterrupt', 'LookupError',
 'MemoryError', 'NameError', 'None', 'NotADirectoryError', 'NotImplemented',
 'NotImplementedError', 'OSError', 'OverflowError',
 'PendingDeprecationWarning', 'PermissionError', 'ProcessLookupError',
 'ReferenceError', 'ResourceWarning', 'RuntimeError', 'RuntimeWarning',
 'StopIteration', 'SyntaxError', 'SyntaxWarning', 'SystemError',
 'SystemExit', 'TabError', 'TimeoutError', 'True', 'TypeError',
 'UnboundLocalError', 'UnicodeDecodeError', 'UnicodeEncodeError',
 'UnicodeError', 'UnicodeTranslateError', 'UnicodeWarning', 'UserWarning',
 'ValueError', 'Warning', 'ZeroDivisionError', '_', '__build_class__',
 '__debug__', '__doc__', '__import__', '__name__', '__package__', 'abs',
 'all', 'any', 'ascii', 'bin', 'bool', 'bytearray', 'bytes', 'callable',
 'chr', 'classmethod', 'compile', 'complex', 'copyright', 'credits',
 'delattr', 'dict', 'dir', 'divmod', 'enumerate', 'eval', 'exec', 'exit',
 'filter', 'float', 'format', 'frozenset', 'getattr', 'globals', 'hasattr',
 'hash', 'help', 'hex', 'id', 'input', 'int', 'isinstance', 'issubclass',
 'iter', 'len', 'license', 'list', 'locals', 'map', 'max', 'memoryview',
 'min', 'next', 'object', 'oct', 'open', 'ord', 'pow', 'print', 'property',
 'quit', 'range', 'repr', 'reversed', 'round', 'set', 'setattr', 'slice',
 'sorted', 'staticmethod', 'str', 'sum', 'super', 'tuple', 'type', 'vars',
 'zip']
```

6.4

“ ” Python A.B A B NumPy Pillow

“ ” .wav .aiff .au

```
sound/                          Top-level package
      __init__.py               Initialize the sound package
```

( )

( )

```
      formats/                Subpackage for file format conversions
              __init__.py
              wavread.py
              wavwrite.py
              aiffread.py
              aiffwrite.py
              auread.py
              auwrite.py
              ...
      effects/                Subpackage for sound effects
              __init__.py
              echo.py
              surround.py
              reverse.py
              ...
      filters/                Subpackage for filters
              __init__.py
              equalizer.py
              vocoder.py
              karaoke.py
              ...
```

Python sys.path

Python __init__.py string __init__.py __all__

:

```
import sound.effects.echo
```

sound.effects.echo

```
sound.effects.echo.echofilter(input, output, delay=0.7, atten=4)
```

```
from sound.effects import echo
```

echo :

```
echo.echofilter(input, output, delay=0.7, atten=4)
```

:

```
from sound.effects.echo import echofilter
```

echo echofilter() :

```
echofilter(input, output, delay=0.7, atten=4)
```

from package import item item import item ImportError

import item.subitem.subsubitem

6.4.1 *

from sound.effects import *

import __init__.py __all__ from package import * * sound/effects/__init__.py :

```
__all__ = ["echo", "surround", "reverse"]
```

from sound.effects import * sound

__all__ from sound.effects import * sound.effects sound.effects __init__.py __init__.py` import :

```
import sound.effects.echo
import sound.effects.surround
from sound.effects import *
```

echo surround from...import sound.effects __all__

import *

from Package import specific_submodule

6.4.2

sound sound.filters.vocoder sound.effects echo from sound.effects import echo

import from module import name surround :

```
from . import echo
from .. import formats
from ..filters import equalizer
```

"__main__" Python

6.4.3

__path__ __init__.py

CHAPTER 7

7.1

print() write() sys.stdout

- f F { } Python

```
>>> year = 2016
>>> event = 'Referendum'
>>> f'Results of the {year} {event}'
'Results of the 2016 Referendum'
```

- str.format() { }

```
>>> yes_votes = 42_572_654
>>> no_votes = 43_132_495
>>> percentage = yes_votes / (yes_votes + no_votes)
>>> '{:-9} YES votes  {:2.2%}'.format(yes_votes, percentage)
' 42572654 YES votes  49.67%'
```

-

repr() or str()

str() repr() SyntaxError str() repr()

:

```
>>> s = 'Hello, world.'
>>> str(s)
'Hello, world.'
>>> repr(s)
"'Hello, world.'"
>>> str(1/7)
'0.14285714285714285'
>>> x = 10 * 3.25
>>> y = 200 * 200
>>> s = 'The value of x is ' + repr(x) + ', and y is ' + repr(y) + '...'
>>> print(s)
The value of x is 32.5, and y is 40000...
>>> # The repr() of a string adds string quotes and backslashes:
... hello = 'hello, world\n'
>>> hellos = repr(hello)
>>> print(hellos)
'hello, world\n'
>>> # The argument to repr() may be any Python object:
... repr((x, y, ('spam', 'eggs')))
"(32.5, 40000, ('spam', 'eggs'))"
```

string Template $x

7.1.1

f- f F {expression} Python

pi :

```
>>> import math
>>> print(f'The value of pi is approximately {math.pi:.3f}.')
The value of pi is approximately 3.142.
```

':' :

```
>>> table = {'Sjoerd': 4127, 'Jack': 4098, 'Dcab': 7678}
>>> for name, phone in table.items():
...     print(f'{name:10} ==> {phone:10d}')
...
Sjoerd     ==>       4127
Jack       ==>       4098
Dcab       ==>       7678
```

'!a' ascii() '!s' str() '!r' repr():

```
>>> animals = 'eels'
>>> print(f'My hovercraft is full of {animals}.')
My hovercraft is full of eels.
>>> print(f'My hovercraft is full of {animals!r}.')
My hovercraft is full of 'eels'.
```

formatspec

7.1.2 format()

str.format() :

```
>>> print('We are the {} who say "{}!"'.format('knights', 'Ni'))
We are the knights who say "Ni!"
```

str.format() str.format()

```
>>> print('{0} and {1}'.format('spam', 'eggs'))
spam and eggs
>>> print('{1} and {0}'.format('spam', 'eggs'))
eggs and spam
```

str.format() :

```
>>> print('This {food} is {adjective}.'.format(
...       food='spam', adjective='absolutely horrible'))
This spam is absolutely horrible.
```

:

```
>>> print('The story of {0}, {1}, and {other}.'.format('Bill', 'Manfred',
                                                       other='Georg'))
The story of Bill, Manfred, and Georg.
```

'[]' :

```
>>> table = {'Sjoerd': 4127, 'Jack': 4098, 'Dcab': 8637678}
>>> print('Jack: {0[Jack]:d}; Sjoerd: {0[Sjoerd]:d}; '
...       'Dcab: {0[Dcab]:d}'.format(table))
Jack: 4098; Sjoerd: 4127; Dcab: 8637678
```

'**' :

```
>>> table = {'Sjoerd': 4127, 'Jack': 4098, 'Dcab': 8637678}
>>> print('Jack: {Jack:d}; Sjoerd: {Sjoerd:d}; Dcab: {Dcab:d}'.format(**table))
Jack: 4098; Sjoerd: 4127; Dcab: 8637678
```

vars()

:

```
>>> for x in range(1, 11):
...     print('{0:2d} {1:3d} {2:4d}'.format(x, x*x, x*x*x))
...
 1   1    1
 2   4    8
 3   9   27
 4  16   64
 5  25  125
 6  36  216
 7  49  343
 8  64  512
 9  81  729
10 100 1000
```

str.format() formatstrings

7.1.3

:

```
>>> for x in range(1, 11):
...     print(repr(x).rjust(2), repr(x*x).rjust(3), end=' ')
...     # Note use of 'end' on previous line
...     print(repr(x*x*x).rjust(4))
...
 1   1    1
 2   4    8
 3   9   27
 4  16   64
 5  25  125
 6  36  216
 7  49  343
 8  64  512
 9  81  729
10 100 1000
```

print()

str.rjust() str.ljust() str.center()
x.ljust(n)[:n]

str.zfill() :

```
>>> '12'.zfill(5)
'00012'
>>> '-3.14'.zfill(7)
'-003.14'
>>> '3.14159265359'.zfill(5)
'3.14159265359'
```

7.1.4

% sprintf() :

```
>>> import math
>>> print('The value of pi is approximately %5.3f.' % math.pi)
The value of pi is approximately 3.142.
```

old-string-formatting

7.2

open() *file object* open(filename, mode)

```
>>> f = open('workfile', 'w')
```

mode 'r' 'w' 'a' 'r+'
mode 'r'

text *mode* (open()) mode 'b' *binary* *mode*

(Unix \n, Windows \r\n) \n \n JPEG EXE

with with try-finally :

```
>>> with open('workfile') as f:
...     read_data = f.read()
>>> f.closed
True
```

with f.close() Python Python

with f.close() :

```
>>> f.close()
>>> f.read()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: I/O operation on closed file.
```

7.2.1

f

f.read(size) *size* *size*
size f.read() ('')

```
>>> f.read()
'This is the entire file.\n'
>>> f.read()
''
```

f.readline() \n f.readline() '\n' :

```
>>> f.readline()
'This is the first line of the file.\n'
>>> f.readline()
'Second line of the file\n'
>>> f.readline()
''
```

:

```
>>> for line in f:
...     print(line, end='')
...
This is the first line of the file.
Second line of the file
```

list(f) f.readlines()

f.write(string) *string* :

```
>>> f.write('This is a test\n')
15
```

:

```
>>> value = ('the answer', 42)
>>> s = str(value)  # convert the tuple to string
>>> f.write(s)
18
```

f.tell()

f.seek(offset, from_what) *offset* *from_what* *from_what** *0* *1* *2* *from_what* 0 :

```
>>> f = open('workfile', 'rb+')
>>> f.write(b'0123456789abcdef')
16
>>> f.seek(5)      # Go to the 6th byte in the file
5
>>> f.read(1)
b'5'
>>> f.seek(-3, 2)  # Go to the 3rd byte before the end
13
>>> f.read(1)
b'd'
```

b seek(0, 2) *offset* f.tell() *offset*

isatty() truncate()

7.2.2 json

read() int() '123' 123

Python JSON (JavaScript Object Notation) *serializing* *deserializing* json Python

: JSON

x JSON :

```
>>> import json
>>> json.dumps([1, 'simple', 'list'])
'[1, "simple", "list"]'
```

dumps() dump() *text file* f *text file* :

```
json.dump(x, f)
```

f *text file* :

```
x = json.load(f)
```

JSON json

:

pickle -

JSON *pickle* Python Python pickle

CHAPTER 8

8.1

Python :

```
>>> while True print('Hello world')
  File "<stdin>", line 1
    while True print('Hello world')
                    ^
SyntaxError: invalid syntax
```

" " token print() (':')

8.2

* * Python :

```
>>> 10 * (1/0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
>>> 4 + spam*3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'spam' is not defined
>>> '2' + 2
```

( )

( )

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Can't convert 'int' object to str implicitly
```

ZeroDivisionError NameError
TypeError

bltin-exceptions

8.3

Control-C KeyboardInterrupt :

```
>>> while True:
...     try:
...         x = int(input("Please enter a number: "))
...         break
...     except ValueError:
...         print("Oops!  That was no valid number.  Try again...")
...
```

try

- *try* try except
- *except* try
- try except except try
- except try

try except try try except :

```
... except (RuntimeError, TypeError, NameError):
...     pass
```

except except — except B, C, D

```
class B(Exception):
    pass

class C(B):
    pass

class D(C):
    pass

for cls in [B, C, D]:
    try:
        raise cls()
    except D:
```

( )

( )

```
        print("D")
    except C:
        print("C")
    except B:
        print("B")
```

except except B B B B — except

except :

```
import sys

try:
    f = open('myfile.txt')
    s = f.readline()
    i = int(s.strip())
except OSError as err:
    print("OS error: {0}".format(err))
except ValueError:
    print("Could not convert data to an integer.")
except:
    print("Unexpected error:", sys.exc_info()[0])
    raise
```

try ... except *else* except try :

```
for arg in sys.argv[1:]:
    try:
        f = open(arg, 'r')
    except OSError:
        print('cannot open', arg)
    else:
        print(arg, 'has', len(f.readlines()), 'lines')
        f.close()
```

else try try ... except

except instance.args __str__() .args
:

```
>>> try:
...     raise Exception('spam', 'eggs')
... except Exception as inst:
...     print(type(inst))    # the exception instance
...     print(inst.args)     # arguments stored in .args
...     print(inst)          # __str__ allows args to be printed directly,
...                          # but may be overridden in exception subclasses
...     x, y = inst.args     # unpack args
...     print('x =', x)
...     print('y =', y)
...
<class 'Exception'>
```

( )

( )

```
('spam', 'eggs')
('spam', 'eggs')
x = spam
y = eggs
```

, ,

try try :

```
>>> def this_fails():
...     x = 1/0
...
>>> try:
...     this_fails()
... except ZeroDivisionError as err:
...     print('Handling run-time error:', err)
...
Handling run-time error: division by zero
```

8.4

raise :

```
>>> raise NameError('HiThere')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: HiThere
```

raise Exception :

```
raise ValueError  # shorthand for 'raise ValueError()'
```

raise

```
>>> try:
...     raise NameError('HiThere')
... except NameError:
...     print('An exception flew by!')
...     raise
...
An exception flew by!
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
NameError: HiThere
```

8.5

Python Exception

:

```
class Error(Exception):
    """Base class for exceptions in this module."""
    pass

class InputError(Error):
    """Exception raised for errors in the input.

    Attributes:
        expression -- input expression in which the error occurred
        message -- explanation of the error
    """

    def __init__(self, expression, message):
        self.expression = expression
        self.message = message

class TransitionError(Error):
    """Raised when an operation attempts a state transition that's not
    allowed.

    Attributes:
        previous -- state at beginning of transition
        next -- attempted new state
        message -- explanation of why the specific transition is not allowed
    """

    def __init__(self, previous, next, message):
        self.previous = previous
        self.next = next
        self.message = message
```

"Error"

8.6

try :

```
>>> try:
...     raise KeyboardInterrupt
... finally:
...     print('Goodbye, world!')
...
Goodbye, world!
KeyboardInterrupt
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
```

finally try try except except else finally try break, continue return finally " " :

```
>>> def divide(x, y):
...     try:
...         result = x / y
...     except ZeroDivisionError:
...         print("division by zero!")
...     else:
...         print("result is", result)
...     finally:
...         print("executing finally clause")
...
>>> divide(2, 1)
result is 2.0
executing finally clause
>>> divide(2, 0)
division by zero!
executing finally clause
>>> divide("2", "1")
executing finally clause
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 3, in divide
TypeError: unsupported operand type(s) for /: 'str' and 'str'
```

finally TypeError except finally

finally

8.7

:

```
for line in open("myfile.txt"):
    print(line, end="")
```

with :

```
with open("myfile.txt") as f:
    for line in f:
        print(line, end="")
```

f

CHAPTER 9

Python C++ Modula-3 Python
Python

C++ *public* *virtual* Modula-3 Smalltalk C++
3 C++

(Lacking universally accepted terminology to talk about classes, I will make occasional use of Smalltalk and C++ terms. I would use Modula-3 terms, since its object-oriented semantics are closer to those of Python than C++, but I expect that few readers have heard of it.)

9.1

Python Python
— Pascal

9.2 Python

Python Python

namespace Python `abs()`
“ ” _

— `z.real` `real` `z` `modname.funcname` `modname` `funcname`
[1]

`modname.the_answer = 42` `del` `del modname.the_answer` `modname` `the_answer`

[1] `__dict__` `__dict__`

Python __main__
builtins

Python “ ”

Although scopes are determined statically, they are used dynamically. At any time during execution, there are at least three nested scopes whose namespaces are directly accessible:

-
-
-
-

nonlocal

—

Python – global – — del x x
import

global nonlocal

9.2.1

This is an example demonstrating how to reference the different scopes and namespaces, and how `global` and `nonlocal` affect variable binding:

```
def scope_test():
    def do_local():
        spam = "local spam"

    def do_nonlocal():
        nonlocal spam
        spam = "nonlocal spam"

    def do_global():
        global spam
        spam = "global spam"

    spam = "test spam"
    do_local()
    print("After local assignment:", spam)
    do_nonlocal()
    print("After nonlocal assignment:", spam)
    do_global()
    print("After global assignment:", spam)

scope_test()
print("In global scope:", spam)
```

```
After local assignment: test spam
After nonlocal assignment: nonlocal spam
After global assignment: nonlocal spam
In global scope: global spam
```

scope_test *spam* nonlocal *scope_test* *spam* global

global *spam*

9.3

9.3.1

:

```
class ClassName:
    <statement-1>
    .
    .
    .
    <statement-N>
```

(def) if

— —

—

(ClassName)

9.3.2

Python : obj.name :

```
class MyClass:
    """A simple example class"""
    i = 12345

    def f(self):
        return 'hello world'
```

MyClass.i MyClass.f MyClass.i __doc__ : "A simple example class"

:

```
x = MyClass()
```

x

“ ” __init__() :

```
def __init__(self):
    self.data = []
```

__init__() __init__() :

```
x = MyClass()
```

__init__() __init__() :

```
>>> class Complex:
...     def __init__(self, realpart, imagpart):
...         self.r = realpart
...         self.i = imagpart
...
>>> x = Complex(3.0, -4.5)
>>> x.r, x.i
(3.0, -4.5)
```

9.3.3

Smalltalk “ ” C++ “ ” x MyClass 16 :

```
x.counter = 1
while x.counter < 10:
    x.counter = x.counter * 2
print(x.counter)
del x.counter
```

“ ” Python append, insert, remove, sort

x.f MyClass.f x.i MyClass.i x.f MyClass.f —

9.3.4

:

```
x.f()
```

MyClass 'hello world' : x.f :

```
xf = x.f
while True:
    print(xf())
```

hello world

x.f() f() Python — ...

x.f() MyClass.f(x) n

9.3.5

:

```
class Dog:

    kind = 'canine'         # class variable shared by all instances

    def __init__(self, name):
        self.name = name    # instance variable unique to each instance

>>> d = Dog('Fido')
>>> e = Dog('Buddy')
>>> d.kind                  # shared by all dogs
'canine'
>>> e.kind                  # shared by all dogs
'canine'
>>> d.name                  # unique to d
'Fido'
>>> e.name                  # unique to e
'Buddy'
```

mutable *tricks* *Dog* :

```
class Dog:

    tricks = []             # mistaken use of a class variable

    def __init__(self, name):
        self.name = name

    def add_trick(self, trick):
        self.tricks.append(trick)

>>> d = Dog('Fido')
>>> e = Dog('Buddy')
>>> d.add_trick('roll over')
>>> e.add_trick('play dead')
>>> d.tricks                # unexpectedly shared by all dogs
['roll over', 'play dead']
```

:

```
class Dog:

    def __init__(self, name):
        self.name = name
        self.tricks = []    # creates a new empty list for each dog
```

( )

( )

```
    def add_trick(self, trick):
        self.tricks.append(trick)

>>> d = Dog('Fido')
>>> e = Dog('Buddy')
>>> d.add_trick('roll over')
>>> e.add_trick('play dead')
>>> d.tricks
['roll over']
>>> e.tricks
['play dead']
```

9.4

“ ” Python — C Python
C Python

— —

self : self Python Python

:

```
# Function defined outside the class
def f1(self, x, y):
    return min(x, x+y)

class C:
    f = f1

    def g(self):
        return 'hello world'

    h = g
```

f, g h C C — h g

self :

```
class Bag:
    def __init__(self):
        self.data = []

    def add(self, x):
        self.data.append(x)

    def addtwice(self, x):
        self.add(x)
        self.add(x)
```

object.__class__

9.5

" " :

```
class DerivedClassName(BaseClassName):
    <statement-1>
    .
    .
    .
    <statement-N>
```

BaseClassName :

```
class DerivedClassName(modname.BaseClassName):
```

: DerivedClassName()

C++ Python virtual

BaseClassName.methodname(self, arguments)

BaseClassName

Python

- isinstance() : isinstance(obj, int) obj.__class__ int int True
- issubclass() : issubclass(bool, int) True bool int issubclass(float, int) False float int

9.5.1

Python supports a form of multiple inheritance as well. A class definition with multiple base classes looks like this:

```
class DerivedClassName(Base1, Base2, Base3):
    <statement-1>
    .
    .
    .
    <statement-N>
```

DerivedClassName Base1 Base1 Base2

super() super

object object https://www.python.org/download/releases/2.3/mro/

9.6

“ ” Python Python (_spam) API ( )
__spam _classname__spam classname

:

```
class Mapping:
    def __init__(self, iterable):
        self.items_list = []
        self.__update(iterable)

    def update(self, iterable):
        for item in iterable:
            self.items_list.append(item)

    __update = update   # private copy of original update() method

class MappingSubclass(Mapping):

    def update(self, keys, values):
        # provides new signature for update()
        # but does not break __init__()
        for item in zip(keys, values):
            self.items_list.append(item)
```

MappingSubclass __update Mapping _Mapping__update MappingSubclass
_MappingSubclass__update

exec() eval() global getattr(), setattr()
delattr() __dict__

9.7

Pascal “record” C “struct” :

```
class Employee:
    pass

john = Employee()  # Create an empty employee record

# Fill the fields of the record
john.name = 'John Doe'
john.dept = 'computer lab'
john.salary = 1000
```

Python read() readline()

: m.__self__ m() m.__func__

9.8

for :

```
for element in [1, 2, 3]:
    print(element)
for element in (1, 2, 3):
    print(element)
for key in {'one':1, 'two':2}:
    print(key)
for char in "123":
    print(char)
for line in open("myfile.txt"):
    print(line, end='')
```

Python for iter() __next__()
__next__() StopIteration for next() __next__() :

```
>>> s = 'abc'
>>> it = iter(s)
>>> it
<iterator object at 0x00A1DB50>
>>> next(it)
'a'
>>> next(it)
'b'
>>> next(it)
'c'
>>> next(it)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    next(it)
StopIteration
```

__iter__() __next__() __next__() __iter__() self:

```
class Reverse:
    """Iterator for looping over a sequence backwards."""
    def __init__(self, data):
        self.data = data
        self.index = len(data)

    def __iter__(self):
        return self

    def __next__(self):
        if self.index == 0:
            raise StopIteration
        self.index = self.index - 1
        return self.data[self.index]
```

```
>>> rev = Reverse('spam')
>>> iter(rev)
```

( )

( )

```
<__main__.Reverse object at 0x00A1DB50>
>>> for char in rev:
...     print(char)
...
m
a
p
s
```

9.9

Generator yield next() :

```
def reverse(data):
    for index in range(len(data)-1, -1, -1):
        yield data[index]
```

```
>>> for char in reverse('golf'):
...     print(char)
...
f
l
o
g
```

__iter__() __next__()

self.index self.data

StopIteration

9.10

:

```
>>> sum(i*i for i in range(10))                 # sum of squares
285

>>> xvec = [10, 20, 30]
>>> yvec = [7, 5, 3]
>>> sum(x*y for x,y in zip(xvec, yvec))         # dot product
260

>>> from math import pi, sin
>>> sine_table = {x: sin(x*pi/180) for x in range(0, 91)}

>>> unique_words = set(word  for line in page  for word in line.split())
```

( )

( )

```
>>> valedictorian = max((student.gpa, student.name) for student in graduates)

>>> data = 'golf'
>>> list(data[i] for i in range(len(data)-1, -1, -1))
['f', 'l', 'o', 'g']
```

CHAPTER 10

10.1

os :

```
>>> import os
>>> os.getcwd()      # Return the current working directory
'C:\\Python37'
>>> os.chdir('/server/accesslogs')   # Change current working directory
>>> os.system('mkdir today')   # Run the command mkdir in the system shell
0
```

import os from os import * open() os.open()

dir() help() os:

```
>>> import os
>>> dir(os)
<returns a list of all module functions>
>>> help(os)
<returns an extensive manual page created from the module's docstrings>
```

shutil :

```
>>> import shutil
>>> shutil.copyfile('data.db', 'archive.db')
'archive.db'
>>> shutil.move('/build/executables', 'installdir')
'installdir'
```

10.2

glob :

```
>>> import glob
>>> glob.glob('*.py')
['primes.py', 'random.py', 'quote.py']
```

10.3

sys *argv* python demo.py one two three

```
>>> import sys
>>> print(sys.argv)
['demo.py', 'one', 'two', 'three']
```

getopt Unix getopt() *sys.argv* argparse

10.4

sys *stdin* *stdout* *stderr* *stdout* :

```
>>> sys.stderr.write('Warning, log file not found starting a new one\n')
Warning, log file not found starting a new one
```

sys.exit()

10.5

re :

```
>>> import re
>>> re.findall(r'\bf[a-z]*', 'which foot or hand fell fastest')
['foot', 'fell', 'fastest']
>>> re.sub(r'(\b[a-z]+) \1', r'\1', 'cat in the the hat')
'cat in the hat'
```

:

```
>>> 'tea for too'.replace('too', 'two')
'tea for two'
```

10.6

math C :

```
>>> import math
>>> math.cos(math.pi / 4)
0.70710678118654757
>>> math.log(1024, 2)
10.0
```

random :

```
>>> import random
>>> random.choice(['apple', 'pear', 'banana'])
'apple'
>>> random.sample(range(100), 10)   # sampling without replacement
[30, 83, 16, 4, 8, 81, 41, 50, 18, 33]
>>> random.random()    # random float
0.17970987693706186
>>> random.randrange(6)    # random integer chosen from range(6)
4
```

statistics :

```
>>> import statistics
>>> data = [2.75, 1.75, 1.25, 0.25, 0.5, 1.25, 3.5]
>>> statistics.mean(data)
1.6071428571428572
>>> statistics.median(data)
1.25
>>> statistics.variance(data)
1.3720238095238095
```

SciPy <https://scipy.org>

10.7

urllib.request URL smtplib :

```
>>> from urllib.request import urlopen
>>> with urlopen('http://tycho.usno.navy.mil/cgi-bin/timer.pl') as response:
...     for line in response:
...         line = line.decode('utf-8')  # Decoding the binary data to text.
...         if 'EST' in line or 'EDT' in line:  # look for Eastern Time
...             print(line)

<BR>Nov. 25, 09:43:32 PM EST

>>> import smtplib
>>> server = smtplib.SMTP('localhost')
>>> server.sendmail('soothsayer@example.org', 'jcaesar@example.org',
... """To: jcaesar@example.org
... From: soothsayer@example.org
...
... Beware the Ides of March.
```

( )

( )

```
... """)
>>> server.quit()
```

localhost

10.8

datetime

```
>>> # dates are easily constructed and formatted
>>> from datetime import date
>>> now = date.today()
>>> now
datetime.date(2003, 12, 2)
>>> now.strftime("%m-%d-%y. %d %b %Y is a %A on the %d day of %B.")
'12-02-03. 02 Dec 2003 is a Tuesday on the 02 day of December.'

>>> # dates support calendar arithmetic
>>> birthday = date(1964, 7, 31)
>>> age = now - birthday
>>> age.days
14368
```

10.9

zlib, gzip, bz2, lzma, zipfile tarfile :

```
>>> import zlib
>>> s = b'witch which has which witches wrist watch'
>>> len(s)
41
>>> t = zlib.compress(s)
>>> len(t)
37
>>> zlib.decompress(t)
b'witch which has which witches wrist watch'
>>> zlib.crc32(s)
226805979
```

10.10

Python Python

timeit :

```
>>> from timeit import Timer
>>> Timer('t=a; a=b; b=t', 'a=1; b=2').timeit()
```

( )

( )

```
0.57535828626024577
>>> Timer('a,b = b,a', 'a=1; b=2').timeit()
0.54962537085770791
```

timeit profile pstats

10.11

doctest doctest :

```
def average(values):
    """Computes the arithmetic mean of a list of numbers.

    >>> print(average([20, 30, 70]))
    40.0
    """
    return sum(values) / len(values)

import doctest
doctest.testmod()   # automatically validate the embedded tests
```

unittest doctest :

```
import unittest

class TestStatisticalFunctions(unittest.TestCase):

    def test_average(self):
        self.assertEqual(average([20, 30, 70]), 40.0)
        self.assertEqual(round(average([1, 5, 7]), 1), 4.3)
        with self.assertRaises(ZeroDivisionError):
            average([])
        with self.assertRaises(TypeError):
            average(20, 30, 70)

unittest.main()  # Calling from the command line invokes all tests
```

10.12

Python “ ” :

- xmlrpc.client xmlrpc.server XML
- email MIME **RFC 2822** smtplib poplib
- json csv XML xml.etree.ElementTree xml.dom xml.sax Python
- sqlite3 SQLite SQL

- gettext locale codecs

CHAPTER 11

11.1

reprlib repr() :

```
>>> import reprlib
>>> reprlib.repr(set('supercalifragilisticexpialidocious'))
"{'a', 'c', 'd', 'e', 'f', 'g', ...}"
```

pprint “ ” :

```
>>> import pprint
>>> t = [[[['black', 'cyan'], 'white', ['green', 'red']], [['magenta',
...     'yellow'], 'blue']]]
...
>>> pprint.pprint(t, width=30)
[[[['black', 'cyan'],
   'white',
   ['green', 'red']],
  [['magenta', 'yellow'],
   'blue']]]
```

textwrap :

```
>>> import textwrap
>>> doc = """The wrap() method is just like fill() except that it returns
... a list of strings instead of one big string with newlines to separate
... the wrapped lines."""
...
```

( )

( )

```
>>> print(textwrap.fill(doc, width=40))
The wrap() method is just like fill()
except that it returns a list of strings
instead of one big string with newlines
to separate the wrapped lines.
```

locale locale format grouping :

```
>>> import locale
>>> locale.setlocale(locale.LC_ALL, 'English_United States.1252')
'English_United States.1252'
>>> conv = locale.localeconv()          # get a mapping of conventions
>>> x = 1234567.8
>>> locale.format("%d", x, grouping=True)
'1,234,567'
>>> locale.format_string("%s%.*f", (conv['currency_symbol'],
...                      conv['frac_digits'], x), grouping=True)
'$1,234,567.80'
```

11.2

string Template

$ Python $$ $:

```
>>> from string import Template
>>> t = Template('${village}folk send $$10 to $cause.')
>>> t.substitute(village='Nottingham', cause='the ditch fund')
'Nottinghamfolk send $10 to the ditch fund.'
```

substitute() KeyError safe_substitute() ——

```
>>> t = Template('Return the $item to $owner.')
>>> d = dict(item='unladen swallow')
>>> t.substitute(d)
Traceback (most recent call last):
  ...
KeyError: 'owner'
>>> t.safe_substitute(d)
'Return the unladen swallow to $owner.'
```

Template :

```
>>> import time, os.path
>>> photofiles = ['img_1074.jpg', 'img_1076.jpg', 'img_1077.jpg']
>>> class BatchRename(Template):
...     delimiter = '%'
>>> fmt = input('Enter rename style (%d-date %n-seqnum %f-format):  ')
Enter rename style (%d-date %n-seqnum %f-format):  Ashley_%n%f
```

( )

( )

```
>>> t = BatchRename(fmt)
>>> date = time.strftime('%d%b%y')
>>> for i, filename in enumerate(photofiles):
...     base, ext = os.path.splitext(filename)
...     newname = t.substitute(d=date, n=i, f=ext)
...     print('{0} --> {1}'.format(filename, newname))

img_1074.jpg --> Ashley_0.jpg
img_1076.jpg --> Ashley_1.jpg
img_1077.jpg --> Ashley_2.jpg
```

XML HTML

11.3

struct pack() unpack() zipfile ZIP Pack "H" "I"
"<" :

```
import struct

with open('myfile.zip', 'rb') as f:
    data = f.read()

start = 0
for i in range(3):                      # show the first 3 file headers
    start += 14
    fields = struct.unpack('<IIIHH', data[start:start+16])
    crc32, comp_size, uncomp_size, filenamesize, extra_size = fields

    start += 16
    filename = data[start:start+filenamesize]
    start += filenamesize
    extra = data[start:start+extra_size]
    print(filename, hex(crc32), comp_size, uncomp_size)

    start += extra_size + comp_size     # skip to the next header
```

11.4

I/O

threading :

```
import threading, zipfile

class AsyncZip(threading.Thread):
    def __init__(self, infile, outfile):
        threading.Thread.__init__(self)
        self.infile = infile
```

( )

( )

```
        self.outfile = outfile

    def run(self):
        f = zipfile.ZipFile(self.outfile, 'w', zipfile.ZIP_DEFLATED)
        f.write(self.infile)
        f.close()
        print('Finished background zip of:', self.infile)

background = AsyncZip('mydata.txt', 'myarchive.zip')
background.start()
print('The main program continues to run in foreground.')

background.join()    # Wait for the background task to finish
print('Main program waited until background was done.')
```

threading

queue Queue

11.5

logging sys.stderr

```
import logging
logging.debug('Debugging information')
logging.info('Informational message')
logging.warning('Warning:config file %s not found', 'server.conf')
logging.error('Error occurred')
logging.critical('Critical error -- shutting down')
```

:

```
WARNING:root:Warning:config file server.conf not found
ERROR:root:Error occurred
CRITICAL:root:Critical error -- shutting down
```

informational debugging HTTP DEBUG INFO WARNING ERROR CRITICAL

Python

11.6

Python *garbage collection*

weakref :

```
>>> import weakref, gc
>>> class A:
...     def __init__(self, value):
...         self.value = value
```

( )

( )

```
...     def __repr__(self):
...         return str(self.value)
...
>>> a = A(10)                   # create a reference
>>> d = weakref.WeakValueDictionary()
>>> d['primary'] = a            # does not create a reference
>>> d['primary']                # fetch the object if it is still alive
10
>>> del a                       # remove the one reference
>>> gc.collect()                # run garbage collection right away
0
>>> d['primary']                # entry was automatically removed
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    d['primary']                # entry was automatically removed
  File "C:/python37/lib/weakref.py", line 46, in __getitem__
    o = self.data[key]()
KeyError: 'primary'
```

11.7

array array() ("H") Python int 16 :

```
>>> from array import array
>>> a = array('H', [4000, 10, 700, 22222])
>>> sum(a)
26932
>>> a[1:3]
array('H', [10, 700])
```

collections deque() :

```
>>> from collections import deque
>>> d = deque(["task1", "task2", "task3"])
>>> d.append("task4")
>>> print("Handling", d.popleft())
Handling task1
```

```
unsearched = deque([starting_node])
def breadth_first_search(unsearched):
    node = unsearched.popleft()
    for m in gen_moves(node):
        if is_goal(m):
            return m
        unsearched.append(m)
```

bisect :

```
>>> import bisect
>>> scores = [(100, 'perl'), (200, 'tcl'), (400, 'lua'), (500, 'python')]
>>> bisect.insort(scores, (300, 'ruby'))
>>> scores
[(100, 'perl'), (200, 'tcl'), (300, 'ruby'), (400, 'lua'), (500, 'python')]
```

heapq :

```
>>> from heapq import heapify, heappop, heappush
>>> data = [1, 3, 5, 7, 9, 2, 4, 6, 8, 0]
>>> heapify(data)                      # rearrange the list into heap order
>>> heappush(data, -5)                 # add a new entry
>>> [heappop(data) for i in range(3)]  # fetch the three smallest entries
[-5, 0, 1]
```

11.8

decimal Decimal float

-
-
-
-
-

70 5 :

```
>>> from decimal import *
>>> round(Decimal('0.70') * Decimal('1.05'), 2)
Decimal('0.74')
>>> round(.70 * 1.05, 2)
0.73
```

Decimal Decimal

Decimal :

```
>>> Decimal('1.00') % Decimal('.10')
Decimal('0.00')
>>> 1.00 % 0.10
0.09999999999999995

>>> sum([Decimal('0.1')]*10) == Decimal('1.0')
True
>>> sum([0.1]*10) == 1.0
False
```

decimal :

```
>>> getcontext().prec = 36
>>> Decimal(1) / Decimal(7)
Decimal('0.142857142857142857142857142857142857')
```

CHAPTER 12

12.1

Python

Python A 1.0 B 2.0 1.0 2.0

virtual environment Python

A 1.0 B 2.0 B 3.0 A

12.2

venv venv Python Python python3 Python

venv :

```
python3 -m venv tutorial-env
```

tutorial-env Python

Windows :

```
tutorial-env\Scripts\activate.bat
```

Unix MacOS :

```
source tutorial-env/bin/activate
```

bash shell **csh** **fish** shell activate.csh activate.fish

Activating the virtual environment will change your shell's prompt to show what virtual environment you're using, and modify the environment so that running `python` will get you that particular version and installation of Python. For example:

```
$ source ~/envs/tutorial-env/bin/activate
(tutorial-env) $ python
Python 3.5.1 (default, May  6 2016, 10:59:36)
  ...
>>> import sys
>>> sys.path
['', '/usr/local/lib/python35.zip', ...,
'~/envs/tutorial-env/lib/python3.5/site-packages']
>>>
```

12.3 pip

`pip` `pip` Python Package Index <https://pypi.org> Python Package Index
`pip` :

```
(tutorial-env) $ pip search astronomy
skyfield               - Elegant astronomy for Python
gary                   - Galactic astronomy and gravitational dynamics.
novas                  - The United States Naval Observatory NOVAS astronomy library
astroobs               - Provides astronomy ephemeris to plan telescope observations
PyAstronomy            - A collection of astronomy related tools for Python.
...
```

`pip` "search" "install" "uninstall" "freeze" installing-index `pip`

```
(tutorial-env) $ pip install novas
Collecting novas
  Downloading novas-3.1.1.3.tar.gz (136kB)
Installing collected packages: novas
  Running setup.py install for novas
Successfully installed novas-3.1.1.3
```

`==`

```
(tutorial-env) $ pip install requests==2.6.0
Collecting requests==2.6.0
  Using cached requests-2.6.0-py2.py3-none-any.whl
Installing collected packages: requests
Successfully installed requests-2.6.0
```

`pip` `pip install --upgrade`

```
(tutorial-env) $ pip install --upgrade requests
Collecting requests
Installing collected packages: requests
  Found existing installation: requests 2.6.0
```

( )

( )

```
    Uninstalling requests-2.6.0:
      Successfully uninstalled requests-2.6.0
Successfully installed requests-2.7.0
```

pip uninstall

pip show

```
(tutorial-env) $ pip show requests
---
Metadata-Version: 2.0
Name: requests
Version: 2.7.0
Summary: Python HTTP for Humans.
Home-page: http://python-requests.org
Author: Kenneth Reitz
Author-email: me@kennethreitz.com
License: Apache 2.0
Location: /Users/akuchling/envs/tutorial-env/lib/python3.4/site-packages
Requires:
```

pip list

```
(tutorial-env) $ pip list
novas (3.1.1.3)
numpy (1.9.2)
pip (7.0.3)
requests (2.7.0)
setuptools (16.0)
```

pip freeze‘ pip install requirements.txt

```
(tutorial-env) $ pip freeze > requirements.txt
(tutorial-env) $ cat requirements.txt
novas==3.1.1.3
numpy==1.9.2
requests==2.7.0
```

requirements.txt install -r

```
(tutorial-env) $ pip install -r requirements.txt
Collecting novas==3.1.1.3 (from -r requirements.txt (line 1))
  ...
Collecting numpy==1.9.2 (from -r requirements.txt (line 2))
  ...
Collecting requests==2.7.0 (from -r requirements.txt (line 3))
  ...
Installing collected packages: novas, numpy, requests
  Running setup.py install for novas
Successfully installed novas-3.1.1.3 numpy-1.9.2 requests-2.7.0
```

pip pip installing-index Python distributing-index

CHAPTER 13

Python - Python

Python

- library-index:
  Python Unix HTTP CGI
- installing-index Python
- reference-index: Python

Python

- https://www.python.org Python Web Python
- https://docs.python.org Python
- https://pypi.org Python Package Index Cheese Shop Python
- https://code.activestate.com/recipes/langs/python/ Python Cookbook Python Cookbook O'Reilly Associates ISBN 0-596-00797-3
- http://www.pyvideo.org Python
- https://scipy.org Ecientific Python

Python *comp.lang.python* python-list@python.org
https://mail.python.org/pipermail/

FAQ

CHAPTER 14

Python Korn shell GNU Bash shell GNU Readline

14.1 Tab

Tab Python string.a '.'
__getattr__() .python_history Python

14.2

Python

IPython tab bpython

CHAPTER 15

2

```
0.125
```

1/10 + 2/100 + 5/1000

```
0.001
```

0/2 + 0/4 + 1/8 10 2

1/3

```
0.3
```

:

```
0.33
```

:

```
0.333
```

1/3 1/3

2 0.1 2 2 1/10

```
0.0001100110011001100110011001100110011001100110011...
```

8 53 2 1/10 `3602879701896397 / 2 **
55` 1/10 1/10

Python Python 0.1

```
>>> 0.1
0.1000000000000000055511151231257827021181583404541015625
```

Python

```
>>> 1 / 10
0.1
```

1/10 1/10

0.1 0.10000000000000001 0.1000000000000000055511151231257827021181583404541015625
3602879701896397 / 2 ** 55 eval(repr(x)) == x

Python repr() 17 0.10000000000000001 Python 3.1 Python
0.1

Python

:

```
>>> format(math.pi, '.12g')  # give 12 significant digits
'3.14159265359'

>>> format(math.pi, '.2f')   # give 2 digits after the point
'3.14'

>>> repr(math.pi)
'3.141592653589793'
```

0.1 1/10 0.1 0.3:

```
>>> .1 + .1 + .1 == .3
False
```

0.1 1/10 0.3 3/10 round() :

```
>>> round(.1, 1) + round(.1, 1) + round(.1, 1) == round(.3, 1)
False
```

round() “ ” :

```
>>> round(.1 + .1 + .1, 10) == round(.3, 10)
True
```

“ ” ”0.1” “ ”

“ ” Python 2**53 1

str() formatstrings str.format()

decimal

fractions 1/3

Python NumPy SciPy <https://scipy.org>

Python float.as_integer_ratio() :

```
>>> x = 3.14159
>>> x.as_integer_ratio()
(3537115888337719, 1125899906842624)
```

:

```
>>> x == 3537115888337719 / 1125899906842624
True
```

`float.hex()` 16 :

```
>>> x.hex()
'0x1.921f9f01b866ep+1'
```

:

```
>>> x == float.fromhex('0x1.921f9f01b866ep+1')
True
```

Python Java C99 .

`math.fsum()` “ ” :

```
>>> sum([0.1] * 10) == 1.0
False
>>> math.fsum([0.1] * 10) == 1.0
True
```

15.1

”0.1”

2 Python Perl C C++ Java Fortran

1/10 2000 11 IEEE-754 Python IEEE-754 “ ” 754 53
0.1 $J/2^{**}N$ J 53

```
1 / 10 ~= J / (2**N)
```

```
J ~= 2**N / 10
```

J 53 (`>= 2**52 < 2**53`) N 56:

```
>>> 2**52 <=  2**56 // 10  < 2**53
True
```

56 N J 53 J :

```
>>> q, r = divmod(2**56, 10)
>>> r
6
```

10 :

```
>>> q+1
7205759403792794
```

754 1/10 :

```
7205759403792794 / 2 ** 56
```

:

```
3602879701896397 / 2 ** 55
```

1/10 1/10 1/10

“ ”1/10 754 :

```
>>> 0.1 * 2 ** 55
3602879701896397.0
```

10**55 55 :

```
>>> 3602879701896397 * 10 ** 55 // 2 ** 55
1000000000000000055511151231257827021181583404541015625
```

0.1000000000000000055511151231257827021181583404541015625
Python 17 :

```
>>> format(0.1, '.17f')
'0.10000000000000001'
```

fractions decimal :

```
>>> from decimal import Decimal
>>> from fractions import Fraction

>>> Fraction.from_float(0.1)
Fraction(3602879701896397, 36028797018963968)

>>> (0.1).as_integer_ratio()
(3602879701896397, 36028797018963968)

>>> Decimal.from_float(0.1)
Decimal('0.1000000000000000055511151231257827021181583404541015625')

>>> format(Decimal.from_float(0.1), '.17')
'0.10000000000000001'
```

CHAPTER 16

16.1

16.1.1

try except

Control-C Delete [1] KeyboardInterrupt try

16.1.2 Python

BSD Unix Python shell :

```
#!/usr/bin/env python3.5
```

PATH #! Unix '\n' Windows '\r\n' '#' Python

chmod

```
$ chmod +x myscript.py
```

Windows " " Python .py python.exe Python .pyw

16.1.3

Python PYTHONSTARTUP Unix shell .profile

This file is only read in interactive sessions, not when Python reads commands from a script, and not when /dev/tty is given as the explicit source of commands (which otherwise behaves like an interactive session). It is executed in the same namespace where interactive commands are executed, so that objects that it defines

[1] GNU Readline

or imports can be used without qualification in the interactive session. You can also change the prompts `sys.ps1` and `sys.ps2` in this file.

`if os.path.isfile('.pythonrc.py'): exec(open('.pythonrc.py').read())`
:

```
import os
filename = os.environ.get('PYTHONSTARTUP')
if filename and os.path.isfile(filename):
    with open(filename) as fobj:
        startup_file = fobj.read()
    exec(startup_file)
```

16.1.4

Python `sitecustomize` `usercustomize` site-packages Python :

```
>>> import site
>>> site.getusersitepackages()
'/home/user/.local/lib/python3.5/site-packages'
```

`usercustomize.py` Python `-s`

`sitecustomize` site-packages `usercustomize` `site`

APPENDIX A

>>> Python

... Python

2to3 Python 2.x Python 3.x

2to3 `lib2to3` `Tools/scripts/2to3` 2to3-reference

abstract base class – ABC *duck-typing* `hasattr()` ABC `isinstance()` `issubclass()` `abc` Python ABC `collections.abc` `numbers` `io` `importlib.abc` `abc` ABC

annotation – *type hint*

`__annotations__`

variable annotation *function annotation* **PEP 484** **PEP 526**

argument – *function* *method*

- : `name=` `**` 3 5 `complex()` :

```
complex(real=3, imag=5)
complex(**{'real': 3, 'imag': 5})
```

- : / * *iterable* 3 5 :

```
complex(3, 5)
complex(*(3, 5))
```

calls

parameter **PEP 362**

asynchronous context manager – `__aenter__()` `__aexit__()` `async with` **PEP 492**

asynchronous generator – *asynchronous generator iterator* `async def` `yield` `async for`

`await` `async for` `async with`

asynchronous generator iterator – *asynchronous generator*

asynchronous iterator `__anext__()` `yield`

`yield` (try) `__anext__()` **PEP 492** **PEP 525**

asynchronous iterable – `async for` `__aiter__()` *asynchronous iterator* **PEP 492**

asynchronous iterator – `__aiter__()` `__anext__()` `__anext__` *awaitable* `async for` `__anext__()` `StopAsyncIteration` **PEP 492**

attribute – *o* *a* *o.a*

awaitable – `await` *coroutine* `__await__()` **PEP 492**

BDFL “ ” Guido van Rossum Python

binary file – *file object* `'rb'`, `'wb'` or `'rb+'` `sys.stdin.buffer` `sys.stdout.buffer` `io.BytesIO` `gzip.GzipFile`

text file `str`

bytes-like object – bufferobjects C-*contiguous* `bytes` `bytearray` `array.array` `memoryview`

“ ” `bytearray` `bytearray` `memoryview` (” ”) `bytes` `bytes` `memoryview`

bytecode – Python CPython Python `.pyc` ” ” *virtual machine* Python Python

dis

class –

class variable – ( )

coercion – `int(3.15)` `3` `3+4.5` int, float `TypeError` `float(3)+4.5` `3+4.5`

complex number – `-1` `i` `j` Python `j` `3+1j` `math` `cmath`

context manager – `with` `__enter__()` `__exit__()` **PEP 343**

contiguous – *C-* *Fortran* C Fortran C-Fortran

coroutine – `async def` **PEP 492**

coroutine function – *coroutine* `async def` `await` `async for` `async with` **PEP 492**

CPython Python python.org ”CPython” Jython IronPython

decorator – `@wrapper` `classmethod()` `staticmethod()`

:

```
def f(...):
    ...
f = staticmethod(f)

@staticmethod
def f(...):
    ...
```

descriptor – `__get__()`, `__set__()` `__delete__()` *a.b* *a* *b* *b* Python

descriptors

dictionary – `__hash__()` `__eq__()` Perl hash

dictionary view – `dict.keys()`, `dict.values()` `dict.items()` `list(dictview)` dict-views

docstring – `__doc__`

duck-typing – “ ” `type()` `isinstance()` ( ) `hasattr()` *EAFP*

EAFP “ ” Python `try` `except` *LBYL* C

expression – *statement* `while`

extension module – C C++ Python C API

f-string – **f-** `'f'` `'F'` “f- ” **PEP 498**

file object – API `read()` `write()` /

: , `io` `open()`

file-like object – *file object*

finder – *loader*

Python 3.3 : `sys.meta_path` *path entry finders* `sys.path_hooks`

PEP 302, **PEP 420** **PEP 451**

floor division – `//` `11 // 4` `2` `2.75` `(-11) // 4` `-3` `-2.75` **PEP 238**

function – *parameter*, *method* function

function annotation – *annotation*

`int` `int` :

```
def sum_two_numbers(a: int, b: int) -> int:
   return a + b
```

function

variable annotation **PEP 484**

___future___

__future__ :

```
>>> import __future__
>>> __future__.division
_Feature((2, 2, 0, 'alpha', 2), (3, 0, 0, 'alpha', 0), 8192)
```

garbage collection – Python gc

generator – *generator iterator* yield for- next()

generator iterator – *generator*

yield try

generator expression – for if :

```
>>> sum(i*i for i in range(10))         # sum of squares 0, 1, 4, ... 81
285
```

generic function –

single dispatch functools.singledispatch() **PEP 443**

GIL *global interpreter lock*

global interpreter lock – *CPython* Python *bytecode* dict
CPython

GIL I/O GIL

“ ”

hash-based pyc – **pyc** pyc-invalidation

hashable – __hash__() __eq__()

Python id()

IDLE Python IDE “ ” Python

immutable –

import path – *path based finder* sys.path __path__

importing – Python Python

importer – *finder* *loader*

interactive – Python python
help(x)

interpreted – Python / *interactive*

interpreter shutdown – Python

__main__

iterable – list str tuple dict __iter__() *Sequence* __getitem__()

for zip() map() ... iter() iter()
for *iterator sequence generator*

iterator – __next__() next() StopIteration
__next__() StopIteration __iter__()
list iter() for

typeiter

key function – locale.strxfrm()

Python min(), max(), sorted(), list.sort(), heapq.merge(), heapq.nsmallest(), heapq.nlargest() itertools.groupby()

str.lower() lambda lambda r: (r[0], r[2]) operator
attrgetter() itemgetter() methodcaller()

keyword argument – *argument*

lambda *expression* lambda lambda [parameters]: expression

LBYL “ ” *EAFP* if

LBYL “ ” “ ” if key in mapping: return mapping[key] *mapping*
key EAFP

list – Python *sequence* O(1)

list comprehension – result = ['{:#04x}'.format(x) for x in range(256)
if x % 2 == 0] 0 255 0x.. if range(256)

loader – load_module() *finder* **PEP 302** *abstract base class* importlib.abc.Loader

mapping – Mapping MutableMapping dict, collections.defaultdict,
collections.OrderedDict collections.Counter

meta path finder – sys.meta_path *finder* *path entry finders*

importlib.abc.MetaPathFinder

metaclass – Python

metaclasses

method – *argument* (self) *function* *nested scope*

method resolution order – Python 2.3 2.3 Python

module – Python Python *importing* Python
package

module spec – importlib.machinery.ModuleSpec

MRO *method resolution order*

mutable – id() *immutable*

named tuple – time.localtime() *year* t[0] t.
tm_year

time.struct_time collections.namedtuple()
Employee(name='jones', title='programmer')

namespace – builtins.open os.open()
random.seed() itertools.islice() random itertools

namespace package – **PEP 420** *package* *regular package* __init__.py
module

nested scope – nonlocal

new-style class – Python Python
__slots__ __getattribute__()

object – object *new-style class*

package – Python *module* __path__ Python
regular package *namespace package*

parameter – *function* *argument*

- *positional-or-keyword* *foo* *bar*:

```
def func(foo, bar=None): ...
```

- *positional-only* Python abs()
- *keyword-only* * *kw_only1* *kw_only2*:

```
def func(arg, *, kw_only1, kw_only2): ...
```

- *var-positional* * *args*:

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

- *var-keyword* ** *kwargs*

argument inspect.Parameter function **PEP 362**

path entry – *import path* *path based finder*

path entry finder – sys.path_hooks (*path entry hook*) *finder* *path entry*
importlib.abc.PathEntryFinder

path entry hook – *path entry* sys.path_hook *path entry finder*

path based finder – *import path*

path-like object – str bytes os.PathLike os.PathLike
os.fspath() str bytes os.fsdecode() os.fsencode() str bytes
PEP 519

PEP "Python " PEP Python Python PEP
PEP Python PEP
PEP 1

portion – zip **PEP 420**

positional argument – *argument*

provisional API – **API** API – API

API “ ” ——

PEP 411

provisional package – *provisional API*

Python 3000 Python 3.x 3 “Py3k”

Pythonic Python Python `for` Python :

```
for i in range(len(food)):
    print(food[i])
```

Pythonic :

```
for piece in food:
    print(piece)
```

qualified name – “ ” **PEP 3155** :

```
>>> class C:
...     class D:
...         def meth(self):
...             pass
...
>>> C.__qualname__
'C'
>>> C.D.__qualname__
'C.D'
>>> C.D.meth.__qualname__
'C.D.meth'
```

`email.mime.text`:

```
>>> import email.mime.text
>>> email.mime.text.__name__
'email.mime.text'
```

reference count – Python *CPython* `sys` `getrefcount()`

regular package – *package* `__init__.py`

namespace package

___slots___

sequence – *iterable* `__getitem__()` `__len__()` `list` `str` `tuple` `bytes` `dict` `__getitem__()` `__len__()` *immutable*

`collections.abc.Sequence` `__getitem__()` `__len__()` `count()`, `index()`, `__contains__()` `__reversed__()` `register()`

single dispatch – *generic function*

slice – *sequence* [] variable_name[1:3:5] slice

special method – Python specialnames

statement – " " *expression* if while for

struct sequence – *named tuple* _make() _asdict() sys.float_info os.stat()

text encoding – Unicode

text file – str *file object* *text encoding* 'r' 'w' sys.stdin sys.stdout io.StringIO

binary file

triple-quoted string – " '

type – Python __class__ type(obj)

type alias –

:

```
from typing import List, Tuple

def remove_gray_shades(
        colors: List[Tuple[int, int, int]]) -> List[Tuple[int, int, int]]:
    pass
```

:

```
from typing import List, Tuple

Color = Tuple[int, int, int]

def remove_gray_shades(colors: List[Color]) -> List[Color]:
    pass
```

typing **PEP 484**

type hint – *annotation*

Python IDE

typing.get_type_hints()

typing **PEP 484**

universal newlines – Unix '\n' Windows '\r\n' Macintosh '\r' **PEP 278** **PEP 3116** bytes.splitlines()

variable annotation – *annotation*

:

```
class C:
    field: 'annotation'
```

int :

```
count: int = 0
```

annassign

function annotation **PEP 484** **PEP 526**

virtual environment – Python Python Python

venv

virtual machine – Python *bytecode*

Zen of Python – **Python** Python "import this"

APPENDIX B

Sphinx Python reStructuredText
Python reporting-bugs

- Fred L. Drake, Jr. Python
- reStructuredText Docutils
- Fredrik Lundh Alternative Python Reference Sphinx

B.1 Python

Python Python Python Misc/ACKS Python
Python Python -

APPENDIX C

History and License

C.1 History of the software

Python was created in the early 1990s by Guido van Rossum at Stichting Mathematisch Centrum (CWI, see https://www.cwi.nl/) in the Netherlands as a successor of a language called ABC. Guido remains Python's principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for National Research Initiatives (CNRI, see https://www.cnri.reston.va.us/) in Reston, Virginia where he released several versions of the software.

In May 2000, Guido and the Python core development team moved to BeOpen.com to form the BeOpen PythonLabs team. In October of the same year, the PythonLabs team moved to Digital Creations (now Zope Corporation; see http://www.zope.com/). In 2001, the Python Software Foundation (PSF, see https://www.python.org/psf/) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

All Python releases are Open Source (see https://opensource.org/ for the Open Source Definition). Historically, most, but not all, Python releases have also been GPL-compatible; the table below summarizes the various releases.

	Derived from	Year	Owner	GPL compatible?
0.9.0 thru 1.2	n/a	1991-1995	CWI	yes
1.3 thru 1.5.2	1.2	1995-1999	CNRI	yes
1.6	1.5.2	2000	CNRI	no
2.0	1.6	2000	BeOpen.com	no
1.6.1	1.6	2001	CNRI	no
2.1	2.0+1.6.1	2001	PSF	no
2.0.1	2.0+1.6.1	2001	PSF	yes
2.1.1	2.1+2.0.1	2001	PSF	yes
2.1.2	2.1.1	2002	PSF	yes
2.1.3	2.1.2	2002	PSF	yes
2.2 and above	2.1.1	2001-now	PSF	yes

: GPL-compatible doesn't mean that we're distributing Python under the GPL. All Python licenses, unlike the GPL, let you distribute a modified version without making your changes open source. The GPL-compatible licenses make it possible to combine Python with other software that is released under the GPL; the others don't.

Thanks to the many outside volunteers who have worked under Guido's direction to make these releases possible.

C.2 Terms and conditions for accessing or otherwise using Python

C.2.1 PSF LICENSE AGREEMENT FOR PYTHON 3.7.2

1. This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and
   the Individual or Organization ("Licensee") accessing and otherwise using Python
   3.7.2 software in source or binary form and its associated documentation.

2. Subject to the terms and conditions of this License Agreement, PSF hereby
   grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce,
   analyze, test, perform and/or display publicly, prepare derivative works,
   distribute, and otherwise use Python 3.7.2 alone or in any derivative
   version, provided, however, that PSF's License Agreement and PSF's notice of
   copyright, i.e., "Copyright © 2001-2019 Python Software Foundation; All Rights
   Reserved" are retained in Python 3.7.2 alone or in any derivative version
   prepared by Licensee.

3. In the event Licensee prepares a derivative work that is based on or
   incorporates Python 3.7.2 or any part thereof, and wants to make the
   derivative work available to others as provided herein, then Licensee hereby
   agrees to include in any such work a brief summary of the changes made to Python
   3.7.2.

4. PSF is making Python 3.7.2 available to Licensee on an "AS IS" basis.
   PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF
   EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR
   WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE
   USE OF PYTHON 3.7.2 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 3.7.2
   FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF
   MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 3.7.2, OR ANY DERIVATIVE
   THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material breach of
   its terms and conditions.

7. Nothing in this License Agreement shall be deemed to create any relationship
   of agency, partnership, or joint venture between PSF and Licensee. This License
   Agreement does not grant permission to use PSF trademarks or trade name in a
   trademark sense to endorse or promote products or services of Licensee, or any
   third party.

```
8. By copying, installing or otherwise using Python 3.7.2, Licensee agrees
   to be bound by the terms and conditions of this License Agreement.
```

C.2.2 BEOPEN.COM LICENSE AGREEMENT FOR PYTHON 2.0

BEOPEN PYTHON OPEN SOURCE LICENSE AGREEMENT VERSION 1

```
1. This LICENSE AGREEMENT is between BeOpen.com ("BeOpen"), having an office at
   160 Saratoga Avenue, Santa Clara, CA 95051, and the Individual or Organization
   ("Licensee") accessing and otherwise using this software in source or binary
   form and its associated documentation ("the Software").

2. Subject to the terms and conditions of this BeOpen Python License Agreement,
   BeOpen hereby grants Licensee a non-exclusive, royalty-free, world-wide license
   to reproduce, analyze, test, perform and/or display publicly, prepare derivative
   works, distribute, and otherwise use the Software alone or in any derivative
   version, provided, however, that the BeOpen Python License is retained in the
   Software, alone or in any derivative version prepared by Licensee.

3. BeOpen is making the Software available to Licensee on an "AS IS" basis.
   BEOPEN MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED.  BY WAY OF
   EXAMPLE, BUT NOT LIMITATION, BEOPEN MAKES NO AND DISCLAIMS ANY REPRESENTATION OR
   WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE
   USE OF THE SOFTWARE WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

4. BEOPEN SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF THE SOFTWARE FOR
   ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF USING,
   MODIFYING OR DISTRIBUTING THE SOFTWARE, OR ANY DERIVATIVE THEREOF, EVEN IF
   ADVISED OF THE POSSIBILITY THEREOF.

5. This License Agreement will automatically terminate upon a material breach of
   its terms and conditions.

6. This License Agreement shall be governed by and interpreted in all respects
   by the law of the State of California, excluding conflict of law provisions.
   Nothing in this License Agreement shall be deemed to create any relationship of
   agency, partnership, or joint venture between BeOpen and Licensee.  This License
   Agreement does not grant permission to use BeOpen trademarks or trade names in a
   trademark sense to endorse or promote products or services of Licensee, or any
   third party.  As an exception, the "BeOpen Python" logos available at
   http://www.pythonlabs.com/logos.html may be used according to the permissions
   granted on that web page.

7. By copying, installing or otherwise using the software, Licensee agrees to be
   bound by the terms and conditions of this License Agreement.
```

C.2.3 CNRI LICENSE AGREEMENT FOR PYTHON 1.6.1

```
1. This LICENSE AGREEMENT is between the Corporation for National Research
   Initiatives, having an office at 1895 Preston White Drive, Reston, VA 20191
```

( )

( )

("CNRI"), and the Individual or Organization ("Licensee") accessing and
otherwise using Python 1.6.1 software in source or binary form and its
associated documentation.

2. Subject to the terms and conditions of this License Agreement, CNRI hereby
grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce,
analyze, test, perform and/or display publicly, prepare derivative works,
distribute, and otherwise use Python 1.6.1 alone or in any derivative version,
provided, however, that CNRI's License Agreement and CNRI's notice of copyright,
i.e., "Copyright © 1995-2001 Corporation for National Research Initiatives; All
Rights Reserved" are retained in Python 1.6.1 alone or in any derivative version
prepared by Licensee. Alternately, in lieu of CNRI's License Agreement,
Licensee may substitute the following text (omitting the quotes): "Python 1.6.1
is made available subject to the terms and conditions in CNRI's License
Agreement. This Agreement together with Python 1.6.1 may be located on the
Internet using the following unique, persistent identifier (known as a handle):
1895.22/1013. This Agreement may also be obtained from a proxy server on the
Internet using the following URL: http://hdl.handle.net/1895.22/1013."

3. In the event Licensee prepares a derivative work that is based on or
incorporates Python 1.6.1 or any part thereof, and wants to make the derivative
work available to others as provided herein, then Licensee hereby agrees to
include in any such work a brief summary of the changes made to Python 1.6.1.

4. CNRI is making Python 1.6.1 available to Licensee on an "AS IS" basis. CNRI
MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE,
BUT NOT LIMITATION, CNRI MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY
OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF
PYTHON 1.6.1 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. CNRI SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 1.6.1 FOR
ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF
MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 1.6.1, OR ANY DERIVATIVE
THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material breach of
its terms and conditions.

7. This License Agreement shall be governed by the federal intellectual property
law of the United States, including without limitation the federal copyright
law, and, to the extent such U.S. federal law does not apply, by the law of the
Commonwealth of Virginia, excluding Virginia's conflict of law provisions.
Notwithstanding the foregoing, with regard to derivative works based on Python
1.6.1 that incorporate non-separable material that was previously distributed
under the GNU General Public License (GPL), the law of the Commonwealth of
Virginia shall govern this License Agreement only as to issues arising under or
with respect to Paragraphs 4, 5, and 7 of this License Agreement. Nothing in
this License Agreement shall be deemed to create any relationship of agency,
partnership, or joint venture between CNRI and Licensee. This License Agreement
does not grant permission to use CNRI trademarks or trade name in a trademark
sense to endorse or promote products or services of Licensee, or any third
party.

( )

( )

```
8. By clicking on the "ACCEPT" button where indicated, or by copying, installing
   or otherwise using Python 1.6.1, Licensee agrees to be bound by the terms and
   conditions of this License Agreement.
```

C.2.4 CWI LICENSE AGREEMENT FOR PYTHON 0.9.0 THROUGH 1.2

```
Copyright © 1991 - 1995, Stichting Mathematisch Centrum Amsterdam, The
Netherlands.  All rights reserved.

Permission to use, copy, modify, and distribute this software and its
documentation for any purpose and without fee is hereby granted, provided that
the above copyright notice appear in all copies and that both that copyright
notice and this permission notice appear in supporting documentation, and that
the name of Stichting Mathematisch Centrum or CWI not be used in advertising or
publicity pertaining to distribution of the software without specific, written
prior permission.

STICHTING MATHEMATISCH CENTRUM DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS
SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO
EVENT SHALL STICHTING MATHEMATISCH CENTRUM BE LIABLE FOR ANY SPECIAL, INDIRECT
OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE,
DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS
SOFTWARE.
```

C.3 Licenses and Acknowledgements for Incorporated Software

This section is an incomplete, but growing list of licenses and acknowledgements for third-party software incorporated in the Python distribution.

C.3.1 Mersenne Twister

The `_random` module includes code based on a download from http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/MT2002/emt19937ar.html. The following are the verbatim comments from the original code:

```
A C-program for MT19937, with initialization improved 2002/1/26.
Coded by Takuji Nishimura and Makoto Matsumoto.

Before using, initialize the state by using init_genrand(seed)
or init_by_array(init_key, key_length).

Copyright (C) 1997 - 2002, Makoto Matsumoto and Takuji Nishimura,
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
```

( )

( )

```
are met:

 1. Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.

 2. Redistributions in binary form must reproduce the above copyright
    notice, this list of conditions and the following disclaimer in the
    documentation and/or other materials provided with the distribution.

 3. The names of its contributors may not be used to endorse or promote
    products derived from this software without specific prior written
    permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.


Any feedback is very welcome.
http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html
email: m-mat @ math.sci.hiroshima-u.ac.jp (remove space)
```

C.3.2

The `socket` module uses the functions, `getaddrinfo()`, and `getnameinfo()`, which are coded in separate source files from the WIDE Project, http://www.wide.ad.jp/.

```
Copyright (C) 1995, 1996, 1997, and 1998 WIDE Project.
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.
3. Neither the name of the project nor the names of its contributors
   may be used to endorse or promote products derived from this software
   without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE PROJECT AND CONTRIBUTORS ``AS IS'' AND
```

( )

```
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE PROJECT OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.
```

C.3.3 Asynchronous socket services

The `asynchat` and `asyncore` modules contain the following notice:

```
Copyright 1996 by Sam Rushing

                        All Rights Reserved

Permission to use, copy, modify, and distribute this software and
its documentation for any purpose and without fee is hereby
granted, provided that the above copyright notice appear in all
copies and that both that copyright notice and this permission
notice appear in supporting documentation, and that the name of Sam
Rushing not be used in advertising or publicity pertaining to
distribution of the software without specific, written prior
permission.

SAM RUSHING DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE,
INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN
NO EVENT SHALL SAM RUSHING BE LIABLE FOR ANY SPECIAL, INDIRECT OR
CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS
OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT,
NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN
CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
```

C.3.4 Cookie management

The `http.cookies` module contains the following notice:

```
Copyright 2000 by Timothy O'Malley <timo@alum.mit.edu>

               All Rights Reserved

Permission to use, copy, modify, and distribute this software
and its documentation for any purpose and without fee is hereby
granted, provided that the above copyright notice appear in all
copies and that both that copyright notice and this permission
notice appear in supporting documentation, and that the name of
```

( )

```
Timothy O'Malley  not be used in advertising or publicity
pertaining to distribution of the software without specific, written
prior permission.

Timothy O'Malley DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS
SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY
AND FITNESS, IN NO EVENT SHALL Timothy O'Malley BE LIABLE FOR
ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS,
WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR
PERFORMANCE OF THIS SOFTWARE.
```

C.3.5 Execution tracing

The `trace` module contains the following notice:

```
portions copyright 2001, Autonomous Zones Industries, Inc., all rights...
err...  reserved and offered to the public under the terms of the
Python 2.2 license.
Author: Zooko O'Whielacronx
http://zooko.com/
mailto:zooko@zooko.com

Copyright 2000, Mojam Media, Inc., all rights reserved.
Author: Skip Montanaro

Copyright 1999, Bioreason, Inc., all rights reserved.
Author: Andrew Dalke

Copyright 1995-1997, Automatrix, Inc., all rights reserved.
Author: Skip Montanaro

Copyright 1991-1995, Stichting Mathematisch Centrum, all rights reserved.


Permission to use, copy, modify, and distribute this Python software and
its associated documentation for any purpose without fee is hereby
granted, provided that the above copyright notice appears in all copies,
and that both that copyright notice and this permission notice appear in
supporting documentation, and that the name of neither Automatrix,
Bioreason or Mojam Media be used in advertising or publicity pertaining to
distribution of the software without specific, written prior permission.
```

C.3.6 UUencode and UUdecode functions

The `uu` module contains the following notice:

```
Copyright 1994 by Lance Ellinghouse
Cathedral City, California Republic, United States of America.
                       All Rights Reserved
Permission to use, copy, modify, and distribute this software and its
documentation for any purpose and without fee is hereby granted,
provided that the above copyright notice appear in all copies and that
both that copyright notice and this permission notice appear in
supporting documentation, and that the name of Lance Ellinghouse
not be used in advertising or publicity pertaining to distribution
of the software without specific, written prior permission.
LANCE ELLINGHOUSE DISCLAIMS ALL WARRANTIES WITH REGARD TO
THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND
FITNESS, IN NO EVENT SHALL LANCE ELLINGHOUSE CENTRUM BE LIABLE
FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Modified by Jack Jansen, CWI, July 1995:
- Use binascii module to do the actual line-by-line conversion
  between ascii and binary. This results in a 1000-fold speedup. The C
  version is still 5 times faster, though.
- Arguments more compliant with Python standard
```

C.3.7 XML Remote Procedure Calls

The `xmlrpc.client` module contains the following notice:

```
    The XML-RPC client interface is

Copyright (c) 1999-2002 by Secret Labs AB
Copyright (c) 1999-2002 by Fredrik Lundh

By obtaining, using, and/or copying this software and/or its
associated documentation, you agree that you have read, understood,
and will comply with the following terms and conditions:

Permission to use, copy, modify, and distribute this software and
its associated documentation for any purpose and without fee is
hereby granted, provided that the above copyright notice appears in
all copies, and that both that copyright notice and this permission
notice appear in supporting documentation, and that the name of
Secret Labs AB or the author not be used in advertising or publicity
pertaining to distribution of the software without specific, written
prior permission.

SECRET LABS AB AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD
TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANT-
ABILITY AND FITNESS.  IN NO EVENT SHALL SECRET LABS AB OR THE AUTHOR
BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY
DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS,
```

( )

```
WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS
ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE
OF THIS SOFTWARE.
```

C.3.8 test_epoll

The `test_epoll` module contains the following notice:

```
Copyright (c) 2001-2006 Twisted Matrix Laboratories.

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
"Software"), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE
LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

C.3.9 Select kqueue

The `select` module contains the following notice for the kqueue interface:

```
Copyright (c) 2000 Doug White, 2006 James Knight, 2007 Christian Heimes
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
```

( )

```
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.
```

C.3.10 SipHash24

The file `Python/pyhash.c` contains Marek Majkowski' implementation of Dan Bernstein's SipHash24 algorithm. The contains the following note:

```
<MIT License>
Copyright (c) 2013  Marek Majkowski <marek@popcount.org>

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.
</MIT License>

Original location:
   https://github.com/majek/csiphash/

Solution inspired by code from:
   Samuel Neves (supercop/crypto_auth/siphash24/little)
   djb (supercop/crypto_auth/siphash24/little2)
   Jean-Philippe Aumasson (https://131002.net/siphash/siphash24.c)
```

C.3.11 strtod and dtoa

The file `Python/dtoa.c`, which supplies C functions dtoa and strtod for conversion of C doubles to and from strings, is derived from the file of the same name by David M. Gay, currently available from http://www.netlib.org/fp/. The original file, as retrieved on March 16, 2009, contains the following copyright and licensing notice:

```
/****************************************************************
 *
 * The author of this software is David M. Gay.
 *
 * Copyright (c) 1991, 2000, 2001 by Lucent Technologies.
 *
 * Permission to use, copy, modify, and distribute this software for any
```

```
* purpose without fee is hereby granted, provided that this entire notice
* is included in all copies of any software which is or includes a copy
* or modification of this software and in all copies of the supporting
* documentation for such software.
*
* THIS SOFTWARE IS BEING PROVIDED "AS IS", WITHOUT ANY EXPRESS OR IMPLIED
* WARRANTY.  IN PARTICULAR, NEITHER THE AUTHOR NOR LUCENT MAKES ANY
* REPRESENTATION OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY
* OF THIS SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.
*
******************************************************************/
```

C.3.12 OpenSSL

The modules `hashlib`, `posix`, `ssl`, `crypt` use the OpenSSL library for added performance if made available by the operating system. Additionally, the Windows and Mac OS X installers for Python may include a copy of the OpenSSL libraries, so we include a copy of the OpenSSL license here:

```
LICENSE ISSUES
==============

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of
the OpenSSL License and the original SSLeay license apply to the toolkit.
See below for the actual license texts. Actually both licenses are BSD-style
Open Source licenses. In case of any license issues related to OpenSSL
please contact openssl-core@openssl.org.

OpenSSL License
---------------

  /* ====================================================================
   * Copyright (c) 1998-2008 The OpenSSL Project.  All rights reserved.
   *
   * Redistribution and use in source and binary forms, with or without
   * modification, are permitted provided that the following conditions
   * are met:
   *
   * 1. Redistributions of source code must retain the above copyright
   *    notice, this list of conditions and the following disclaimer.
   *
   * 2. Redistributions in binary form must reproduce the above copyright
   *    notice, this list of conditions and the following disclaimer in
   *    the documentation and/or other materials provided with the
   *    distribution.
   *
   * 3. All advertising materials mentioning features or use of this
   *    software must display the following acknowledgment:
   *    "This product includes software developed by the OpenSSL Project
   *    for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
   *
   * 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
```

```
    *    endorse or promote products derived from this software without
    *    prior written permission. For written permission, please contact
    *    openssl-core@openssl.org.
    *
    * 5. Products derived from this software may not be called "OpenSSL"
    *    nor may "OpenSSL" appear in their names without prior written
    *    permission of the OpenSSL Project.
    *
    * 6. Redistributions of any form whatsoever must retain the following
    *    acknowledgment:
    *    "This product includes software developed by the OpenSSL Project
    *    for use in the OpenSSL Toolkit (http://www.openssl.org/)"
    *
    * THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
    * EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
    * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
    * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE OpenSSL PROJECT OR
    * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
    * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
    * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
    * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
    * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
    * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
    * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
    * OF THE POSSIBILITY OF SUCH DAMAGE.
    * ====================================================================
    *
    * This product includes cryptographic software written by Eric Young
    * (eay@cryptsoft.com).  This product includes software written by Tim
    * Hudson (tjh@cryptsoft.com).
    *
    */

Original SSLeay License
-----------------------

   /* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
    * All rights reserved.
    *
    * This package is an SSL implementation written
    * by Eric Young (eay@cryptsoft.com).
    * The implementation was written so as to conform with Netscapes SSL.
    *
    * This library is free for commercial and non-commercial use as long as
    * the following conditions are aheared to.  The following conditions
    * apply to all code found in this distribution, be it the RC4, RSA,
    * lhash, DES, etc., code; not just the SSL code.  The SSL documentation
    * included with this distribution is covered by the same copyright terms
    * except that the holder is Tim Hudson (tjh@cryptsoft.com).
    *
    * Copyright remains Eric Young's, and as such any Copyright notices in
    * the code are not to be removed.
```

```
   * If this package is used in a product, Eric Young should be given attribution
   * as the author of the parts of the library used.
   * This can be in the form of a textual message at program startup or
   * in documentation (online or textual) provided with the package.
   *
   * Redistribution and use in source and binary forms, with or without
   * modification, are permitted provided that the following conditions
   * are met:
   * 1. Redistributions of source code must retain the copyright
   *    notice, this list of conditions and the following disclaimer.
   * 2. Redistributions in binary form must reproduce the above copyright
   *    notice, this list of conditions and the following disclaimer in the
   *    documentation and/or other materials provided with the distribution.
   * 3. All advertising materials mentioning features or use of this software
   *    must display the following acknowledgement:
   *    "This product includes cryptographic software written by
   *     Eric Young (eay@cryptsoft.com)"
   *    The word 'cryptographic' can be left out if the rouines from the library
   *    being used are not cryptographic related :-).
   * 4. If you include any Windows specific code (or a derivative thereof) from
   *    the apps directory (application code) you must include an acknowledgement:
   *    "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
   *
   * THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
   * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
   * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
   * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
   * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
   * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
   * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
   * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
   * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
   * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
   * SUCH DAMAGE.
   *
   * The licence and distribution terms for any publically available version or
   * derivative of this code cannot be changed.  i.e. this code cannot simply be
   * copied and put under another distribution licence
   * [including the GNU Public Licence.]
   */
```

C.3.13 expat

The `pyexpat` extension is built using an included copy of the expat sources unless the build is configured `--with-system-expat`:

```
Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd
                               and Clark Cooper

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
```

( )

```
"Software"), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,
TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

C.3.14 libffi

The `_ctypes` extension is built using an included copy of the libffi sources unless the build is configured `--with-system-libffi`:

```
Copyright (c) 1996-2008  Red Hat, Inc and others.

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
``Software''), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be included
in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ``AS IS'', WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT.  IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,
WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
DEALINGS IN THE SOFTWARE.
```

C.3.15 zlib

The `zlib` extension is built using an included copy of the zlib sources if the zlib version found on the system is too old to be used for the build:

```
Copyright (C) 1995-2011 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied
warranty.  In no event will the authors be held liable for any damages
arising from the use of this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
   claim that you wrote the original software. If you use this software
   in a product, an acknowledgment in the product documentation would be
   appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be
   misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly        Mark Adler
jloup@gzip.org          madler@alumni.caltech.edu
```

C.3.16 cfuhash

The implementation of the hash table used by the `tracemalloc` is based on the cfuhash project:

```
Copyright (c) 2005 Don Owens
All rights reserved.

This code is released under the BSD license:

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

  * Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.

  * Redistributions in binary form must reproduce the above
    copyright notice, this list of conditions and the following
    disclaimer in the documentation and/or other materials provided
    with the distribution.

  * Neither the name of the author nor the names of its
    contributors may be used to endorse or promote products derived
    from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
```

( )

```
FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
OF THE POSSIBILITY OF SUCH DAMAGE.
```

C.3.17 libmpdec

The `_decimal` module is built using an included copy of the libmpdec library unless the build is configured `--with-system-libmpdec`:

```
Copyright (c) 2008-2016 Stefan Krah. All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.
```

APPENDIX D

Python

History and License

Non-alphabetical

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z