

---

# The Python Language Reference

3.7.2

Guido van Rossum  
and the Python development team

12, 2019

Python Software Foundation  
Email: [docs@python.org](mailto:docs@python.org)



---

## Contents

---

<b>1</b>		<b>3</b>
1.1	.....	3
1.2	.....	3
<b>2</b>		<b>5</b>
2.1	.....	5
2.2	.....	7
2.3	.....	7
2.4	.....	8
2.5	.....	13
2.6	.....	13
<b>3</b>		<b>15</b>
3.1	.....	15
3.2	.....	15
3.3	.....	20
3.4	.....	31
<b>4</b>		<b>35</b>
4.1	.....	35
4.2	.....	35
4.3	.....	36
<b>5</b>		<b>39</b>
5.1	<code>importlib</code> .....	39
5.2	.....	39
5.3	.....	40
5.4	.....	41
5.5	.....	45
5.6	.....	46
5.7	<code>__main__</code> .....	46
5.8	.....	47
5.9	.....	47
<b>6</b>		<b>49</b>
6.1	.....	49
6.2	.....	49

6.3	.....	54
6.4	await .....	57
6.5	.....	57
6.6	.....	57
6.7	.....	57
6.8	.....	58
6.9	.....	58
6.10	.....	59
6.11	.....	61
6.12	.....	61
6.13	lambda .....	61
6.14	.....	62
6.15	.....	62
6.16	.....	62
<b>7</b>		<b>65</b>
7.1	.....	65
7.2	.....	66
7.3	assert .....	68
7.4	pass .....	68
7.5	del .....	68
7.6	return .....	68
7.7	yield .....	69
7.8	raise .....	69
7.9	break .....	70
7.10	continue .....	71
7.11	import .....	71
7.12	global .....	73
7.13	nonlocal .....	73
<b>8</b>		<b>75</b>
8.1	if .....	76
8.2	while .....	76
8.3	for .....	76
8.4	try .....	77
8.5	with .....	78
8.6	.....	79
8.7	.....	80
8.8	.....	81
<b>9</b>		<b>85</b>
9.1	Python .....	85
9.2	.....	85
9.3	.....	86
9.4	.....	86
<b>10</b>		<b>87</b>
<b>A</b>		<b>91</b>
<b>B</b>		<b>101</b>
B.1	Python .....	101
<b>C History and License</b>		<b>103</b>
C.1	History of the software .....	103

C.2	Terms and conditions for accessing or otherwise using Python . . . . .	104
C.3	Licenses and Acknowledgements for Incorporated Software . . . . .	107
<b>D</b>		<b>121</b>
		<b>123</b>



Python “ ”  
extending-index Python c-api-index library-index C/C++ tutorial-index C C++





---



---

Python

Python — Python :-)

” ” — CPython Python ( )

Python library-index

## 1.1

Python

:

**CPython** Python C

**Jython** Java Python Java Java **Jython**

**Python for .NET** CPython .NET .NET Brian Lloyd **Python for .NET**

**IronPython** .NET Python Python.NET IL Python Python .NET

Jython Jim Hugunin **IronPython**

**PyPy** Python Python JIT ( Python ) PyPy

Python

## 1.2

BNF :

```

name      ::=  lc_letter (lc_letter | "_")*
lc_letter ::=  "a"..."z"

name  lc_letter      lc_letter      lc_letter      'a' 'z' (
      (      )  ::=  (|)              (*)              (+)              ([ ])      (      ) *
+
      (      )      :              (      )      ASCII      (<...>)              ' ' ,
      :
      (" ")      BNF

```

---

Python  
Python      Unicode      UTF-8    **PEP 3120**      `SyntaxError`

## 2.1

Python

### 2.1.1

NEWLINE      NEWLINE (      )

### 2.1.2

ASCII CR (      )      - Unix    ASCII LF (      ), Windows    ASCII CR LF (      ), Macintosh  
Python      Python API    C    (    \n    ASCII LF    )

### 2.1.3

(#)

### 2.1.4

Python      `coding[=:]\s*([-w.]+)`

```
# -*- coding: <encoding-name> -*-
```

GNU Emacs

```
# vim:fileencoding=<encoding-name>
```

Bram Moolenaar VIM

UTF-8 UTF-8 (b'\xef\xbb\xbf') UTF-8 ( Microsoft **notepad** )

Python

## 2.1.5

(\) :

```
if 1900 < year < 2100 and 1 <= month <= 12 \
    and 1 <= day <= 31 and 0 <= hour < 24 \
    and 0 <= minute < 60 and 0 <= second < 60:    # Looks like a valid date
    return 1
```

( )

## 2.1.6

:

```
month_names = ['Januari', 'Februari', 'Maart',      # These are the
               'April',   'Mei',      'Juni',      # Dutch names
               'Juli',    'Augustus', 'September', # for the months
               'Oktober', 'November', 'December']  # of the year
```

NEWLINE ( )

## 2.1.7

( NEWLINE ) - - ( )

## 2.1.8

( )

( ) ( Unix )

TabError

: UNIX

( )

INDENT DEDENT

DEDENT

DEDENT

INDENT

( ) Python :

```
def perm(l):
    # Compute the list of all permutations of l
    if len(l) <= 1:
        return [l]
    r = []
    for i in range(len(l)):
        s = l[:i] + l[i+1:]
        p = perm(s)
        for x in p:
            r.append(l[i:i+1] + x)
    return r
```

:

```
def perm(l):                                # error: first line indented
for i in range(len(l)):                    # error: not indented
    s = l[:i] + l[i+1:]
    p = perm(l[:i] + l[i+1:])              # error: unexpected indent
    for x in p:
        r.append(l[i:i+1] + x)
    return r                                # error: inconsistent dedent
```

( — return r )

2.1.9

( ab a b )

2.2

NEWLINE, INDENT DEDENT : , , , ( )

2.3

( )

Python Unicode UAX-31 PEP 3131  
ASCII (U+0001..U+007F) Python 2.x : A Z \_ 0 9  
Python 3.0 ASCII ( PEP 3131) unicodedata Unicode

identifier ::= xid\_start xid\_continue\*  
id\_start ::= <all characters in general categories Lu, Ll, Lt, Lm, Lo, Nl, the underscore, and  
id\_continue ::= <all characters in id\_start, plus characters in the categories Mn, Mc, Nd, Pc and  
xid\_start ::= <all characters in id\_start whose NFKC normalization is in "id\_start xid\_continue\*"  
xid\_continue ::= <all characters in id\_continue whose NFKC normalization is in "id\_continue\*">

- Unicode :  
• *Lu* -  
• *Ll* -  
• *Lt* -  
• *Lm* -  
• *Lo* -  
• *Nl* -  
• *Mn* -  
• *Mc* -  
• *Nd* -  
• *Pc* -  
• *Other\_ID\_Start* - [PropList.txt](#)  
• *Other\_ID\_Continue* -

NFKC NFKC

Unicode 4.1 HTML <https://www.dcl.hpi.uni-potsdam.de/home/loewis/table-3131.html>

2.3.1

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

2.3.2

```
( ) :
_* from module import * _ builtins _ import
: _ gettext
_*-- ( ) Python --*--
_* " "
```

2.4

## 2.4.1

```

:

stringliteral ::= [stringprefix](shortstring | longstring)
stringprefix  ::= "r" | "u" | "R" | "U" | "f" | "F"
                | "fr" | "Fr" | "fR" | "FR" | "rf" | "rF" | "Rf" | "RF"
shortstring   ::= "'" shortstringitem* "'" | '"' shortstringitem* '"'
longstring    ::= "'" longstringitem* "'" | '"' longstringitem* '"'
shortstringitem ::= shortstringchar | stringescapeseq
longstringitem  ::= longstringchar | stringescapeseq
shortstringchar ::= <any source character except "\" or newline or the quote>
longstringchar  ::= <any source character except "\">
stringescapeseq ::= "\" <any source character>

bytesliteral   ::= bytesprefix(shortbytes | longbytes)
bytesprefix    ::= "b" | "B" | "br" | "Br" | "bR" | "BR" | "rb" | "rB" | "Rb" | "RB"
shortbytes     ::= "'" shortbytesitem* "'" | '"' shortbytesitem* '"'
longbytes      ::= "'" longbytesitem* "'" | '"' longbytesitem* '"'
shortbytesitem ::= shortbyteschar | bytesescapeseq
longbytesitem  ::= longbyteschar | bytesescapeseq
shortbyteschar ::= <any ASCII character except "\" or newline or the quote>
longbyteschar  ::= <any ASCII character except "\">
bytesescapeseq ::= "\" <any ASCII character>

stringprefix bytesprefix UTF-8
:      ( ' )      (      )      ( \ )
      'b' 'B' bytes str ASCII 128
      'r' 'R'      '\U' '\u' Python 2.x Python 3.x
'ur'

3.3 :      'rb' 'br'
3.3 :      (u'value') Python 2.x 3.x PEP 414
'f' 'F'      'f' 'r' 'b' 'u'
      ( )      ( " " ' " )
'r' 'R'      C      :

```

<code>\newline</code>		
<code>\\</code>	<code>(\)</code>	
<code>\'</code>	<code>(')</code>	
<code>\"</code>	<code>(")</code>	
<code>\a</code>	ASCII (BEL)	
<code>\b</code>	ASCII (BS)	
<code>\f</code>	ASCII (FF)	
<code>\n</code>	ASCII (LF)	
<code>\r</code>	ASCII (CR)	
<code>\t</code>	ASCII (TAB)	
<code>\v</code>	ASCII (VT)	
<code>\ooo</code>	<i>ooo</i>	(1,3)
<code>\xhh</code>	<i>hh</i>	(2,3)

:

<code>\N{name}</code>	Unicode <i>name</i>	(4)
<code>\uxxxx</code>	16 <i>xxxx</i>	(5)
<code>\Uxxxxxxxx</code>	32 16 <i>xxxxxxxx</i>	(6)

:

(1) C

(2) C

(3) Unicode

(4) 3.3 : <sup>1</sup>

(5)

(6) Unicode

C ( : )

3.6 : DeprecationWarning Python SyntaxError

`r"\\"`

:

`r"\\"`

(

)

( )

## 2.4.2

( )

`"hello" 'world' "helloworld"`

:

```
re.compile("[A-Za-z_]"      # letter or underscore
           "[A-Za-z0-9_]*"  # letter, digit or underscore
           )
```

`'+'`

(

)

<sup>1</sup> <http://www.unicode.org/Public/11.0.0/ucd/NameAliases.txt>



## 2.4.3

3.6 .

*f-string*    'f'    'F'                    {}  
               (                )                :

```
f_string      ::= (literal_char | "{" | ")") | replacement_field)*
replacement_field ::= "{" f_expression ["!" conversion] [":" format_spec] "}"
f_expression  ::= (conditional_expression | "*" or_expr)
                  ("," conditional_expression | "," "*" or_expr)* [","]
                  | yield_expression
conversion    ::= "s" | "r" | "a"
format_spec   ::= (literal_char | NULL | replacement_field)*
literal_char  ::= <any code point except "{", "}" or NULL>
```

```
'{'    '}'    '{'    Python    '!'    ':'
'}'

Python                lambda                (                )

'!'    str()    '!'    repr()    '!'    ascii()

format()                __format__()
```

:

```
>>> name = "Fred"
>>> f"He said his name is {name!r}."
"He said his name is 'Fred'."
>>> f"He said his name is {repr(name)}." # repr() is equivalent to !r
"He said his name is 'Fred'."
>>> width = 10
>>> precision = 4
>>> value = decimal.Decimal("12.34567")
>>> f"result: {value:{width}.{precision}}" # nested fields
'result:        12.35'
>>> today = datetime(year=2017, month=1, day=27)
>>> f"{today:%B %d, %Y}" # using date format specifier
'January 27, 2017'
>>> number = 1024
>>> f"{number:#0x}" # using integer format specifier
'0x400'
```

:

```
f"abc {a["x"]} def"        # error: outer string literal ended prematurely
f"abc {a['x']} def"        # workaround: use different quoting
```

:

```
f"newline: {ord('\n')}}" # raises SyntaxError
```

```
>>> newline = ord('\n')
>>> f"newline: {newline}"
'newline: 10'
```

```
>>> def foo():
...     f"Not a docstring"
...
>>> foo.__doc__ is None
True
```

PEP 498 str.format()

2.4.4

: ( )  
-1 ' ' 1

2.4.5

:

```
integer      ::=  decinteger | bininteger | octinteger | hexinteger
decinteger   ::=  nonzerodigit (["_"] digit)* | "0"+ (["_"] "0")*
bininteger   ::=  "0" ("b" | "B") (["_"] bindigit)+
octinteger   ::=  "0" ("o" | "O") (["_"] octdigit)+
hexinteger   ::=  "0" ("x" | "X") (["_"] hexdigit)+
nonzerodigit ::=  "1"..."9"
digit        ::=  "0"..."9"
bindigit     ::=  "0" | "1"
octdigit     ::=  "0"..."7"
hexdigit     ::=  digit | "a"..."f" | "A"..."F"
```

0x

Python 3.0 C

:

7	2147483647	0o177	0b100110111
3	79228162514264337593543950336	0o377	0xdeadbeef
	100_000_000_000		0b_1110_0101

3.6 :

2.4.6

:

floatnumber ::= pointfloat | exponentfloat

pointfloat ::= [digitpart] fraction | digitpart "."

exponentfloat ::= (digitpart | pointfloat) exponent

digitpart ::= digit ("\_" digit)\*

fraction ::= "." digitpart

exponent ::= ("e" | "E") ["+" | "-"] digitpart

10      077e010      77e10

:

3.14	10.	.001	1e100	3.14e-10	0e0	3.14_15_93
------	-----	------	-------	----------	-----	------------

3.6 :

2.4.7

:

imagnumber ::= (floatnumber | digitpart) ("j" | "J")

0.0                                      (3+4j)                                      :

3.14j	10.j	10j	.001j	1e100j	3.14e-10j	3.14_15_93j
-------	------	-----	-------	--------	-----------	-------------

2.5

:

+	-	*	**	/	//	%	@
<<	>>	&		^	~		
<	>	<=	>=	==	!=		

2.6

:

(	)	[	]	{	}		
,	:	.	;	@	=	->	
+=	-=	*=	/=	//=	%=	@=	
&=	=	^=	>>=	<<=	**=		

ASCII :

'	"	#	\
ASCII	Python	:	
\$	?	~	

## 3.1

Python Python . “ ”

`'is'` `id()`

**CPython implementation detail:** `CPython id(x) x`

`( " ") type() ( )` <sup>1</sup>

`( )`

---

**CPython implementation detail:** `CPython ( ) gc`

`CPython ( )`

`'try...except'`

`" "` `close()` `'try...finally'` `'with'`

`( )`

`:` `a = 1; b = 1 a b`

`c = []; d = [] c d` `( c = d = [] c d)`

## 3.2

Python ( C, Java ) Python ( )

, ,

---

<sup>1</sup>

None  
None  
None  
NotImplemented NotImplemented ( )  
implementing-the-arithmetic-operations  
Ellipsis ... Ellipsis  
numbers.Number Python  
Python :  
numbers.Integral ( )  
:  
(int)  
( ) 2  
(bool) False True False True 0 1  
"False" "True"  
  
numbers.Real (float) ( C Java ) Python  
Python  
numbers.Complex (complex) z z.real z.imag  
len() n 0, 1, ..., n-1 a i a[i]  
: a[i:j] k i <= k < j 0  
"step" " ": a[i:j:k] a x x = i + n\*k, n >= 0 i <= x < j  
:  
( )  
:  
Unicode U+0000 - U+10FFFF Python char 1  
ord() 0 - 10FFFF chr() 0 - 10FFFF 1 str.encode()  
str bytes bytes.decode()  
Python (' ') ( )  
8 0 <= x < 256 ( b'abc') bytes()  
decode()  
del ( )  
:  
Python ( 0 1 )  
bytearray() ( ) bytes  
array collections  
len()  
: ( 1 1.0)  
:  
set() add()  
frozenset() frozenset hashable

```

    a[k]    a    k    del    len()

:

                                :    (    1    1.0)

    {...}    (    )

dbm.ndbm dbm.gnu    collections

(    ):

    (    )

:
```

__doc__	None	
__name__		
__qualname__	<i>qualified name</i> 3.3 .	
__module__	None	
__defaults__	None	
__code__		
__globals__	—	
__dict__		
__closure__	None	cell_contents
__annotations__		'return'
__kwdefaults__		

”Writable”

cell\_contents

```

    (    )

: __self__    __func__    __doc__    ( __func__.__doc__    ) __name__    (
__func__.__name__    ) __module__    None

    (    )

    (    )

        __self__    __func__

        __func__    __func__

        __self__    __func__

    (__func__)    (__self__)    C    f()    x C    x.f(1)

C.f(x, 1)

    __self__    ” ”    x.f(1)    C.f(1)    f(C,1)    f

                                (    )

    yield    yield    iterator.__next__()

yield    return    StopIteration
```

```

    async def
    coroutine
    await
    async with
    async for

    async def
    yield
    async for
    iterator.__anext__()
    awaitable
    yield
    return
    StopAsyncIteration

    C
    len()
    math.sin() (math
    )
    C
    : __doc__
    None; __name__
    ; __self__
    None (
    ); __module__
    None
    C
    alist.append()
    alist
    __self__
    alist

    " "
    __new__()
    __new__()
    __init__()
    __call__()

    Python
    import
    importlib.import_module()
    __import__()
    __globals__
    m.x
    m.__dict__["x"]
    m.x = 1
    m.__dict__["x"] = 1
    (
    ) : __name__
    ; __doc__
    None; __annotations__ (
    )
    ; __file__
    __file__
    C
    ;
    : __dict__

    CPython implementation detail:
    CPython
    (
    )
    C.x
    C.__dict__["x"]
    (
    )
    C3
    ' '
    Python
    C3
    MRO
    2.3
    https://www.python.org/download/releases/2.3/mro/.
    (
    C)
    __self__
    C
    __dict__

    (
    )
    (
    )
    : __name__
    ; __module__
    ; __dict__
    ; __bases__
    ; __doc__
    None; __annotations__ (
    )
    (
    )
    "Classes"
    __dict__
    __self__
    __getattr__()

    __setattr__()
    __delattr__()

    : __dict__
    ; __class__

    I/O
    (
    )
    file object
    : open()
    os.popen(), os.fdopen()
    socket
    makefile()
    (
    )

    sys.stdin, sys.stdout
    sys.stderr
    io.TextIOBase

    Python
    bytecode
    (
    )
    (
    )
    (
    )
    : co_name
    ; co_argcount
    (
    ); co_nlocals
    (
    ); co_varnames
    (
    ); co_cellvars
    ; co_freevars
    ; co_code
    ; co_consts

```



```

        ; co_names          ; co_filename          ; co_firstlineno          ; co_lnotab
        ( ); co_stacksize   ( ); co_flags
co_flags : *arguments      0x04 ; **keywords      0x08`` ;
        ``0x20
        (from __future__ import division) co_flags : 0x2000 ;
Python 0x10 0x1000
co_flags
    co_consts          None
    ( )
    : f_back          ( ) None; f_code          ; f_locals          ; f_globals          ;
f_builtins          ( ) ; f_lasti          ( )
    : f_trace          None          ( ) - f_trace_lines False
        f_trace_opcodes True
f_lineno          —          ( ) f_lineno          Jump ( )
    :
frame.clear()
        ( )
        RuntimeError
3.4 .
        types.TracebackType
        ( try ) sys.exc_info()
__traceback__
    ( ) sys.last_traceback
        tb_next
    : tb_frame          ; tb_lineno          ; tb_lasti          except finally try
    : tb_next          ( ) None
3.7 : Python          tb_next
    __getitem__() slice()
    : start ; stop ; step ; None
    :
slice.indices(self, length)
        length          length          start stop step
staticmethod()
        ” ” classmethod()

```

### 3.3

```

    ( ) Python
type(x).__getitem__(x, i)
    ( AttributeError TypeError)
    None
    __iter__()
    None
    iter()
    TypeError ( __getitem__()).2
    ( W3C NodeList )

```

#### 3.3.1

```

object.__new__(cls[, ...])
    cls
    __new__()
    ( )
    ( ) __new__()
    ( cls )
    super().__new__(cls[, ...])
    __new__()
    cls
    __init__()
    __init__(self[, ...])
    self
    __new__()

    __new__()
    cls
    __init__()
    __new__()
    ( int, str tuple)

object.__init__(self[, ...])
    ( __new__())
    __init__()
    __init__()
    :
    super().__init__([args...]).
    __new__()
    __init__()
    ( __new__()
    __init__() )
    __init__()
    None
    TypeError

object.__del__(self)
    ( )
    __del__()
    __del__()
    __del__()
    ( ! )
    __del__()
    CPython
    __del__()

```

---

```

: del x
x.__del__() — x
x

```

---

**CPython implementation detail:** It is possible for a reference cycle to prevent the reference count of an object from going to zero. In this case, the cycle will be later detected and deleted by the *cyclic garbage collector*. A common cause of reference cycles is when an exception has been caught in a local variable. The frame's locals then reference the exception, which references its own traceback, which references the locals of all frames caught in the traceback.

```

:
gc

```

```

:
    __del__()
    sys.stderr
    • __del__()
    __del__()
    __del__()
    • __del__()
    None Python
    __del__()

```

---

<sup>2</sup> `__hash__()`, `__iter__()`, `__reversed__()` `__contains__()`

`TypeError` `None`

```
object.__repr__(self)
    repr()          “ ”          Python          <...some useful
    description...>          __repr__()  __str__()          “ ”          __repr__()
```

```
object.__str__(self)
    str(object)      format()  print()          “ ”
    object.__repr__()  __str__()          Python
    object          object.__repr__()
```

```
object.__bytes__(self)
    bytes          bytes
```

```
object.__format__(self, format_spec)
    format()          str.format()          “ ”  format_spec          format_spec
    __format__()
    formatspec
```

```
3.4 : object  __format__          TypeError
```

```
3.7 : object.__format__(x, '')      str(x)      format(str(self), '')
```

```
object.__lt__(self, other)
object.__le__(self, other)
object.__eq__(self, other)
object.__ne__(self, other)
object.__gt__(self, other)
object.__ge__(self, other)
    “ ”          x<y      x.__lt__(y) x<=y      x.__le__(y) x==y      x.__eq__(y) x!=y      x.
    __ne__(y) x>y      x.__gt__(y) x>=y      x.__ge__(y)
    NotImplemented          False  True          if      Python
    bool()
    __ne__()      __eq__()          NotImplemented          (x<y or x==y)
x<=y          functools.total_ordering()
    __hash__()          hashable
    __lt__()  __gt__()          __le__()  __ge__()          __eq__()  __ne__()
```

```
object.__hash__(self)
    hash()          set frozenset          dict __hash__()
    :
```

```
def __hash__(self):
    return hash((self.name, self.nick, self.color))
```

```
:  hash()          __hash__()          Py_ssize_t          64      8      32      4          __hash__()
    python -c "import sys; print(sys.hash_info.width)"
```

```
__eq__()          __hash__()          __eq__()          __hash__()          __eq__()
__hash__()
```

```

    __eq__() __hash__()          x.__hash__()          x == y    x is y    hash(x)
== hash(y)

```

```

    __eq__() __hash__() __hash__() None __hash__() None
TypeError isinstance(obj, collections.abc.Hashable)

```

```

    __eq__() __hash__() __hash__ = <ParentClass>.__hash__
    __eq__() __hash__ = None __hash__() TypeError
isinstance(obj, collections.abc.Hashable)

```

---

```

:      str bytes  datetime  __hash__()      “ ”      Python      Python
      O(n^2)      http://www.ocert.org/advisories/ocert-2011-003.html
      Python      32  64
PYTHONHASHSEED.

```

---

3.3 :

```

object.__bool__(self)
      bool()      False      True      __len__()      __len__()
__bool__()

```

### 3.3.2

```

      x.name      .
object.__getattr__(self, name)
      AttributeError      ( __getattr__()      name      self
      AttributeError      name      __get__()      AttributeError)      AttributeError
      __getattr__()      __getattr__()      __setattr__()
      __getattr__()      __getattr__()
object.__getattribute__(self, name)
      __getattr__()      __getattribute__()
      AttributeError      AttributeError      object.
__getattribute__(self, name)

```

---

:

---

```

object.__setattr__(self, name, value)
      name      value
      __setattr__()      object.__setattr__(self, name, value)
object.__delattr__(self, name)
      __setattr__()      del obj.name
object.__dir__(self)
      dir()      dir()

```

<code>__getattr__</code>	<code>__dir__</code>	<code>__getattr__</code>	<code>__dict__</code>
<code>AttributeError</code>	<code>object.__getattr__()</code>	<code>__getattr__</code>	<code>__dict__</code>
<code>AttributeError</code>			
<code>__dir__</code>	<code>dir()</code>		
	<code>__class__</code>	<code>types.ModuleType</code>	:

```
import sys
from types import ModuleType

class VerboseModule(ModuleType):
    def __repr__(self):
        return f'Verbose {self.__name__}'

    def __setattr__(self, attr, value):
        print(f'Setting {attr}...')
        super().__setattr__(attr, value)

sys.modules[__name__].__class__ = VerboseModule
```

:	<code>__getattr__</code>	<code>__class__</code>	-
---	--------------------------	------------------------	---

3.5 : `__class__`

3.7 : `__getattr__` `__dir__`

:

PEP 562 - `__getattr__` `__dir__` `__getattr__` `__dir__`

	" "	<code>__dict__</code>	
<code>object.__get__(self, instance, owner)</code>		None	<code>AttributeError</code>
<code>object.__set__(self, instance, value)</code>			
<code>instance</code>		<code>value</code>	
<code>object.__delete__(self, instance)</code>			
<code>instance</code>			
<code>object.__set_name__(self, owner, name)</code>			
<code>owner</code>		<code>name</code>	
3.6 .			
<code>__objclass__</code>	<code>inspect</code>		CPython C

" " `__get__()`, `__set__()` `__delete__()`

```

        a.x      a.__dict__['x']      type(a).__dict__['x']      type(a)
Python
a.x      a :
        : x.__get__(a)
a.x      : type(a).__dict__['x'].__get__(a, type(a))
A.x      : A.__dict__['x'].__get__(None, A)
a super      super(B, obj).m()      obj.__class__.__mro__      B      A      : A.
__dict__['m'].__get__(obj, obj.__class__)

        __get__() __set__()
__get__()      __set__() /      __delete__()
__get__() __set__()      __get__()      __set__()      __get__()
Python ( staticmethod() classmethod())
property()

```

### \_\_slots\_\_

```

__slots__      __dict__      __weakref__ (      __slots__      )
__dict__
object.__slots__      __slots__      __dict__      __weakref__

```

### \_\_slots\_\_

- `__slots__` `__dict__` `__weakref__`
- `__dict__` `__slots__` `AttributeError` `'__dict__'`
- `__weakref__` `__slots__` `'__weakref__'` `__slots__`
- `__slots__` ( ) `__slots__`
- `__slots__` `__slots__` `__dict__` `__weakref__` `__slots__`  
( )
- 
- `__slots__` “ ” `int bytes tuple`
- `__slots__`
- `__class__` `__slots__`
- `TypeError`

### 3.3.3

```

__init_subclass__      __init_subclass__

```

classmethod object.\_\_init\_subclass\_\_(cls)  
cls

\_\_init\_subclass\_\_ \_\_init\_subclass\_\_ :

```
class Philosopher:
    def __init_subclass__(cls, default_name, **kwargs):
        super().__init_subclass__(**kwargs)
        cls.default_name = default_name

class AustralianPhilosopher(Philosopher, default_name="Bruce"):
    pass
```

object.\_\_init\_subclass\_\_

:	metaclass	__init_subclass__	type(cls)
---	-----------	-------------------	-----------

3.6 .

type() type(name, bases, namespace)  
metaclass MyClass MySubclass Meta :

```
class Meta(type):
    pass

class MyClass(metaclass=Meta):
    pass

class MySubclass(MyClass):
    pass
```

- :
- MRO
  - 
  - 
  - 
  -

MRO

type \_\_mro\_entries\_\_  
:

PEP 560 - typing

```

    :
    •         type()
    •         type()
    •         type()

                type(cls)                                TypeError

                __prepare__          namespace = metaclass.__prepare__(name, bases, **kwds)

    __prepare__
:
PEP 3115 - Python 3000    __prepare__

    exec(body, globals(), namespace)          exec()

                __class__

                metaclass(name, bases, namespace, **kwds)          __prepare__
    __class__    super          super().    __class__          super()

CPython implementation detail: CPython 3.6    __class__    __classcell__
type.__new__          Python 3.6    DeprecationWarning    Python 3.8    RuntimeError
    type    type.__new__          :
    •    type.__new__          __set_name__()
    •    __set_name__
    •    __init_subclass__()

    type.__new__          __dict__
:
PEP 3135 -    __class__

/
```



## 3.3.4

```

    isinstance()  isinstance()
    abc.ABCMeta      ABC “ ”      ABC
class.__instancecheck__(self, instance)
    instance  class      isinstance(instance, class)
class.__subclasscheck__(self, subclass)
    Return true  subclass  class      issubclass(subclass, class)

:
PEP 3119 -      __instancecheck__()  __subclasscheck__()  isinstance()  issubclass()
              abc

```

## 3.3.5

```

    PEP 484      ( List[int])
classmethod object.__class_getitem__(cls, key)
    key

:
PEP 560 - typing

```

## 3.3.6

```

object.__call__(self[, args...])
    “ ”      x(arg1, arg2, ...)  x.__call__(arg1, arg2, ...)

```

## 3.3.7

```

                                k 0 <= k < N  N      keys(),
values(), items(), get(), clear(), setdefault(), pop(), popitem(), copy()  update()
Python      collections.abc      MutableMapping      __getitem__(), __setitem__(),
__delitem__()  keys()      Python      append(), count(), index(), extend(),
insert(), pop(), remove(), reverse()  sort()      __add__(), __radd__(), __iadd__(),
__mul__(), __rmul__()  __imul__()      __contains__()  in  in
__iter__()      __iter__()  keys()

object.__len__(self)
    len()      >= 0      __bool__()  __len__()

    CPython implementation detail:  CPython      sys.maxsize      sys.maxsize      ( len())
    OverflowError      OverflowError      __bool__()

object.__length_hint__(self)
    operator.length_hint()      >=

3.4 .

```

:

`a[1:2] = b``a[slice(1, 2, None)] = b`

None

<code>object.__getitem__(self, key)</code>					
<code>self[key]</code>			<code>__getitem__()</code>	<code>key</code>	<code>TypeError</code>
	<code>IndexError</code>	<code>key</code>	<code>KeyError</code>		

<code>: for</code>	<code>IndexError</code>
--------------------	-------------------------

<code>object.__setitem__(self, key, value)</code>			
<code>self[key]</code>	<code>__getitem__()</code>	<code>key</code>	<code>__getitem__()</code>

<code>object.__delitem__(self, key)</code>			
<code>self[key]</code>	<code>__getitem__()</code>	<code>key</code>	<code>__getitem__()</code>

<code>object.__missing__(self, key)</code>			
<code>dict.__getitem__()</code>	<code>dict</code>	<code>self[key]</code>	

`object.__iter__(self)`

typeiter

<code>object.__reversed__(self)</code>			
<code>reversed()</code>			
<code>__reversed__()</code>	<code>reversed()</code>	<code>(__len__() __getitem__())</code>	<code>reversed()</code>
<code>__reversed__()</code>			
<code>(in not in)</code>			

<code>object.__contains__(self, item)</code>			
<code>item self</code>			
<code>__contains__()</code>	<code>__iter__()</code>	<code>__getitem__()</code>	

### 3.3.8

```

object.__add__(self, other)
object.__sub__(self, other)
object.__mul__(self, other)
object.__matmul__(self, other)
object.__truediv__(self, other)
object.__floordiv__(self, other)
object.__mod__(self, other)
object.__divmod__(self, other)

```

```

object.__pow__(self, other[, modulo])
object.__lshift__(self, other)
object.__rshift__(self, other)
object.__and__(self, other)
object.__xor__(self, other)
object.__or__(self, other)
        (+, -, *, @, /, //, %, divmod(), pow(), **, <<, >>, &, ^, |)      x + y      x
    __add__()      x.__add__(y) __divmod__()      __floordiv__()      __mod__()
    __truediv__()      pow()      __pow__()

    NotImplemented

```

```

object.__radd__(self, other)
object.__rsub__(self, other)
object.__rmul__(self, other)
object.__rmatmul__(self, other)
object.__rtruediv__(self, other)
object.__rfloordiv__(self, other)
object.__rmod__(self, other)
object.__rdivmod__(self, other)
object.__rpow__(self, other)
object.__rlshift__(self, other)
object.__rrshift__(self, other)
object.__rand__(self, other)
object.__rxor__(self, other)
object.__ror__(self, other)
        (+, -, *, @, /, //, %, divmod(), pow(), **, <<, >>, &, ^, |)      3      4
    x - y      y      __rsub__()      x.__sub__(y)      NotImplemented      y.__rsub__(x)
    pow()      __rpow__() (      )

```

---

```

:
```

---

```

object.__iadd__(self, other)
object.__isub__(self, other)
object.__imul__(self, other)
object.__imatmul__(self, other)
object.__itruediv__(self, other)
object.__ifloordiv__(self, other)
object.__imod__(self, other)
object.__ipow__(self, other[, modulo])
object.__ilshift__(self, other)
object.__irshift__(self, other)
object.__iand__(self, other)
object.__ixor__(self, other)
object.__ior__(self, other)
        (+, -=, *=, @=, /=, //=, %=, **=, <=<=, >>=, &=, ^=, |=)      ( self)      (
    self)      x      __iadd__()      x += y      x = x.__iadd__(y)      x + y
    x.__add__(y)      y.__radd__(x)      ( faq-augmented-assignment-tuple-error)

object.__neg__(self)
object.__pos__(self)

```

---

```

3 " "      NotImplemented
4 ( __add__())

```

```

None —

```

```

object.__abs__(self)
object.__invert__(self)
    (-, +, abs() ~)

object.__complex__(self)
object.__int__(self)
object.__float__(self)
    complex(), int() float()

object.__index__(self)
    operator.index() Python bin(), hex() oct() )

```

---

```

:      __index__()      __int__()

```

---

```

object.__round__(self[, ndigits])
object.__trunc__(self)
object.__floor__(self)
object.__ceil__(self)
    round() math trunc(), floor() ceil() ndigits __round__()
Integral ( int)
    __int__() int() __trunc__()

```

### 3.3.9 with

```

with
    with with

typecontextmanager

object.__enter__(self)
    with as

object.__exit__(self, exc_type, exc_value, traceback)
    None

    __exit__()

:

```

**PEP 343** - "with" Python *with*

### 3.3.10

```

:
```

```

>>> class C:
...     pass
...
>>> c = C()
>>> c.__len__ = lambda: 5
>>> len(c)
Traceback (most recent call last):

```

( )

( )

```
File "<stdin>", line 1, in <module>
TypeError: object of type 'C' has no len()
```

```
__hash__() __repr__() :
```

```
>>> 1.__hash__() == hash(1)
True
>>> int.__hash__() == hash(int)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: descriptor '__hash__' of 'int' object needs an argument
```

```
‘ ’ :
```

```
>>> type(1).__hash__(1) == hash(1)
True
>>> type(int).__hash__(int) == hash(int)
True
```

```
__getattribute__() :
```

```
>>> class Meta(type):
...     def __getattribute__(*args):
...         print("Metaclass getattribute invoked")
...         return type.__getattribute__(*args)
...
>>> class C(object, metaclass=Meta):
...     def __len__(self):
...         return 10
...     def __getattribute__(*args):
...         print("Class getattribute invoked")
...         return object.__getattribute__(*args)
...
>>> c = C()
>>> c.__len__()                                # Explicit lookup via instance
Class getattribute invoked
10
>>> type(c).__len__(c)                         # Explicit lookup via type
Metaclass getattribute invoked
10
>>> len(c)                                    # Implicit lookup
10
```

```
__getattribute__()
```

## 3.4

### 3.4.1

*awaitable*      `__await__()`      *async def*      *Coroutine*

---

:	<code>types.coroutine()</code>	<code>asyncio.coroutine()</code>	<i>generator iterator</i>	<code>__await__()</code>
---	--------------------------------	----------------------------------	---------------------------	--------------------------

---

<code>object.__await__(self)</code>	<i>iterator</i>	<i>awaitable</i>	<code>asyncio.Future</code>	<i>await</i>
-------------------------------------	-----------------	------------------	-----------------------------	--------------

3.5 .

:

**PEP 492**

### 3.4.2

<i>Coroutine</i>	<i>awaitable</i>	<code>__await__()</code>	<code>StopIteration</code>	<i>value</i>
------------------	------------------	--------------------------	----------------------------	--------------

`StopIteration`

( - )

3.5.2 : `RuntimeError`

<code>coroutine.send(value)</code>	<i>value</i>	<code>None</code>	<code>__await__()</code>	<i>value</i>	<code>None</code>	<code>send()</code>
------------------------------------	--------------	-------------------	--------------------------	--------------	-------------------	---------------------

<code>StopIteration</code>	<code>__await__()</code>
----------------------------	--------------------------

<code>coroutine.throw(type[, value[, traceback]])</code>	<i>throw()</i>	<code>StopIteration</code>	<code>__await__()</code>
--	----------------	----------------------------	--------------------------

`coroutine.close()`

<i>close()</i>	<code>GeneratorExit</code>
----------------	----------------------------

### 3.4.3

`__anext__`

*async for*

`object.__aiter__(self)`

`object.__anext__(self)`

`StopAsyncIteration`

:

```
class Reader:
    async def readline(self):
        ...

    def __aiter__(self):
        return self

    async def __anext__(self):
        val = await self.readline()
```

( )

( )

```

if val == b'':
    raise StopAsyncIteration
return val

```

3.5 .

3.7 : Python 3.7 `__aiter__`

Python 3.7 `__aiter__` `TypeError`

### 3.4.4

`__aenter__` `__aexit__`

*async with*

```

object.__aenter__(self)
    __enter__()

```

```

object.__aexit__(self, exc_type, exc_value, traceback)
    __exit__()

```

:

```

class AsyncContextManager:
    async def __aenter__(self):
        await log('entering context')

    async def __aexit__(self, exc_type, exc, tb):
        await log('exiting context')

```

3.5 .





## 4.1

Python	Python	-c
	eval() exec()	

## 4.2

### 4.2.1

	<i>import</i>	<i>for</i>	<i>with</i>	<i>except</i>	<i>as</i>	<i>import</i>	<i>from ...</i>
import *							
del							
	<i>nonlocal</i>	<i>global</i>	(		)		

### 4.2.2

NameError	NameError	UnboundLocalError	UnboundLocalError
NameError			

# Python

```

    global
    global
    global
    global
    nonlocal
    SyntaxError
    __main__
    exec()
    eval()
    -
    :
```

```
class A:
    a = 42
    b = list(a + i for i in range(10))
```

### 4.2.3

```

CPython implementation detail:  __builtins__          import builtins
                                __builtins__          __main__    __builtins__
builtins    __builtins__  builtins

```

#### 4.2.4

42:

```
i = 10
def f():
    print(i)
i = 42
f()
```

```
eval()    exec()                                1 exec()    eval()
```

### 4.3

```
Python          Python          raise          try ...      except          finally

Python          “ ”

                                SystemExit

except
```

:	Python API	Python
---	------------	--------

---

1

*try*      *try*      *raise*      *raise*



---



---

```

    module Python    importing    import    importlib.import_module()
__import__()
import                import    __import__()    __import__()    import
    import
__import__()                ( sys.modules) import
import    __import__()    ( importlib.import_module())    __import__()

    Python    module 1    ModuleNotFoundError    Python

3.3 :    PEP 302    — sys.meta_path    ( PEP 420)

```

## 5.1 importlib

```

importlib    API    importlib.import_module()    __import__()    API
importlib

```

## 5.2

```

Python                Python C                Python

                __path__
                Python    sys    email    email.mime    email.mime.text

```

---

<sup>1</sup> `types.ModuleType`

### 5.2.1

Python                      Python 3.2                      `__init__.py`                      `__init__.py`  
                                  `__init__.py`                      Python                      Python  
                  parent                      :

```
parent/
  __init__.py
  one/
    __init__.py
  two/
    __init__.py
  three/
    __init__.py
```

parent.one      parent/`__init__.py`      parent/one/`__init__.py`      parent.two      parent.three  
 parent/two/`__init__.py`      parent/three/`__init__.py`

### 5.2.2

                                 zip                      Python  
                  `__path__`                      (                      `sys.path`)  
                  parent/`__init__.py`                      parent                      parent/one                      parent/two                      Python  
                  parent  
                  [PEP 420](#)

## 5.3

Python                      *import*                      `importlib.import_module()`                      `__import__()`  
                                  foo.bar.baz                      Python                      foo      foo.bar      foo.bar.baz  
 ModuleNotFoundError

### 5.3.1

`sys.modules`                      foo.bar.baz      `sys.modules`                      foo, foo.bar      foo.  
 bar.baz  
                  `sys.modules`                      None      ModuleNotFoundError                      Python  
`sys.modules`                      Python                      None  
 ModuleNotFoundError  
                  `sys.modules`                      `importlib.reload()`

5.3.2

sys.modules	Python			
Python		import path	import path	zip
URL				

3.4 : Python

5.3.3

:			
sys.modules	sys.path	sys.meta_path	
sys.path ( package.__path__ )		sys.path_hooks	

5.3.4

sys.modules	Python	sys.meta_path	find_spec()
	None	sys.meta_path	ModuleNotFoundError
find_spec()	foo.bar.baz	None	__path__
__path__	ModuleNotFoundError		
	foo.bar.baz	(mpf)	mpf.find_spec("foo", None, None)
foo foo.bar	mpf.find_spec("foo.bar", foo.__path__, None)	foo.bar	mpf.
find_spec("foo.bar.baz", foo.bar.__path__, None)			
None	None		
Python	sys.meta_path	import path	( path based finder )
3.4 :	find_spec()	find_module()	find_spec()

5.4

```

:
module = None
if spec.loader is not None and hasattr(spec.loader, 'create_module'):
    # It is assumed 'exec_module' will also be defined on the loader.
    module = spec.loader.create_module(spec)
if module is None:
    module = ModuleType(spec.name)
# The import-related module attributes get set here:
()
```

( )

```

_init_module_attrs(spec, module)

if spec.loader is None:
    if spec.submodule_search_locations is not None:
        # namespace package
        sys.modules[spec.name] = module
    else:
        # unsupported
        raise ImportError
elif not hasattr(spec.loader, 'exec_module'):
    module = spec.loader.load_module(spec.name)
    # Set __loader__ and __package__ if missing.
else:
    sys.modules[spec.name] = module
    try:
        spec.loader.exec_module(module)
    except BaseException:
        try:
            del sys.modules[spec.name]
        except KeyError:
            pass
        raise
return sys.modules[spec.name]

```

```

:
• sys.modules
• sys.modules sys.modules
• - sys.modules sys.modules sys.
modules
• “_init_module_attrs”
•
• exec_module() 2
3.4 : importlib.abc.Loader.load_module()

```

### 5.4.1

```

importlib.abc.Loader.exec_module() exec_module()
:
• Python (module.__dict__)
• ImportError exec_module()
find_spec() self
create_module() create_module() None
3.4 : create_module()

```

---

<sup>2</sup> importlib sys.modules sys.modules Python



```
3.4 : load_module()  exec_module()

        load_module()          exec_module()  load_module()          exec_module()
load_module()
:
• sys.modules          importlib.reload()          sys.modules
sys.modules
• sys.modules
• sys.modules
3.5 : exec_module()  create_module()  DeprecationWarning
3.6 : exec_module()  create_module()  ImportError
```

5.4.2

```
( importlib API, import  import-from  __import__())
foo  spam.foo  spam  foo  :  spam
```

```
spam/
__init__.py
foo.py
bar.py
```

```
spam/__init__.py  :
```

```
from .foo import Foo
from .bar import Bar
```

```
spam  foo  bar  :
```

```
>>> import spam
>>> spam.foo
<module 'spam.foo' from '/tmp/imports/spam/foo.py'>
>>> spam.bar
<module 'spam.bar' from '/tmp/imports/spam/bar.py'>
```

```
Python
( )  foo  sys.modules['spam']  sys.modules['spam.foo']
```

5.4.3

```
__spec__  ModuleSpec
3.4 .
```

5.4.4

```
__name__
    __name__
__loader__
    __loader__
__package__
    __package__          __name__          __package__          __name__
    __package__          PEP 366
    __name__          PEP 366          __spec__.parent
    3.6 : __package__          __spec__.parent
__spec__
    __spec__          __spec__          __main__          __spec__          None.
    __package__          __spec__.parent
    3.4 .
    3.6 : __package__          __spec__.parent
__path__
    __path__          __path__          __path__          __path__
    __path__
__file__
__cached__
    __file__          __file__ ( )
    __file__          __cached__          ( PEP 3147)
    __file__          __cached__          __file__ / __cached__
```

### 5.4.5 module.\_\_path\_\_

```
__path__
__path__          sys.path          __path__          sys.path
__path__          sys.path          __path__          sys.path_hooks ( )          __path__
__init__.py          __path__          PEP 420          PEP 420          __path__
__init__.py          __path__
```

### 5.4.6 repr

```
repr          repr
spec (__spec__)          repr          spec          repr          module.__name__, module.
__file__          module.__loader__          repr
:
• __spec__          repr          "name", "loader", "origin"          "has_location"
• __file__          repr
• __file__          __loader__          None          repr          repr
```

- repr      `__name__`

3.4 : `loader.module_repr()`      repr

Python 3.3      `module_repr()`      repr

### 5.4.7

Python `.pyc`      `.py`      Python  
 Python “ ”      `.pyc`      `.pyc`      Python  
     Python  
 --check-hash-based-pycs  
 3.7 :      `.pyc`      Python

## 5.5

Python      *path based finder* (`PathFinder`)      *import path*  
     Python      `(.py )` Python      `(.pyc )`      `( .so )`      `zipimport`  
     zip  
     URL  
     URL      HTTP      *path entry*  
*finder*  
     *meta path finder*      *path entry finder*  
     `sys.meta_path`  
     `sys.meta_path`

### 5.5.1

*path based finder*      *path entry*      Python  
     *path based finder*      `find_spec()`      *import path*  
     *path based finder*, `sys.path`, `sys.path_hooks`      `sys.path_importer_cache`      `__path__`  
  
     `sys.path`      `PYTHONPATH`      `sys.path`      `zip`      “ ”      `site`  
     URL      `sys.path`  
     *path based finder*      *meta path finder*      `find_spec()`      *import path*      `find_spec()`  
     `path`      ———      `__path__`      `path`      `None`      `sys.path`  
          *path entry finder* (`PathEntryFinder`)      `stat()`  
     `sys.path_importer_cache` (      *importer* <sub>3</sub> )      *path entry*      *path entry finder*  
          `sys.path_importer_cache`  
     `sys.path_hooks`      *path entry finder*      *import path*  
     `ImportError`      `ImportError`      *path entry*      *path entry finder*  
          UTF-8      `ImportError`  


---

     <sub>3</sub>      `sys.path_importer_cache`      `imp.NullImporter`      `None`      `portingpythoncode`

```
sys.path_hooks      path entry finder      find_spec()      sys.path_importer_cache      None
(      )      None      meta path finder

sys.path_hooks      path entry hook          path entry finder
-      -      sys.path      sys.path_importer_cache
sys.path_importer_cache      importlib.machinery.PathFinder.find_spec()
```

## 5.5.2

```
find_spec()

find_spec()      find_spec()      “ ”

portion      "loader"      None      "submodule_search_locations"

3.4 : find_spec()      find_loader()      find_module()      find_spec()

find_spec()      find_spec()

find_loader()      find_loader()      portion      None
Python      None

find_loader()      None

find_module()      find_module()      path

find_module()      find_loader()      find_module()
find_loader()      find_module()
```

## 5.6

```
sys.meta_path      ,
API      __import__()
find_spec()      ModuleNotFoundError      None
```

## 5.7 `__main__`

```
Python      __main__      __main__      sys      builtins
__main__
```

### 5.7.1 `__main__.__spec__`

```
__main__      __main__.__spec__      None
Python -m      __spec__      __spec__      __main__      zip      sys.path
__main__.__spec__      None      __main__      :

•
• -c
• stdin
```

```
•
__main__.__spec__ None __main__ -m
__main__ __main__.__spec__ if __name__ == "__main__":
__main__
```

5.8

```
XXX
XXX * (import_machinery.rst)
XXX runpy pkgutil “ ”
XXX __main__
XXX __main__ / ( PEP 395 )
```

5.9

```
Python
sys.meta_path PEP 302 PEP 420
PEP 420 Python 3.3 PEP 420 find_loader() find_module()
PEP 366 __package__
PEP 328 __name__ PEP 366 __package__
PEP 338
PEP 451 spec API
```



---

Python  
:

BNF

name ::= othername  
name othername

## 6.1

“ ” :

- 
- 
- 

‘%’

## 6.2

“ ” :

atom ::= *identifier* | *literal* | *enclosure*  
enclosure ::= *parenth\_form* | *list\_display* | *dict\_display* | *set\_display*  
| *generator\_expression* | *yield\_atom*

### 6.2.1

```
                                NameError

:                                Ham    __spam
_Ham__spam                      255
```

### 6.2.2

Python :

```
literal ::= stringliteral | bytesliteral
          | integer | floatnumber | imagnumber
```

### 6.2.3

```
parenth_form ::= "(" [starred_expression] ")"
```

— ” ”

### 6.2.4

Python “ ” :

- -
- :

```
comprehension ::= expression comp_for
comp_for      ::= ["async"] "for" target_list "in" or_test [comp_iter]
comp_iter     ::= comp_for | comp_if
comp_if       ::= "if" expression_nocond [comp_iter]
```

for for if for if

for “ ”

```
for for for : [x*y for x in
range(10) for y in range(x, x+10)].
```



yield yield from ( Python 3.7 DeprecationWarning Python 3.8+  
 SyntaxError)  
 Python 3.6 *async def* async for *asynchronous iterator* async def for  
 async for for async for *await* async for await  
 PEP 530  
 3.6 :  
 3.7 : yield and yield from

## 6.2.5

:  
 list\_display ::= "[" [starred\_list | *comprehension*] "]"

## 6.2.6

:  
 set\_display ::= "{" (starred\_list | *comprehension*) "}"  
 {}

## 6.2.7

/ :  
 dict\_display ::= "{" [key\_datum\_list | *dict\_comprehension*] "}"  
 key\_datum\_list ::= *key\_datum* ("," *key\_datum*)\* [","]  
 key\_datum ::= *expression* ":" *expression* | "\*\*" *or\_expr*  
 dict\_comprehension ::= *expression* ":" *expression comp\_for*

/ /  
 \*\* *mapping* /  
 3.5 : PEP 448  
 "for" "if"  
 ( *hashable* ) ( )

## 6.2.8

```

generator_expression ::= "(" expression comp_for ")"

        __next__()
for          : (x*y for x in range(10) for y in range(x, x+10)).

        yield yield from          ( Python 3.7          DeprecationWarning Python 3.8+
SyntaxError)
        async for await          ( )
3.6 :
3.7 : Python 3.7          async def          3.7
3.7 : yield and yield from

```

## 6.2.9 yield

```

yield_atom      ::= "(" yield_expression ")"
yield_expression ::= "yield" [expression_list | "from" expression]
yield generator asynchronous generator          yield          yield          async def
yield          :

```

```

def gen(): # defines a generator function
    yield 123

async def agen(): # defines an asynchronous generator function
    yield 123

```

```

        yield          ( Python 3.7          DeprecationWarning Python 3.8+
SyntaxError)
3.7 : yield

```

```

        yield          yield          expression_list
__next__() ( for next() ) None. send(), send
        yield          yield
try        yield          - close() . close finally
        yield from <expr>          send() throw()
send() AttributeError TypeError throw()
        StopIteration value yield StopIteration
3.3 : yield from <expr>

```

yield

:

PEP 255 - Python *yield*

PEP 342 - API

PEP 380 - `yield_from`

PEP 525 - PEP 492

-

ValueError

generator.\_\_next\_\_()

yield

\_\_next\_\_()

yield

None

yield

*expression\_list*

\_\_next\_\_()

StopIteration

*for*

next()

generator.send(value)

" "

*value*

yield

*send()*

StopIteration

*send()*

None

yield

generator.throw(*type*[, *value*[, *traceback*]])

*type*

StopIteration

generator.close()

GeneratorExit

GeneratorExit ( )

RuntimeError

*close()*

:

```
>>> def echo(value=None):
...     print("Execution starts when 'next()' is called for the first time.")
...     try:
...         while True:
...             try:
...                 value = (yield value)
...             except Exception as e:
...                 value = e
...     finally:
...         print("Don't forget to clean up when 'close()' is called.")
...
>>> generator = echo(1)
>>> print(next(generator))
Execution starts when 'next()' is called for the first time.
1
>>> print(next(generator))
None
>>> print(generator.send(2))
```

( )

( )

```

2
>>> generator.throw(TypeError, "spam")
TypeError('spam',)
>>> generator.close()
Don't forget to clean up when 'close()' is called.

```

yield from , “Python ” pep-380

*async def*      yield      *asynchronous generator*

*awaitable*      *async for*      *for*

yield      *expression\_list*

yield      yield

*\_\_anext\_\_()*      None      *asend()*

yield      *try*      then a yield expression within a *try*      yield

*finally*      - *aclose()*      *finally*

- *aclose()*      *sys.set\_asyncgen\_hooks()*      -

For a reference example of a      Lib/asyncio/base\_events.py      *asyncio.Loop*.

shutdown\_asyncgens

yield from <expr>

coroutine agen.*\_\_anext\_\_()*

yield      yield      *\_\_anext\_\_()*      yield      None      yield

yield      *expression\_list*      StopIteration      StopAsyncIteration

*async for*

coroutine agen.*asend(value)*

*send()*      “ ”      *value*      yield      *asend()*

StopIteration      StopAsyncIteration      *asend()*      None      yield

coroutine agen.*athrow(type[, value[, traceback]])*

*type*      StopIteration

StopAsyncIteration

coroutine agen.*aclose()*

GeneratorExit      GeneratorExit ( )      StopIteration

StopAsyncIteration      RuntimeError

*aclose()*

## 6.3

:

```
primary ::= atom | attributeref | subscription | slicing | call
```

### 6.3.1

```
:
```

```
attributeref ::= primary "." identifier
```

```
__getattr__()
```

```
AttributeError
```

### 6.3.2

```
:
```

```
subscription ::= primary "[" expression_list "]"
```

```
__getitem__()
```

```
:
```

```
__getitem__() ( x[-1] x )
```

### 6.3.3

```
del :
```

```
slicing ::= primary "[" slice_list "]"
```

```
slice_list ::= slice_item ("," slice_item)* [","]
```

```
slice_item ::= expression | proper_slice
```

```
proper_slice ::= [lower_bound] ":" [upper_bound] [ ":" [stride] ]
```

```
lower_bound ::= expression
```

```
upper_bound ::= expression
```

```
stride ::= expression
```

```
__getitem__()
start, stop step None
```

## 6.3.4

( *function*):

```

call                ::= primary "(" [argument_list [","] | comprehension] ")"
argument_list       ::= positional_arguments [", " starred_and_keywords]
                        [", " keywords_arguments]
                        | starred_and_keywords [", " keywords_arguments]
                        | keywords_arguments
positional_arguments ::= ["*"] expression ("," ["*"] expression)*
starred_and_keywords ::= ("*" expression | keyword_item)
                        ("," "*" expression | "," keyword_item)*
keywords_arguments  ::= (keyword_item | "***" expression)
                        ("," keyword_item | "," "***" expression)*
keyword_item         ::= identifier "=" expression

```

```

                                __call__()                                parameter
                                .      N      N
                                TypeError      None
                                TypeError
CPython implementation detail:      " "      CPython      C
PyArg_ParseTuple()
                                TypeError      *identifier
                                TypeError      **identifier
                                *expression  expression  iterable      f(x1, x2, *y, x3, x4)      y      y1,
..., yM      M+4      x1, x2, y1, ..., yM, x3, x4
                                *expression      **expression  -      :

```

```

>>> def f(a, b):
...     print(a, b)
...
>>> f(b=1, *(2,))
2 1
>>> f(a=1, *(2,))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: f() got multiple values for keyword argument 'a'
>>> f(1, *(2,))
1 2

```

```

                                *expression
                                **expression  expression  mapping                                TypeError
                                *identifier  **identifier
3.5 :      *  **      (*)      (**)      PEP 448
                                None
—

```

```

:                                     return
:
:      built-in-funcs
:
:
:
:      __call__()
```

6.4 await

*coroutine*      *awaitable*      *coroutine function*

`await_expr ::= "await" primary`

3.5 .

6.5

:

`power ::= (await_expr | primary) ["**" u_expr]`  
`: -1**2      -1`

`pow()`  
`int                                  float      float      10**2   100 10**-2   0.01`  
`0.0            ZeroDivisionError                  complex                  ValueError`

6.6

:

`u_expr ::= power | "-" u_expr | "+" u_expr | "~" u_expr`  
`- ( )`  
`+ ( )`  
`~ ( )                  x                  -(x+1)`  
`TypeError`

6.7

:

```

m_expr ::= u_expr | m_expr "*" u_expr | m_expr "@" m_expr |
           m_expr "/" u_expr | m_expr "/" u_expr |
           m_expr "%" u_expr
a_expr ::= m_expr | a_expr "+" m_expr | a_expr "-" m_expr

* ( )
@ (at)          Python
3.5 .
/ ( ) // ( )    float          'floor'          ZeroDivisionError
% ( )           ZeroDivisionError          3.14%0.7  0.34 ( 3.14  4*0.7 +
0.34)           1
: x == (x//y)*y + (x%y)          divmod() : divmod(x, y) == (x//y, x%y) 2
%
divmod()         Python  old-string-formatting
+ (addition)
- ( )

```

## 6.8

```

:

shift_expr ::= a_expr | shift_expr ("<<" | ">>") a_expr

n      pow(2,n)      n      pow(2,n)

```

## 6.9

```

:

and_expr ::= shift_expr | and_expr "&" shift_expr
xor_expr ::= and_expr | xor_expr "^" and_expr
or_expr  ::= xor_expr | or_expr "|" xor_expr

&      AND ( )
~      XOR ( )
|      OR ( )

```

---

<sup>1</sup> `abs(x%y) < abs(y)`      Python    IEEE 754    `-1e-100 % 1e100`    `1e100`    `-1e-100`  
+ `1e100`    `1e100`    `math.fmod()`    `-1e-100`  
<sup>2</sup> `x`    `y`    `x//y`    `(x-x%y)//y`    Python    `divmod(x,y)[0] * y + x % y`    `x`.



6.10

C Python C a < b < c :

```
comparison ::= or_expr (comp_operator or_expr)*
comp_operator ::= "<" | ">" | "==" | ">=" | "<=" | "!="
               | "is" ["not"] | ["not"] "in"

: True False
x < y <= z x < y and y <= z y x < y z
a, b, c, ..., y, z op1, op2, ..., opN a op1 b op2 c ... y opN z a op1 b and b op2 c
and ... y opN z
a op1 b op2 c a c x < y > z
```

6.10.1

<, >, ==, >=, <= !=

Python

object object \_\_lt\_\_()

(== !=) ( x is y x == y)

(<, >, <= >=) TypeError

- (typesnumeric) fractions.Fraction decimal.Decimal

float('NaN') decimal.Decimal('NaN') x = float('NaN')

3 < x, x < 3, x == x, x != x IEEE 754

- (bytes bytearray )
- (str ) Unicode ( ord() ) 3

- (tuple, list range ) range TypeError

x, x == x :

```
>>> nan = float('NaN')
>>> nan is nan
True
```

( )

3 Unicode ( U+0041) ( " A") Unicode " C" U+00C7

U+0043 ( C) U+0327 ( )

Unicode "\u00C7" == "\u0043\u0327" False " C"

unicodedata.normalize()

( )

```
>>> nan == nan
False          <-- the defined non-reflexive behavior of NaN
>>> [nan] == [nan]
True           <-- list enforces reflexivity and tests identity first
```

- `:`
- `[1,2] == (1,2)`
- `[1,2,x] <= [1,2,y]` “`x <= y`” `[1,2] < [1,2,3]`
- `(dict )` `(, )`
- `(<, >, <= >=)` `TypeError`
- `(set frozenset )`
- `{1,2}` `{2,3}` `min(), max()`
- `sorted()`
- `:`
- `:`
- `x is y` `x == y`
- `:`
- `x == y` `y == x`
- `x != y` `y != x`
- `x < y` `y > x`
- `x <= y` `y >= x`
- `:`
- `x > y and y > z` `x > z`
- `x < y and y <= z` `x < z`
- `:`
- `x == y` `not x != y`
- `x < y` `not x >= y ( )`
- `x > y` `not x <= y ( )`
- `total_ordering()`
- `hash()`

Python

## 6.10.2

`in` `not in` `x s` `x in s` `True` `False` `x not in s` `x in s` `in`  
 list, tuple, set, frozenset, dict `collections.deque` `x in y` `any(x is e or x == e for e`  
 in y)

```

    x y    x in y  True      y.find(x) != -1      "" in "abc"  True
__contains__()      y.__contains__(x)    x in y  True  False
__contains__()  __iter__()      y      z x == z  x in y  True      in
    __getitem__()      i  x == y[i]      IndexError  x in y  True      in

not in      in

```

### 6.10.3

```

is  is not      x y      x is y      id()      x is not y      4

```

## 6.11

```

or_test    ::=  and_test | or_test "or" and_test
and_test   ::=  not_test | and_test "and" not_test
not_test   ::=  comparison | "not" not_test

                : False, None,                                __bool__()

not          True  False
x and y     x      x      y
x or y      x      x      y
and  or          False  True      s      s or 'foo'      not
                not 'foo'  False  ''

```

## 6.12

```

conditional_expression ::= or_test ["if" or_test "else" expression]
expression              ::= conditional_expression | lambda_expr
expression_nocond       ::= or_test | lambda_expr_nocond

```

“ ” Python

```

x if C else y      C  x      C  x      y

```

PEP 308

## 6.13 lambda

```

lambda_expr      ::=  "lambda" [parameter_list] ":" expression
lambda_expr_nocond ::=  "lambda" [parameter_list] ":" expression_nocond
lambda           lambda      lambda parameters: expression      :

```

4

is

```
def <lambda>(parameters):  
    return expression
```

lambda

## 6.14

```
expression_list    ::=  expression ("," expression)* [","]  
starred_list       ::=  starred_item ("," starred_item)* [","]  
starred_expression ::=  expression | (starred_item ",")* [starred_item]  
starred_item       ::=  expression | "*" or_expr
```

```
    *          iterable  
3.5  :          PEP 448  
      ( )          : ()
```

## 6.15

Python

:

```
expr1, expr2, expr3, expr4  
(expr1, expr2, expr3, expr4)  
{expr1: expr2, expr3: expr4}  
expr1 + expr2 * (expr3 - expr4)  
expr1(expr2, expr3, *expr4, **expr5)  
expr3, expr4 = expr1, expr2
```

## 6.16

Python

<i>lambda</i>	lambda
<i>if</i> - <i>else</i>	
<i>or</i>	OR
<i>and</i>	AND
<i>not</i> <i>x</i>	NOT
<i>in</i> , <i>not in</i> , <i>is</i> , <i>is not</i> , <i>&lt;</i> , <i>&lt;=</i> , <i>&gt;</i> , <i>&gt;=</i> , <i>!=</i> , <i>==</i>	
	OR
^	XOR
&	AND
<<, >>	
+, -	
*, @, /, //, %	<sup>5</sup>
+ <i>x</i> , - <i>x</i> , ~ <i>x</i>	NOT
**	<sup>6</sup>
<i>await</i> <i>x</i>	await
<i>x</i> [ <i>index</i> ], <i>x</i> [ <i>index: index</i> ], <i>x</i> ( <i>arguments...</i> ), <i>x</i> . <i>attribute</i>	
( <i>expressions...</i> ), [ <i>expressions...</i> ], { <i>key</i> : <i>value...</i> }, { <i>expressions...</i> }	

<sup>5</sup> %

<sup>6</sup> \*\*

2\*\*-1 0.5



:

```
simple_stmt ::= expression_stmt
            | assert_stmt
            | assignment_stmt
            | augmented_assignment_stmt
            | annotated_assignment_stmt
            | pass_stmt
            | del_stmt
            | return_stmt
            | yield_stmt
            | raise_stmt
            | break_stmt
            | continue_stmt
            | import_stmt
            | future_stmt
            | global_stmt
            | nonlocal_stmt
```

## 7.1

( Python None ) :

```
expression_stmt ::= starred_expression
```

None repr() None

## 7.2

```

:

assignment_stmt ::= (target_list "=") + (starred_expression | yield_expression)
target_list     ::= target ("," target)* [","]
target          ::= identifier
                  | "(" [target_list] ")"
                  | "[" [target_list] "]"
                  | attributeref
                  | subscription
                  | slicing
                  | "*" target

(           ,           )

•
•
—           “ ”
—
•           :
—           global nonlocal
—           nonlocal

•           TypeError ( AttributeError )
           a.x           a.x           a.x
           :

class Cls:
    x = 3           # class variable
inst = Cls()
inst.x = inst.x + 1 # writes inst.x as 4 leaving Cls.x as 3

property()

•
           IndexError ( )
           / / /
           __setitem__()

```



•

CPython implementation detail:

“ ”                    a, b = b, a                    [0, 2]:

```
x = [0, 1]
i = 0
i, x[i] = 1, 2                    # i is updated, then x[i] is updated
print(x)
```

:
PEP 3132 -                    \*target

7.2.1

:

augmented\_assignment\_stmt ::= *augtarget augop (expression\_list | yield\_expression)*
augtarget ::= *identifier | attributeref | subscription | slicing*
augop ::= *"+=" | "-=" | "\*=" | "@=" | "/=" | "//=" | "%=" | "\*\*="*
          *| ">>=" | "<=>" | "&=" | "^=" | "|="*

x += 1            x = x + 1                    x
                  a[i] += f(x)    a[i]    f(x)                    a[i]

7.2.2

:

annotated\_assignment\_stmt ::= *augtarget ":" expression ["=" expression]*

\_\_annotations\_\_

*\_\_setitem\_\_()* *\_\_setattr\_\_()*

:
PEP 526 -
PEP 484 -                    typing                    IDE

## 7.3 assert

`assert` :

`assert_stmt ::= "assert" expression ["," expression]`

`assert expression`

```
if __debug__:
    if not expression: raise AssertionError
```

`assert expression1, expression2`

```
if __debug__:
    if not expression1: raise AssertionError(expression2)
```

`__debug__`    `AssertionError`                      `__debug__`    `True`    `False` (    `-0`)  
                 `assert`

`__debug__`

## 7.4 pass

`pass_stmt ::= "pass"`

`pass` — :

```
def f(arg): pass    # a function that does nothing (yet)
class C: pass       # a class with no methods (yet)
```

## 7.5 del

`del_stmt ::= "del" target_list`

`global`                      `NameError`

3.2 :

## 7.6 return

`return_stmt ::= "return" [expression_list]`

```

return
    None
return      ( None)
return      finally try finally
    return      StopIteration      StopIteration      StopIteration.value
    return      StopAsyncIteration      return

```

## 7.7 yield

```

yield_stmt ::= yield_expression
yield      yield      yield      yield      yield

```

```

yield <expr>
yield from <expr>

```

```

yield

```

```

(yield <expr>)
(yield from <expr>)

```

```

yield      generator      yield
yield      yield

```

## 7.8 raise

```

raise_stmt ::= "raise" [expression ["from" expression]]
raise      RuntimeError
raise      BaseException

```

```

__traceback__      with_traceback()      :

```

```

raise Exception("foo occurred").with_traceback(tracebackobj)

```

```

from      __cause__      :

```

```

>>> try:
...     print(1 / 0)
... except Exception as exc:
...     raise RuntimeError("Something bad happened") from exc
...
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
ZeroDivisionError: division by zero

```

( )

( )

The above exception was the direct cause of the following exception:

```
Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
RuntimeError: Something bad happened
```

*finally* clause: `__context__` :

```
>>> try:
...     print(1 / 0)
... except:
...     raise RuntimeError("Something bad happened")
...
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
ZeroDivisionError: division by zero
```

During handling of the above exception, another exception occurred:

```
Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
RuntimeError: Something bad happened
```

from `None` :

```
>>> try:
...     print(1 / 0)
... except:
...     raise RuntimeError("Something bad happened") from None
...
Traceback (most recent call last):
  File "<stdin>", line 4, in <module>
RuntimeError: Something bad happened
```

*try*

3.3 : `None` `raise X from Y` `Y`

3.3 : `__suppress_context__`

## 7.9 break

```
break_stmt ::= "break"
break      for while
           else
           for break
break      finally try finally
```

## 7.10 continue

```

continue_stmt ::= "continue"
continue      for while                               finally
continue      finally try                             finally

```

## 7.11 import

```

import_stmt    ::= "import" module ["as" identifier] ("," module ["as" identifier])*
                  | "from" relative_module "import" identifier ["as" identifier]
                  | "from" relative_module "import" "(" identifier ["as" identifier]
                  | "from" module "import" "*"
module         ::= (identifier ".")* identifier
relative_module ::= "."* module | "."+

```

```

import from :
1.
2. import
import

```

```

:
• as as
•
•

```

```

from :
1. from
2. import
1.
2.
3. ImportError
4. as

```

```

:
```

import foo	# foo imported and bound locally
import foo.bar.baz	# foo.bar.baz imported, foo bound locally
import foo.bar.baz as fbb	# foo.bar.baz imported and bound as fbb
from foo.bar import baz	# foo.bar.baz imported and bound as baz
from foo import attr	# foo imported and foo.attr bound as attr

```

(' * ')      import
              __all__
              ('_')  __all__  API  __all__  API  __all__
— from module import * —      SyntaxError
                                from
mod      pkg.subpkg1      from . import mod      pkg      pkg.mod      from ..subpkg2 import
                                PEP 328
importlib.import_module()

```

### 7.11.1 future

*future* Python

future Python

```

future_stmt ::= "from" "__future__" "import" feature ["as" identifier]
              ("," feature ["as" identifier])*
              | "from" "__future__" "import" "(" feature ["as" identifier]
              ("," feature ["as" identifier])* [","] ")"
feature      ::= identifier

```

future future :

- 
- 
- 
- future

Python 3.7 future

future Python 3 absolute\_import, division, generators, generator\_stop,  
unicode\_literals, print\_function, nested\_scopes with\_statement

future

```

future
import      __future__  future
future
:
```

```
import __future__ [as name]
```

future import

Code compiled by calls to the `exec()` `compile()` future M future  
`compile()` —

future -i future

:

PEP 236 - \_\_future\_\_ \_\_future\_\_

## 7.12 global

```

global_stmt ::= "global" identifier ("," identifier)*
global
global global
global for class import

```

CPython implementation detail:

```

: global global exec() global
global eval() compile()

```

## 7.13 nonlocal

```

nonlocal_stmt ::= "nonlocal" identifier ("," identifier)*
nonlocal
global nonlocal
nonlocal
:
PEP 3104 - nonlocal

```





---

---

```
if, while for          try          /          with
    ‘ , ‘ ,
                else if :
```

```
if test1: if test2: print(x)
```

```
    print()      :
```

```
if x < y < z: print(x); print(y); print(z)
```

```
:
```

```
compound_stmt ::= if_stmt
                | while_stmt
                | for_stmt
                | try_stmt
                | with_stmt
                | funcdef
                | classdef
                | async_with_stmt
                | async_for_stmt
                | async_funcdef
suite          ::= stmt_list NEWLINE | NEWLINE INDENT statement+ DEDENT
statement      ::= stmt_list NEWLINE | compound_stmt
stmt_list      ::= simple_stmt (";" simple_stmt)* [";"]

NEWLINE      DEDENT          ‘ else’ Python if )
```

## 8.1 if

*if* :

```
if_stmt ::= "if" expression ":" suite
          ("elif" expression ":" suite)*
          ["else" ":" suite]

          if else
```

## 8.2 while

*while* :

```
while_stmt ::= "while" expression ":" suite
             ["else" ":" suite]

             else
break else continue
```

## 8.3 for

*for* :

```
for_stmt ::= "for" target_list "in" expression_list ":" suite
            ["else" ":" suite]

            expression_list ( )
( StopIteration ) else
break else continue else
```

The for-loop makes assignments to the variables(s) in the target list. This overwrites all previous assignments to those variables including those made in the suite of the for-loop:

```
for i in range(10):
    print(i)
    i = 5           # this will not affect the for-loop
                   # because i will be overwritten with the next
                   # index in the range
```

```
range()           Pascal for i := a to b do   list(range(3))

[0, 1, 2]
```

---

:

```
for x in a[:]:
    if x < 0: a.remove(x)
```

8.4 try

*try* / :

```
try_stmt ::= try1_stmt | try2_stmt
try1_stmt ::= "try" ":" suite
              ("except" [expression ["as" identifier]] ":" suite)+
              ["else" ":" suite]
              ["finally" ":" suite]
try2_stmt ::= "try" ":" suite
              "finally" ":" suite
```

*except* *try* *try* *except* *except*  
          *except*  
  
          *except* 1  
  
          *except* *try*  
          *except* *except* *as* *except* *except* *try*  
          *as* *except*

```
except E as N:
    foo
```

```
except E as N:
    try:
        foo
    finally:
        del N
```

*except*  
  
*except* *sys* *sys.exc\_info()* *sys.exc\_info()* 3  
      *sys.exc\_info()*  
  
      *try* *return, continue break* *else* *else* *except*  
      *finally* ‘ ’ *try* *except else* *finally* *finally*  
*finally* *finally* *return break* :

```
>>> def f():
...     try:
...         1/0
```

1 *finally*

( )

```
...     finally:
...         return 42
...
>>> f()
42
```

*finally*
*return, break continue try...finally try finally ' ' continue finally*
*return finally finally return :*

```
>>> def foo():
...     try:
...         return 'try'
...     finally:
...         return 'finally'
...
>>> foo()
'finally'
```

*raise raise*

## 8.5 with

*with ( with ) try...except...finally*

```
with_stmt ::= "with" with_item ("," with_item)* ":" suite
with_item ::= expression ["as" target]
```

“ ” *with* :

1. ( *with\_item* )

2. *\_\_exit\_\_*()

3. *\_\_enter\_\_*()

4. *with* *\_\_enter\_\_*()

---

<i>: with</i>	<i>__enter__</i> ()	<i>__exit__</i> ()	6
---------------	---------------------	--------------------	---

---

5.

6. *\_\_exit\_\_*() *\_\_exit\_\_*() None

*\_\_exit\_\_*() *with*
*\_\_exit\_\_*()

*with* :

```
with A() as a:
    with B() as b:
        suite
```

$$\vdots$$

## 8.6

$$\vdots$$

funcdef	::=	[ <i>decorators</i> ] "def" <i>funcname</i> "(" [ <i>parameter_list</i> ] ")" ["->" <i>expression</i> ] ":" <i>suite</i>
decorators	::=	<i>decorator</i> +
decorator	::=	"@" <i>dotted_name</i> "(" (" [ <i>argument_list</i> [","] ])" NEWLINE
dotted_name	::=	<i>identifier</i> ("." <i>identifier</i> )*
parameter_list	::=	<i>defparameter</i> ("," <i>defparameter</i> )* [",," [ <i>parameter_list_starargs</i> ]   <i>parameter_list_starargs</i>
parameter_list_starargs	::=	"*" [ <i>parameter</i> ] ("," <i>defparameter</i> )* [",," ["**" <i>parameter</i> [","]]]   "**" <i>parameter</i> [",,"]
parameter	::=	<i>identifier</i> [":" <i>expression</i> ]
defparameter	::=	<i>parameter</i> ["=" <i>expression</i> ]
funcname	::=	<i>identifier</i>

2

```
@f1(arg)
@f2
def func(): pass
```

```
def func(): pass
func = f1(arg)(f2(func))
```

= “ ”

”\*”

---

2

---

<code>__doc__</code>	<i>docstring</i>
----------------------	------------------

None : “ ”

```
def whats_on_the_telly(penguin=None):
    if penguin is None:
        penguin = []
    penguin.append("property of the zoo")
    return penguin
```

“\*identifier” “\*\*identifier”  
 “\*” “\*identifier”  
 “: expression” \*identifier \*\*identifier “ ” “->  
 expression” Python \_\_annotations\_\_ annotations im-  
 port from \_\_future\_\_  
 “def” lambda *lambda* lambda “def” lambda  
 : “def” def  
 :

PEP 3107 -

PEP 484 -

PEP 526 -

PEP 563 -

## 8.7

( ):

```
classdef ::= [decorators] "class" classname [inheritance] ":" suite
inheritance ::= "(" [argument_list] ")"
classname ::= identifier
```

( ) object :

```
class Foo:
    pass
```

```
class Foo(object):
    pass
```

( )

3

\_\_dict\_\_

3

\_\_doc\_\_ *docstring*

:

```
@f1(arg)
@f2
class Foo: pass
```

```
class Foo: pass
Foo = f1(arg)(f2(Foo))
```

```
:                                self.name = value                                "self.name"
```

:

PEP 3115 - Python 3000

PEP 3129 - PEP 318

## 8.8

3.5 .

### 8.8.1

```
async_funcdef ::= [decorators] "async" "def" funcname "(" [parameter_list] ")"
                ["->" expression] ":" suite
```

Python ( *coroutine*) await async await async for async with

async def await async

yield from SyntaxError

:

```
async def func(param1, param2):
    do_stuff()
    await some_coroutine()
```

### 8.8.2 async for

```
async_for_stmt ::= "async" for_stmt
```

*asynchronous iterable* *iter* *asynchronous iterator* *next*

async for

:

```
async for TARGET in ITER:
    BLOCK
else:
    BLOCK2
```

:

```
iter = (ITER)
iter = type(iter).__aiter__(iter)
running = True
while running:
    try:
        TARGET = await type(iter).__anext__(iter)
    except StopAsyncIteration:
        running = False
    else:
        BLOCK
else:
    BLOCK2
```

`__aiter__()` `__anext__()`  
async for     SyntaxError

### 8.8.3 async with

async\_with\_stmt ::= "async" with\_stmt  
*asynchronous context manager*   *context manager*   *enter*   *exit*  
:

```
async with EXPR as VAR:
    BLOCK
```

:

```
mgr = (EXPR)
aexit = type(mgr).__aexit__
aenter = type(mgr).__aenter__(mgr)

VAR = await aenter
try:
    BLOCK
except:
    if not await aexit(mgr, *sys.exc_info()):
        raise
else:
    await aexit(mgr, None, None, None)
```

`__aenter__()` `__aexit__()`  
async with     SyntaxError

:



PEP 492 - `async` `await` Python



Python

## 9.1 Python

```

None) __main__
    Python
        Python
            sys ( ), builtins (
                __main__
                    -c
                        tty

```

## 9.2

```

:

file_input ::= (NEWLINE | statement)*
:
• Python
•
• exec()

```

## 9.3

:

`interactive_input ::= [stmt_list] NEWLINE | compound_stmt NEWLINE`

## 9.4

`eval()` `eval()` :

`eval_input ::= expression_list NEWLINE*`

Python Python

```
# Grammar for Python

# NOTE WELL: You should also follow all the steps listed at
# https://devguide.python.org/grammar/

# Start symbols for the grammar:
#     single_input is a single interactive statement;
#     file_input is a module or sequence of commands read from an input file;
#     eval_input is the input for the eval() functions.
# NB: compound_stmt in single_input is followed by extra NEWLINE!
single_input: NEWLINE | simple_stmt | compound_stmt NEWLINE
file_input: (NEWLINE | stmt)* ENDMARKER
eval_input: testlist NEWLINE* ENDMARKER

decorator: '@' dotted_name [ '(' [arglist] ')' ] NEWLINE
decorators: decorator+
decorated: decorators (classdef | funcdef | async_funcdef)

async_funcdef: 'async' funcdef
funcdef: 'def' NAME parameters ['>' test] ':' suite

parameters: '(' [typedarglist] ')'
typedarglist: (tfpdef ['=' test] (',' tfpdef ['=' test])* [',' [
    '*' [tfpdef] (',' tfpdef ['=' test])* [',' ['**' tfpdef ['']]
    | '**' tfpdef ['']]
    | '*' [tfpdef] (',' tfpdef ['=' test])* [',' ['**' tfpdef ['']]
    | '**' tfpdef [''])
tfpdef: NAME [':' test]
vararglist: (vfpdef ['=' test] (',' vfpdef ['=' test])* [',' [
    '*' [vfpdef] (',' vfpdef ['=' test])* [',' ['**' vfpdef ['']]
```

( )

( )

```

    | '**' vfpdef [' ','']]
| '*' [vfpdef] (',' vfpdef ['=' test])* [',' '**' vfpdef [' ','']]
| '**' vfpdef [' ','']
)
vfpdef: NAME

stmt: simple_stmt | compound_stmt
simple_stmt: small_stmt ';' small_stmt)* [';'] NEWLINE
small_stmt: (expr_stmt | del_stmt | pass_stmt | flow_stmt |
             import_stmt | global_stmt | nonlocal_stmt | assert_stmt)
expr_stmt: testlist_star_expr (annassign | augassign (yield_expr|testlist) |
                              ('=' (yield_expr|testlist_star_expr))* )
annassign: ':' test ['=' test]
testlist_star_expr: (test|star_expr) (',' (test|star_expr))* [';']
augassign: ('+=' | '-=' | '*=' | '@=' | '/=' | '%=' | '&=' | '|=' | '^=' |
            '<=' | '>=' | '**=' | '//=')
# For normal and annotated assignments, additional restrictions enforced by the
↪ interpreter
del_stmt: 'del' exprlist
pass_stmt: 'pass'
flow_stmt: break_stmt | continue_stmt | return_stmt | raise_stmt | yield_stmt
break_stmt: 'break'
continue_stmt: 'continue'
return_stmt: 'return' [testlist]
yield_stmt: yield_expr
raise_stmt: 'raise' [test ['from' test]]
import_stmt: import_name | import_from
import_name: 'import' dotted_as_names
# note below: the ('.' | '...') is necessary because '...' is tokenized as ELLIPSIS
import_from: ('from' (('.' | '...')* dotted_name | ('.' | '...')+
                  'import' ('*' | '(' import_as_names ')' | import_as_names))
import_as_name: NAME ['as' NAME]
dotted_as_name: dotted_name ['as' NAME]
import_as_names: import_as_name (',' import_as_name)* [';']
dotted_as_names: dotted_as_name (',' dotted_as_name)*
dotted_name: NAME ('.' NAME)*
global_stmt: 'global' NAME (',' NAME)*
nonlocal_stmt: 'nonlocal' NAME (',' NAME)*
assert_stmt: 'assert' test [',' test]

compound_stmt: if_stmt | while_stmt | for_stmt | try_stmt | with_stmt | funcdef |
↪ classdef | decorated | async_stmt
async_stmt: 'async' (funcdef | with_stmt | for_stmt)
if_stmt: 'if' test ':' suite ('elif' test ':' suite)* ['else' ':' suite]
while_stmt: 'while' test ':' suite ['else' ':' suite]
for_stmt: 'for' exprlist 'in' testlist ':' suite ['else' ':' suite]
try_stmt: ('try' ':' suite
          ((except_clause ':' suite)+
           ['else' ':' suite]
           ['finally' ':' suite] |
           'finally' ':' suite))
with_stmt: 'with' with_item (',' with_item)* ':' suite

```

( )

( )

```

with_item: test ['as' expr]
# NB compile.c makes sure that the default except clause is last
except_clause: 'except' [test ['as' NAME]]
suite: simple_stmt | NEWLINE INDENT stmt+ DEDENT

test: or_test ['if' or_test 'else' test] | lambdef
test_nocond: or_test | lambdef_nocond
lambdef: 'lambda' [varargslist] ':' test
lambdef_nocond: 'lambda' [varargslist] ':' test_nocond
or_test: and_test ('or' and_test)*
and_test: not_test ('and' not_test)*
not_test: 'not' not_test | comparison
comparison: expr (comp_op expr)*
# <> isn't actually a valid comparison operator in Python. It's here for the
# sake of a __future__ import described in PEP 401 (which really works :-)
comp_op: '<' | '>' | '==' | '>=' | '<=' | '<>' | '!=' | 'in' | 'not in' | 'is' | 'is not'
star_expr: '*' expr
expr: xor_expr ('|' xor_expr)*
xor_expr: and_expr ('^' and_expr)*
and_expr: shift_expr ('&' shift_expr)*
shift_expr: arith_expr (('<<' | '>>') arith_expr)*
arith_expr: term (('+' | '-') term)*
term: factor (('*' | '@' | '/' | '%' | '//') factor)*
factor: ('+' | '-' | '~') factor | power
power: atom_expr ['**' factor]
atom_expr: ['await'] atom trailer*
atom: '(' [yield_expr|testlist_comp] ')' |
      '[' [testlist_comp] ']' |
      '{' [dictorsetmaker] '}' |
      NAME | NUMBER | STRING+ | '...' | 'None' | 'True' | 'False'
testlist_comp: (test|star_expr) (comp_for | (',' (test|star_expr))* [','])
trailer: '(' [arglist] ')' | '[' subscriptlist ']' | '.' NAME
subscriptlist: subscript (',' subscript)* [',']
subscript: test | [test] ':' [test] [sliceop]
sliceop: ':' [test]
exprlist: (expr|star_expr) (',' (expr|star_expr))* [',']
testlist: test (',' test)* [',']
dictorsetmaker: ( ((test ':' test | '**' expr)
                  (comp_for | (',' (test ':' test | '**' expr))* [','])) |
                ((test | star_expr)
                  (comp_for | (',' (test | star_expr))* [','])) )

classdef: 'class' NAME ['(' [arglist] ')'] ':' suite

arglist: argument (',' argument)* [',']

# The reason that keywords are test nodes instead of NAME is that using NAME
# results in an ambiguity. ast.c makes sure it's a NAME.
# "test '=' test" is really "keyword '=' test", but we have no such token.
# These need to be in a single rule to avoid grammar that is ambiguous
# to our LL(1) parser. Even though 'test' includes 'expr' in star_expr,
# we explicitly match '*' here, too, to give it proper precedence.

```

( )

( )

```
# Illegal combinations and orderings are blocked in ast.c:
# multiple (test comp_for) arguments are blocked; keyword unpackings
# that precede iterable unpackings are blocked; etc.
argument: ( test [comp_for] |
            test '=' test |
            '**' test |
            '*' test )

comp_iter: comp_for | comp_if
sync_comp_for: 'for' exprlist 'in' or_test [comp_iter]
comp_for: ['async'] sync_comp_for
comp_if: 'if' test_nocond [comp_iter]

# not used in grammar, but may appear in "node" passed from Parser to Compiler
encoding_decl: NAME

yield_expr: 'yield' [yield_arg]
yield_arg: 'from' test | testlist
```



---

---

>>> 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))`

*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-*

**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

**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- ” **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 annotation** – *annotation*

`int` `int` :

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

*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

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 / *in-*  
*teractive*

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`

**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__` `__getattr__()`

**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` **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 “ ” \_\_\_\_\_

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

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

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

virtual environment –	Python	Python	Python
venv			
virtual machine –	Python	<i>bytecode</i>	
Zen of Python – Python	Python		"import this"



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	-		



---

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's 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](mailto: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 routines 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 specific 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  
jloup@gzip.org

Mark Adler  
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

Copyright © 2001-2019 Python Software Foundation. All rights reserved.

Copyright © 2000 BeOpen.com. All rights reserved.

Copyright © 1995-2000 Corporation for National Research Initiatives. All rights reserved.

Copyright © 1991-1995 Stichting Mathematisch Centrum. All rights reserved.

---

*History and License*





---

## Non-alphabetical

`...`, 91  
    ellipsis literal, 16  
`'''`  
    string literal, 9  
`.` (*dot*)  
    attribute reference, 55  
    in numeric literal, 12  
`!` (*exclamation*)  
    in formatted string literal, 10  
`-` (*minus*)  
    binary operator, 58  
    unary operator, 57  
`'` (*single quote*)  
    string literal, 8  
`"` (*double quote*)  
    string literal, 8  
`"""`  
    string literal, 9  
`#` (*hash*)  
    comment, 5  
    source encoding declaration, 5  
`%` (*percent*)  
    , 58  
`%=`  
    augmented assignment, 67  
`&` (*ampersand*)  
    , 58  
`&=`  
    augmented assignment, 67  
`()` (*parentheses*)  
    call, 55  
    class definition, 80  
    function definition, 79  
    generator expression, 52  
    in assignment target list, 66  
    tuple display, 50  
`*` (*asterisk*)  
    function definition, 80  
    import statement, 72  
    in assignment target list, 66  
    in expression lists, 62  
    in function calls, 56  
    , 58  
`**`  
    function definition, 80  
    in dictionary displays, 51  
    in function calls, 56  
    , 57  
`**=`  
    augmented assignment, 67  
`*=`  
    augmented assignment, 67  
`+` (*plus*)  
    binary operator, 58  
    unary operator, 57  
`+=`  
    augmented assignment, 67  
`,` (*comma*)  
    argument list, 55  
    expression list, 51, 62, 68, 80  
    identifier list, 73  
    import statement, 71  
    in dictionary displays, 51  
    in target list, 66  
    parameter list, 79  
    slicing, 55  
    tuple display, 50  
    with statement, 78  
`/` (*slash*)  
    , 58  
`//`  
    , 58  
`//=`  
    augmented assignment, 67  
`/=`  
    augmented assignment, 67  
`0b`  
    integer literal, 12

0o	integer literal, 12	escape sequence, 9
0x	integer literal, 12	\a
2to3, 91		escape sequence, 9
:	(colon)	\b
	annotated variable, 67	escape sequence, 9
	compound statement, 7680	\f
	function annotations, 80	escape sequence, 9
	in dictionary expressions, 51	\N
	in formatted string literal, 10	escape sequence, 9
	lambda expression, 61	\n
	slicing, 55	escape sequence, 9
;	(semicolon), 75	\r
<	(less)	escape sequence, 9
	, 59	\t
<<		escape sequence, 9
	, 58	>>>, 91
<<=	augmented assignment, 67	\U
<=		escape sequence, 9
	, 59	\u
!=		escape sequence, 9
	, 59	\v
--	augmented assignment, 67	escape sequence, 9
=	(equals)	\x
	assignment statement, 66	escape sequence, 9
	class definition, 25	^ (caret)
	function definition, 79	, 58
	in function calls, 55	^=
==		augmented assignment, 67
	, 59	_ (underscore)
->	function annotations, 80	in numeric literal, 12
>	(greater)	_, identifiers, 8
	, 59	__, identifiers, 8
>=		__abs__() (object ), 29
	, 59	__add__() (object ), 28
>>		__aenter__() (object ), 33
	, 58	__aexit__() (object ), 33
>>=	augmented assignment, 67	__aiter__() (object ), 32
@ (at)	class definition, 80	__all__ (optional module attribute), 72
	function definition, 79	__and__() (object ), 28
	, 58	__anext__() (agen ), 54
[] (square brackets)	in assignment target list, 66	__anext__() (object ), 32
	list expression, 51	__annotations__ (class attribute), 18
	subscription, 55	__annotations__ (function attribute), 17
\ (backslash)	escape sequence, 9	__annotations__ (module attribute), 18
\\		__await__() (object ), 32
		__bases__ (class attribute), 18
		__bool__() (object method), 27
		__bool__() (object ), 22
		__bytes__() (object ), 21
		__cached__, 44
		__call__() (object method), 57
		__call__() (object ), 27
		__cause__ (exception attribute), 69
		__ceil__() (object ), 30

---

```

__class__ (instance attribute), 18
__class__ (method cell), 26
__class__ (module attribute), 23
__class_getitem__() (object ), 27
__classcell__ (class namespace entry), 26
__closure__ (function attribute), 17
__code__ (function attribute), 17
__complex__() (object ), 30
__contains__() (object ), 28
__context__ (exception attribute), 69
__debug__, 68
__defaults__ (function attribute), 17
__del__() (object ), 20
__delattr__() (object ), 22
__delete__() (object ), 23
__delitem__() (object ), 28
__dict__ (class attribute), 18
__dict__ (function attribute), 17
__dict__ (instance attribute), 18
__dict__ (module attribute), 18
__dir__ (module attribute), 23
__dir__() (object ), 22
__divmod__() (object ), 28
__doc__ (class attribute), 18
__doc__ (function attribute), 17
__doc__ (method attribute), 17
__doc__ (module attribute), 18
__enter__() (object ), 30
__eq__() (object ), 21
__exit__() (object ), 30
__file__, 44
__file__ (module attribute), 18
__float__() (object ), 30
__floor__() (object ), 30
__floordiv__() (object ), 28
__format__() (object ), 21
__func__ (method attribute), 17
__future__, 94
    future statement, 72
__ge__() (object ), 21
__get__() (object ), 23
__getattr__ (module attribute), 23
__getattr__() (object ), 22
__getattribute__() (object ), 22
__getitem__() (mapping object method), 20
__getitem__() (object ), 28
__globals__ (function attribute), 17
__gt__() (object ), 21
__hash__() (object ), 21
__iadd__() (object ), 29
__iand__() (object ), 29
__ifloordiv__() (object ), 29
__ilshift__() (object ), 29
__imatmul__() (object ), 29
__imod__() (object ), 29
__imul__() (object ), 29
__index__() (object ), 30
__init__() (object ), 20
__init_subclass__() (object ), 24
__instancecheck__() (class ), 27
__int__() (object ), 30
__invert__() (object ), 29
__ior__() (object ), 29
__ipow__() (object ), 29
__irshift__() (object ), 29
__isub__() (object ), 29
__iter__() (object ), 28
__itruediv__() (object ), 29
__ixor__() (object ), 29
__kwdefaults__ (function attribute), 17
__le__() (object ), 21
__len__() (mapping object method), 22
__len__() (object ), 27
__length_hint__() (object ), 27
__loader__, 44
__lshift__() (object ), 28
__lt__() (object ), 21
__main__, 36, 85
__matmul__() (object ), 28
__missing__() (object ), 28
__mod__() (object ), 28
__module__ (class attribute), 18
__module__ (function attribute), 17
__module__ (method attribute), 17
__mul__() (object ), 28
__name__, 43
__name__ (class attribute), 18
__name__ (function attribute), 17
__name__ (method attribute), 17
__name__ (module attribute), 18
__ne__() (object ), 21
__neg__() (object ), 29
__new__() (object ), 20
__next__() (generator ), 53
__or__() (object ), 28
__package__, 44
__path__, 44
__pos__() (object ), 29
__pow__() (object ), 28
__prepare__ (metaclass method), 26
__radd__() (object ), 29
__rand__() (object ), 29
__rdivmod__() (object ), 29
__repr__() (object ), 20
__reversed__() (object ), 28
__rfloordiv__() (object ), 29
__rlshift__() (object ), 29

```

---

`--matmul__()` (*object* ), 29  
`--mod__()` (*object* ), 29  
`--mul__()` (*object* ), 29  
`--ror__()` (*object* ), 29  
`--round__()` (*object* ), 30  
`--rpow__()` (*object* ), 29  
`--rrshift__()` (*object* ), 29  
`--rshift__()` (*object* ), 28  
`--rsub__()` (*object* ), 29  
`--rtruediv__()` (*object* ), 29  
`--rxor__()` (*object* ), 29  
`--self__` (*method attribute*), 17  
`--set__()` (*object* ), 23  
`--set_name__()` (*object* ), 23  
`--setattr__()` (*object* ), 22  
`--setitem__()` (*object* ), 28  
`--slots__`, 97  
`--spec__`, 44  
`--str__()` (*object* ), 21  
`--sub__()` (*object* ), 28  
`--subclasscheck__()` (*class* ), 27  
`--traceback__` (*exception attribute*), 69  
`--truediv__()` (*object* ), 28  
`--trunc__()` (*object* ), 30  
`--xor__()` (*object* ), 28  
`{}` (*curly brackets*)  
    dictionary expression, 51  
    in formatted string literal, 10  
    set expression, 51  
`|` (*vertical bar*)  
    , 58  
`|=`  
    augmented assignment, 67  
`~` (*tilde*)  
    , 57  
  
    AssertionError, 68  
    AttributeError, 55  
    GeneratorExit, 53, 54  
    ImportError, 71  
    NameError, 50  
    StopAsyncIteration, 54  
    StopIteration, 53, 69  
    TypeError, 57  
    ValueError, 58  
    ZeroDivisionError, 58  
  
    PYTHONHASHSEED, 22  
  
    assert, 68  
    async def, 81  
    async for, 81  
    async with, 82  
    break, 70, 7678  
  
    class, 80  
    continue, 71, 7678  
    def, 79  
    del, 20, 68  
    for, 70, 71, 76  
    global, 68, 73  
    if, 76  
    import, 18, 71  
    nonlocal, 73  
    pass, 68  
    raise, 69  
    return, 68, 77, 78  
    try, 19, 77  
    while, 70, 71, 76  
    with, 30, 78  
    yield, 69  
  
    % (*percent*), 58  
    & (*ampersand*), 58  
    \* (*asterisk*), 58  
    \*\*, 57  
    / (*slash*), 58  
    //, 58  
    < (*less*), 59  
    <<, 58  
    <=, 59  
    !=, 59  
    ==, 59  
    > (*greater*), 59  
    >=, 59  
    >>, 58  
    @ (*at*), 58  
    ^ (*caret*), 58  
    | (*vertical bar*), 58  
    ~ (*tilde*), 57  
    and, 61  
    in, 61  
    is, 61  
    is not, 61  
    not, 61  
    not in, 61  
    or, 61  
  
A  
abs  
    , 30  
abstract base class -- , 91  
aclose() (*agen* ), 54  
addition, 58  
and  
    bitwise, 58  
    , 61  
annotated  
    assignment, 67

---

- annotation -- , 91
- annotations
  - function, 80
- anonymous
  - function, 61
- argument
  - call semantics, 55
  - function, 17
  - function definition, 79
- argument -- , 91
- arithmetic
  - conversion, 49
  - operation, binary, 57
  - operation, unary, 57
- array
  - , 16
- as
  - except clause, 77
  - import statement, 71
    - , 71, 77, 78
  - with statement, 78
- ASCII, 4, 8
- asend() (*agen* ), 54
- assert
  - , 68
- AssertionError
  - , 68
- assertions
  - debugging, 68
- assignment
  - annotated, 67
  - attribute, 66
  - augmented, 67
  - class attribute, 18
  - class instance attribute, 18
  - slicing, 66
  - statement, 16, 66
  - subscription, 66
  - target list, 66
- async
  - , 81
- async def
  - , 81
- async for
  - in comprehensions, 50
  - , 81
- async with
  - , 82
- asynchronous context manager -- , 91
- asynchronous generator
  - asynchronous iterator, 18
  - function, 18
- asynchronous generator -- , 92
- asynchronous generator iterator -- , 92
- asynchronous iterable -- , 92
- asynchronous iterator -- , 92
- asynchronous-generator
  - , 54
- athrow() (*agen* ), 54
- atom, 49
- attribute, 15
  - assignment, 66
  - assignment, class, 18
  - assignment, class instance, 18
  - class, 18
  - class instance, 18
  - deletion, 68
  - generic special, 15
  - reference, 55
  - special, 15
- attribute -- , 92
- AttributeError
  - , 55
- augmented
  - assignment, 67
- await
  - in comprehensions, 51
  - , 57, 81
- awaitable -- , 92

## B

- b'
  - bytes literal, 9
- b"
  - bytes literal, 9
- backslash character, 6
- BDFL, 92
- binary
  - arithmetic operation, 57
  - bitwise operation, 58
- binary file -- , 92
- binary literal, 12
- binding
  - global name, 73
  - name, 35, 66, 71, 79, 80
- bitwise
  - and, 58
  - operation, binary, 58
  - operation, unary, 57
  - or, 58
  - xor, 58
- blank line, 6
- block, 35
  - code, 35
- BNF, 3, 49
- Boolean
  - operation, 61
  - , 16

---

- break
  - , 70, 7678
- built-in
  - method, 18
- built-in function
  - call, 57
  - , 18, 57
- built-in method
  - call, 57
  - , 18, 57
- builtins
  - , 85
- byte, 16
- bytearray, 16
- bytecode, 18
- bytecode -- , 92
- bytes, 16
  - , 21
- bytes literal, 8
- bytes-like object -- , 92
- C
- C, 9
  - language, 15, 16, 18, 59
- call, 55
  - built-in function, 57
  - built-in method, 57
  - class instance, 57
  - class object, 18, 57
  - function, 17, 57
  - instance, 27, 57
  - method, 57
  - procedure, 65
  - user-defined function, 57
- callable
  - , 17, 55
- C-contiguous, 92
- chaining
  - comparisons, 59
  - exception, 69
- character, 16, 55
- chr
  - , 16
- class
  - attribute, 18
  - attribute assignment, 18
  - body, 26
  - constructor, 20
  - definition, 68, 80
  - instance, 18
  - name, 80
    - , 18, 57, 80
    - , 80
- class -- , 92
- class instance
  - attribute, 18
  - attribute assignment, 18
  - call, 57
  - , 18, 57
- class object
  - call, 18, 57
- class variable -- , 92
- clause, 75
- clear() (*frame* ), 19
- close() (*coroutine* ), 32
- close() (*generator* ), 53
- co\_argcount (*code object attribute*), 18
- co\_cellvars (*code object attribute*), 18
- co\_code (*code object attribute*), 18
- co\_consts (*code object attribute*), 18
- co\_filename (*code object attribute*), 18
- co\_firstlineno (*code object attribute*), 18
- co\_flags (*code object attribute*), 18
- co\_freevars (*code object attribute*), 18
- co\_lnotab (*code object attribute*), 18
- co\_name (*code object attribute*), 18
- co\_names (*code object attribute*), 18
- co\_nlocals (*code object attribute*), 18
- co\_stacksize (*code object attribute*), 18
- co\_varnames (*code object attribute*), 18
- code
  - block, 35
- code object, 18
- coercion -- , 92
- comma
  - trailing, 62
  - tuple display, 50
- command line, 85
- comment, 5
- comparison, 59
- comparisons, 21
  - chaining, 59
- compile
  - , 73
- complex
  - number, 16
    - , 30
    - , 16
- complex literal, 12
- complex number -- , 92
- compound
  - statement, 75
- comprehensions
  - list, 51
- Conditional
  - expression, 61
- conditional
  - expression, 61

constant, 8  
 constructor  
   class, 20  
 container, 15, 18  
 context manager, 30  
 context manager -- , 92  
 contiguous -- , 92  
 continue  
   , 71, 7678  
 conversion  
   arithmetic, 49  
   string, 21, 65  
 coroutine, 31, 52  
   function, 18  
 coroutine -- , 92  
 coroutine function -- , 92  
 CPython, 92

## D

dangling  
   else, 75  
 data, 15  
   type, 15  
   type, immutable, 50  
 datum, 51  
 dbm.gnu  
   , 17  
 dbm.ndbm  
   , 17  
 debugging  
   assertions, 68  
 decimal literal, 12  
 decorator -- , 92  
 DEDENT token, 6, 75  
 def  
   , 79  
 default  
   parameter value, 79  
 definition  
   class, 68, 80  
   function, 68, 79  
 del  
   , 20, 68  
 deletion  
   attribute, 68  
   target, 68  
   target list, 68  
 delimiters, 13  
 descriptor -- , 93  
 destructor, 20, 66  
 dictionary  
   display, 51  
   , 17, 18, 21, 51, 55, 66  
 dictionary -- , 93

dictionary view -- , 93  
 display  
   dictionary, 51  
   list, 51  
   set, 51  
   tuple, 50  
 division, 58  
 divmod  
   , 29  
 docstring, 80  
 docstring -- , 93  
 documentation string, 19  
 duck-typing -- , 93

## E

e  
   in numeric literal, 12  
 EAFP, 93  
 elif  
   , 76  
 Ellipsis  
   , 16  
 else  
   conditional expression, 61  
   dangling, 75  
   , 70, 76, 77  
 empty  
   list, 51  
   tuple, 16, 50  
 encoding declarations (*source file*), 5  
 environment, 35  
 error handling, 36  
 errors, 36  
 escape sequence, 9  
 eval  
   , 73, 86  
 evaluation  
   order, 62  
 exc\_info (*in module sys*), 19  
 except  
   , 77  
 exception, 36, 69  
   chaining, 69  
   handler, 19  
   raising, 69  
 exception handler, 36  
 exclusive  
   or, 58  
 exec  
   , 73  
 execution  
   frame, 35, 80  
   restricted, 36  
   stack, 19

- execution model, 35
- expression, 49
  - Conditional, 61
  - conditional, 61
  - generator, 52
  - lambda, 61, 80
  - list, 62, 65
  - statement, 65
  - yield, 52
- expression -- , 93
- extension
  - module, 15
- extension module -- , 93
- F**
- f'
  - formatted string literal, 9
- f"
  - formatted string literal, 9
- f\_back (*frame attribute*), 19
- f\_builtins (*frame attribute*), 19
- f\_code (*frame attribute*), 19
- f\_globals (*frame attribute*), 19
- f\_lasti (*frame attribute*), 19
- f\_lineno (*frame attribute*), 19
- f\_locals (*frame attribute*), 19
- f\_trace (*frame attribute*), 19
- f\_trace\_lines (*frame attribute*), 19
- f\_trace\_opcodes (*frame attribute*), 19
- False, 16
- file object -- , 93
- file-like object -- , 93
- finalizer, 20
- finally
  - , 6971, 77
- find\_spec
  - finder, 41
- finder, 41
  - find\_spec, 41
- finder -- , 93
- float
  - , 30
- floating point
  - number, 16
  - , 16
- floating point literal, 12
- floor division -- , 93
- for
  - in comprehensions, 50
  - , 70, 71, 76
- form
  - lambda, 61
- format() (*built-in function*)
- \_\_str\_\_() (*object method*), 21

- formatted string literal, 10
- Fortran contiguous, 92
- frame
  - execution, 35, 80
  - , 19
- free
  - variable, 35
- from
  - import statement, 35, 71
  - , 52, 71
  - yield from expression, 52
- frozenset
  - , 16
- f-string, 10
- f-string -- f- , 93
- function
  - annotations, 80
  - anonymous, 61
  - argument, 17
  - call, 17, 57
  - call, user-defined, 57
  - definition, 68, 79
  - generator, 52, 69
  - name, 79
  - user-defined, 17
  - , 17, 18, 57, 79
- function -- , 93
- function annotation -- , 93
- future
  - statement, 72
- G**
- garbage collection, 15
- garbage collection -- , 94
- generator, 94
  - expression, 52
  - function, 17, 52, 69
  - iterator, 17, 69
  - , 19, 52, 53
- generator -- , 94
- generator expression, 94
- generator expression -- , 94
- generator iterator -- , 94
- GeneratorExit
  - , 53, 54
- generic
  - special attribute, 15
- generic function -- , 94
- GIL, 94
- global
  - name binding, 73
  - namespace, 17
  - , 68, 73
- global interpreter lock -- , 94



grammar, 3  
grouping, 6

## H

handle an exception, 36  
handler  
    exception, 19  
hash  
    , 21  
hash character, 5  
hashable, 51  
hashable -- , 94  
hash-based pyc -- pyc, 94  
hexadecimal literal, 12  
hierarchy  
    type, 15  
hooks  
    import, 41  
    meta, 41  
    path, 41  
  
I  
id  
    , 15  
identifier, 7, 50  
identity  
    test, 61  
identity of an object, 15  
IDLE, 94  
if  
    conditional expression, 61  
    in comprehensions, 50  
    , 76  
imaginary literal, 12  
immutable  
    data type, 50  
    object, 50, 51  
    , 16  
immutable -- , 94  
immutable object, 15  
immutable sequence  
    , 16  
immutable types  
    subclassing, 20  
import  
    hooks, 41  
    , 18, 71  
import hooks, 41  
import machinery, 39  
import path -- , 94  
importer -- , 94  
ImportError  
    , 71  
importing -- , 94

in  
    , 76  
    , 61  
inclusive  
    or, 58  
INDENT token, 6  
indentation, 6  
index operation, 16  
indices() (*slice* ), 19  
inheritance, 80  
input, 86  
instance  
    call, 27, 57  
    class, 18  
    , 18, 57  
int  
    , 30  
integer, 16  
    representation, 16  
    , 16  
integer literal, 12  
interactive -- , 94  
interactive mode, 85  
internal type, 18  
interpolated string literal, 10  
interpreted -- , 94  
interpreter, 85  
interpreter shutdown -- , 94  
inversion, 57  
invocation, 17  
io  
    , 18  
is  
    , 61  
is not  
    , 61  
item  
    sequence, 55  
    string, 55  
item selection, 16  
iterable  
    unpacking, 62  
iterable -- , 95  
iterator -- , 95

## J

j  
    in numeric literal, 13  
Java  
    language, 16

## K

key, 51  
key function -- , 95

key/datum pair, 51  
keyword, 8  
keyword argument -- , 95

## L

lambda, 95  
    expression, 61, 80  
    form, 61  
language  
    C, 15, 16, 18, 59  
    Java, 16  
last\_traceback (*in module sys*), 19  
LBYL, 95  
leading whitespace, 6  
len  
    , 16, 17, 27  
lexical analysis, 5  
lexical definitions, 4  
line continuation, 6  
line joining, 5, 6  
line structure, 5  
list  
    assignment, target, 66  
    comprehensions, 51  
    deletion target, 68  
    display, 51  
    empty, 51  
    expression, 62, 65  
    target, 66, 76  
    , 16, 51, 55, 66  
list -- , 95  
list comprehension -- , 95  
literal, 8, 50  
loader, 41  
loader -- , 95  
logical line, 5  
loop  
    over mutable sequence, 76  
    statement, 70, 71, 76  
loop control  
    target, 70

## M

makefile() (*socket method*), 18  
mangling  
    name, 50  
mapping  
    , 17, 18, 55, 66  
mapping -- , 95  
matrix multiplication, 58  
membership  
    test, 61  
meta  
    hooks, 41

meta hooks, 41  
meta path finder -- , 95  
metaclass, 25  
metaclass -- , 95  
metaclass hint, 26  
method  
    built-in, 18  
    call, 57  
    user-defined, 17  
    , 17, 18, 57  
method -- , 95  
method resolution order -- , 95  
minus, 57  
module  
    extension, 15  
    importing, 71  
    namespace, 18  
    , 18, 55  
module -- , 95  
module spec, 41  
module spec -- , 95  
modulo, 58  
MRO, 95  
multiplication, 58  
mutable  
    , 16, 66  
mutable -- , 95  
mutable object, 15  
mutable sequence  
    loop over, 76  
    , 16

## N

name, 7, 35, 50  
    binding, 35, 66, 71, 79, 80  
    binding, global, 73  
    class, 80  
    function, 79  
    mangling, 50  
    rebinding, 66  
    unbinding, 68  
named tuple -- , 95  
NameError  
    , 50  
NameError (*built-in exception*), 35  
names  
    private, 50  
namespace, 35  
    global, 17  
    module, 18  
    package, 40  
namespace -- , 96  
namespace package -- , 96  
negation, 57

nested scope -- , 96  
 NEWLINE token, 5, 75  
 new-style class -- , 96  
 None  
     , 16, 65  
 nonlocal  
     , 73  
 not  
     , 61  
 not in  
     , 61  
 notation, 3  
 NotImplemented  
     , 16  
 null  
     operation, 68  
 number, 12  
     complex, 16  
     floating point, 16  
 numeric  
     , 16, 18  
 numeric literal, 12

## O

object, 15  
     code, 18  
     immutable, 50, 51  
 object -- , 96  
 object.\_\_slots\_\_ ( ), 24  
 octal literal, 12  
 open  
     , 18  
 operation  
     binary arithmetic, 57  
     binary bitwise, 58  
     Boolean, 61  
     null, 68  
     power, 57  
     shifting, 58  
     unary arithmetic, 57  
     unary bitwise, 57  
 operator  
     - (*minus*), 57, 58  
     + (*plus*), 57, 58  
     overloading, 20  
     precedence, 62  
     ternary, 61  
 operators, 13  
 or  
     bitwise, 58  
     exclusive, 58  
     inclusive, 58  
     , 61  
 ord  
     , 16  
 order  
     evaluation, 62  
 output, 65  
     standard, 65  
 overloading  
     operator, 20

## P

package, 39  
     namespace, 40  
     portion, 40  
     regular, 40  
 package -- , 96  
 parameter  
     call semantics, 56  
     function definition, 79  
     value, default, 79  
 parameter -- , 96  
 parenthesized form, 50  
 parser, 5  
 pass  
     , 68  
 path  
     hooks, 41  
 path based finder, 45  
 path based finder -- , 96  
 path entry -- , 96  
 path entry finder -- , 96  
 path entry hook -- , 96  
 path hooks, 41  
 path-like object -- , 96  
 PEP, 96  
 physical line, 5, 6, 9  
 plus, 57  
 popen() (*in module os*), 18  
 portion  
     package, 40  
 portion -- , 97  
 positional argument -- , 97  
 pow  
     , 29  
 power  
     operation, 57  
 precedence  
     operator, 62  
 primary, 54  
 print  
     , 21  
 print() (*built-in function*)  
     \_\_str\_\_() (*object method*), 21  
 private  
     names, 50  
 procedure

- call, 65
- program, 85
- provisional API -- API, 97
- provisional package -- , 97
- Python 3000, 97
- Python
  - PEP 1, 96
  - PEP 236, 72
  - PEP 238, 93
  - PEP 255, 53
  - PEP 278, 98
  - PEP 302, 39, 47, 93, 95
  - PEP 308, 61
  - PEP 318, 81
  - PEP 328, 47, 72
  - PEP 338, 47
  - PEP 342, 53
  - PEP 343, 30, 79, 92
  - PEP 362, 91, 96
  - PEP 366, 44, 47
  - PEP 380, 53
  - PEP 395, 47
  - PEP 411, 97
  - PEP 414, 9
  - PEP 420, 39, 40, 44, 47, 93, 96, 97
  - PEP 443, 94
  - PEP 448, 51, 56, 62
  - PEP 451, 47, 93
  - PEP 484, 27, 67, 80, 91, 93, 98, 99
  - PEP 492, 32, 53, 83, 91, 92
  - PEP 498, 12, 93
  - PEP 519, 96
  - PEP 525, 53, 92
  - PEP 526, 67, 80, 91, 99
  - PEP 530, 51
  - PEP 560, 25, 27
  - PEP 562, 23
  - PEP 563, 80
  - PEP 3104, 73
  - PEP 3107, 80
  - PEP 3115, 26, 81
  - PEP 3116, 98
  - PEP 3119, 27
  - PEP 3120, 5
  - PEP 3129, 81
  - PEP 3131, 7
  - PEP 3132, 67
  - PEP 3135, 26
  - PEP 3147, 44
  - PEP 3155, 97
- PYTHONHASHSEED, 22
- Pythonic, 97
- PYTHONPATH, 45

## Q

- qualified name -- , 97

## R

- r'
  - raw string literal, 9
- r"
  - raw string literal, 9
- raise
  - , 69
- raise an exception, 36
- raising
  - exception, 69
- range
  - , 76
- raw string, 9
- rebinding
  - name, 66
- reference
  - attribute, 55
- reference count -- , 97
- reference counting, 15
- regular
  - package, 40
- regular package -- , 97
- relative
  - import, 72
- repr
  - , 65
- repr() (*built-in function*)
- \_\_repr\_\_() (*object method*), 20
- representation
  - integer, 16
- reserved word, 8
- restricted
  - execution, 36
- return
  - , 68, 77, 78
- round
  - , 30

## S

- scope, 35
- send() (*coroutine* ), 32
- send() (*generator* ), 53
- sequence
  - item, 55
  - , 16, 18, 55, 61, 66, 76
- sequence -- , 97
- set
  - display, 51
  - , 16, 51
- set type

- , 16
- shifting
  - operation, 58
- simple
  - statement, 65
- single dispatch -- , 98
- singleton
  - tuple, 16
- slice, 55
  - , 19
  - , 28
- slice -- , 98
- slicing, 16, 55
  - assignment, 66
- source character set, 5
- space, 6
- special
  - attribute, 15
  - attribute, generic, 15
- special method -- , 98
- stack
  - execution, 19
  - trace, 19
- standard
  - output, 65
- Standard C, 9
- standard input, 85
- start (*slice object attribute*), 19, 55
- statement
  - assignment, 16, 66
  - assignment, annotated, 67
  - assignment, augmented, 67
  - compound, 75
  - expression, 65
  - future, 72
  - loop, 70, 71, 76
  - simple, 65
- statement -- , 98
- statement grouping, 6
- stderr (*in module sys*), 18
- stdin (*in module sys*), 18
- stdio, 18
- stdout (*in module sys*), 18
- step (*slice object attribute*), 19, 55
- stop (*slice object attribute*), 19, 55
- StopAsyncIteration
  - , 54
- StopIteration
  - , 53, 69
- string
  - \_\_format\_\_() (*object method*), 21
  - \_\_str\_\_() (*object method*), 21
  - conversion, 21, 65
  - formatted literal, 10

- immutable sequences, 16
- interpolated literal, 10
- item, 55
  - , 55
- string literal, 8
- struct sequence -- , 98
- subclassing
  - immutable types, 20
- subscription, 16, 17, 55
  - assignment, 66
- subtraction, 58
- suite, 75
- syntax, 3
- sys
  - , 77, 85
- sys.exc\_info, 19
- sys.last\_traceback, 19
- sys.meta\_path, 41
- sys.modules, 40
- sys.path, 45
- sys.path\_hooks, 45
- sys.path\_importer\_cache, 45
- sys.stderr, 18
- sys.stdin, 18
- sys.stdout, 18
- SystemExit (*built-in exception*), 36

## T

- tab, 6
- target, 66
  - deletion, 68
  - list, 66, 76
  - list assignment, 66
  - list, deletion, 68
  - loop control, 70
- tb\_frame (*traceback attribute*), 19
- tb\_lasti (*traceback attribute*), 19
- tb\_lineno (*traceback attribute*), 19
- tb\_next (*traceback attribute*), 19
- termination model, 36
- ternary
  - operator, 61
- test
  - identity, 61
  - membership, 61
- text encoding -- , 98
- text file -- , 98
- throw() (*coroutine* ), 32
- throw() (*generator* ), 53
- token, 5
- trace
  - stack, 19
- traceback
  - , 19, 69, 77

trailing  
    comma, 62  
triple-quoted string, 9  
triple-quoted string -- , 98  
True, 16  
try  
    , 19, 77  
tuple  
    display, 50  
    empty, 16, 50  
    singleton, 16  
    , 16, 55, 62  
type, 15  
    data, 15  
    hierarchy, 15  
    immutable data, 50  
    , 15, 25  
type -- , 98  
type alias -- , 98  
type hint -- , 98  
type of an object, 15  
TypeError  
    , 57  
types, internal, 18

**U**

u'  
    string literal, 8  
u"  
    string literal, 8  
unary  
    arithmetic operation, 57  
    bitwise operation, 57  
unbinding  
    name, 68  
UnboundLocalError, 35  
Unicode, 16  
Unicode Consortium, 9  
universal newlines -- , 98  
UNIX, 85  
unpacking  
    dictionary, 51  
    in function calls, 56  
    iterable, 62  
unreachable object, 15  
unrecognized escape sequence, 10  
user-defined  
    function, 17  
    function call, 57  
    method, 17  
user-defined function  
    , 17, 57, 79  
user-defined method  
    , 17

## V

value  
    default parameter, 79  
value of an object, 15  
ValueError  
    , 58  
values  
    writing, 65  
variable  
    free, 35  
variable annotation -- , 98

abs, 30  
bytes, 21  
chr, 16  
compile, 73  
complex, 30  
divmod, 29  
eval, 73, 86  
exec, 73  
float, 30  
hash, 21  
id, 15  
int, 30  
len, 16, 17, 27  
open, 18  
ord, 16  
pow, 29  
print, 21  
range, 76  
repr, 65  
round, 30  
slice, 19  
type, 15, 25

as, 71, 77, 78  
async, 81  
await, 57, 81  
elif, 76  
else, 70, 76, 77  
except, 77  
finally, 6971, 77  
from, 52, 71  
in, 76  
yield, 52

virtual environment -- , 99  
virtual machine -- , 99

asynchronous-generator, 54  
Boolean, 16  
built-in function, 18, 57  
built-in method, 18, 57  
callable, 17, 55  
class, 18, 57, 80

- class instance, 18, 57
- complex, 16
- dictionary, 17, 18, 21, 51, 55, 66
- Ellipsis, 16
- floating point, 16
- frame, 19
- frozenset, 16
- function, 17, 18, 57, 79
- generator, 19, 52, 53
- immutable, 16
- immutable sequence, 16
- instance, 18, 57
- integer, 16
- list, 16, 51, 55, 66
- mapping, 17, 18, 55, 66
- method, 17, 18, 57
- module, 18, 55
- mutable, 16, 66
- mutable sequence, 16
- None, 16, 65
- NotImplemented, 16
- numeric, 16, 18
- sequence, 16, 18, 55, 61, 66, 76
- set, 16, 51
- set type, 16
- slice, 28
- string, 55
- traceback, 19, 69, 77
- tuple, 16, 55, 62
- user-defined function, 17, 57, 79
- user-defined method, 17

## W

- while
  - , 70, 71, 76
- \_\_main\_\_, 36, 85
- array, 16
- builtins, 85
- dbm.gnu, 17
- dbm.ndbm, 17
- io, 18
- sys, 77, 85
- Windows, 85
- with
  - , 30, 78
- writing
  - values, 65

## X

- xor
  - bitwise, 58

## Y

- yield
  - examples, 53
  - expression, 52
    - , 52
    - , 69

## Z

- Zen of Python -- Python , 99
- ZeroDivisionError
  - , 58