运维必备Python基础入门到精通

视频课程汇总

Python 函数中文手册

函数列表

abs()	divmod()	input()	open()	staticmethod()
all()	enumerate()	int()	ord()	str()
any()	eval()	isinstance()	pow()	sum()
basestring()	execfile()	issubclass()	print()	super()
bin()	file()	iter()	property()	tuple()
bool()	filter()	len()	range()	type()
bytearray()	float()	list()	raw_input()	unichr()
callable()	format()	locals()	reduce()	unicode()
chr()	frozenset()	long()	reload()	vars()
classmethod()	getattr()	map()	repr()	xrange()
cmp()	globals()	max()	reversed()	zip()
compile()	hasattr()	memoryview()	round()	__import__()
complex()	hash()	min()	set()	apply()
delattr()	help()	next()	setattr()	buffer()
dict()	hex()	object()	slice()	coerce()
dir()	id()	oct()	sorted()	intern()

1、取绝对值

abs(x)

Return the absolute value of a number. The argument may be a plain or long integer or a floating point number. If the argument is a complex number, its magnitude is returned.

如果你不知道绝对值什么意思，那就要补一下小学数学了！

基本用法

```
>>> abs(-3)
3
>>>
```

2

all(*iterable*)

Return True if all elements of the *iterable* are true (or if the iterable is empty). Equivalent to:

3

any(*iterable*)

Return True if any element of the *iterable* is true. If the iterable is empty, return False. Equivalent to:

4

basestring()

This abstract type is the superclass for str and unicode. It cannot be called or instantiated, but it can be used to test whether an object is an instance of str or unicode. isinstance(obj, basestring) is equivalent to isinstance(obj, (str, unicode)).

是字符串和字符编码的超类，是抽象类型。不能被调用或者实例化。可以用来判断实例是否为字符串或者字符编码。

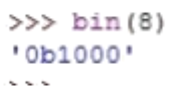
5、二进制转换

bin(*x*)

Convert an integer number to a binary string. The result is a valid Python expression. If *x* is not a Python int object, it has to define an __index__() method that returns an integer.

转换成二进制表达

方法：



```
>>> bin(8)
'0b1000'
```

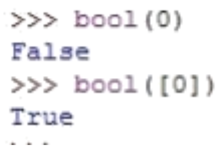
6、布尔类型

bool([*x*])

Convert a value to a Boolean, using the standard truth testing procedure. If *x* is false or omitted, this returns False; otherwise it returns True. bool is also a class, which is a subclass of int. Class bool cannot be subclassed further. Its only instances are False and True

布尔类型的转化

用法：



```
>>> bool(0)
False
>>> bool([0])
True
```

7、二进制数组的转化

bytearray([*source*[, *encoding*[, *errors*]]])

Return a new array of bytes. The bytearray type is a mutable sequence of integers in the range 0 <= x < 256. It has most of the usual methods of mutable sequences, described in *Mutable Sequence Types*, as well as most methods that the str type has, see *String Methods*.

The optional *source* parameter can be used to initialize the array in a few different ways:

- If it is a *string*, you must also give the *encoding* (and optionally, *errors*) parameters; bytearray() then converts the string to bytes using str.encode().
- If it is an *integer*, the array will have that size and will be initialized with null bytes.
- If it is an object conforming to the *buffer* interface, a read-only buffer of the object will be used to initialize the bytes array.
- If it is an *iterable*, it must be an iterable of integers in the range 0 <= x < 256, which are used as the initial contents of the array.

Without an argument, an array of size 0 is created.

8、

callable(*object*)

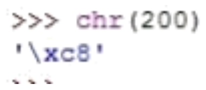
Return True if the *object* argument appears callable, False if not. If this returns true, it is still possible that a call fails, but if it is false, calling *object* will never succeed. Note that classes are callable (calling a class returns a new instance); class instances are callable if they have a __call__() method.

9、数字转化成字符

chr(*i*)

Return a string of one character whose ASCII code is the integer *i*. For example, chr(97) returns the string 'a'. This is the inverse of ord(). The argument must be in the range [0..255], inclusive; ValueError will be raised if *i* is outside that range. See also unichr().

用法：



```
>>> chr(200)
'\xc8'
...
```

10

classmethod(*function*)

Return a class method for *function*.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

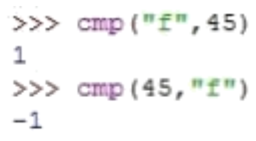
11、两两比较

cmp(*x*, *y*)

> Compare the two objects *x* and *y* and return an integer according to the outcome. The return value is negative if x < y, zero if x == y and strictly positive if x > y.

X 小于 X 输出负（-1），X 等于 Y 输出零（0），X 大于 Y 输出正（1）

用法：



```
>>> cmp("f",45)
1
>>> cmp(45,"f")
-1
```

12

compile(*source*, *filename*, *mode*[, *flags*[, *dont_inherit*]])

Compile the *source* into a code or AST object. Code objects can be executed by an exec statement or evaluated by a call to eval(). *source* can either be a string or an AST object. Refer to the ast module documentation for information on how to work with AST objects.

13

complex([*real*[, *imag*]])

Create a complex number with the value *real* + *imag**j or convert a string or number to a complex number. If the first parameter is a string, it will be interpreted as a complex number and the function must be called without a second parameter. The second parameter can never be a string. Each argument may be any numeric type (including complex). If *imag* is omitted, it defaults to zero and the function serves as a numeric conversion function like int(), long() and float(). If both arguments are omitted, returns 0j.

14

delattr(*object*, *name*)

This is a relative of setattr(). The arguments are an object and a string. The string must be the name of one of the object’ s attributes. The function deletes the named attribute, provided the object allows it. For example, delattr(x, 'foobar') is equivalent to del x.foobar.

15、字典

dict([*arg*])

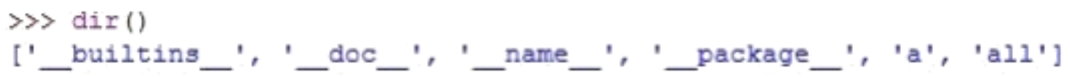
> Create a new data dictionary, optionally with items taken from *arg*. The dictionary type is described in *Mapping Types — dict*.
>
> For other containers see the built in list, set, and tuple classes, and the collections module.

16、很重要的函数，属性输出

dir([*object*])

Without arguments, return the list of names in the current local scope. With an argument, attempt to return a list of valid attributes for that object.

方法



```
>>> dir()
['__builtins__', '__doc__', '__name__', '__package__', 'a', 'all']
```

17

divmod(*a*, *b*)

Take two (non complex) numbers as arguments and return a pair of numbers consisting of their quotient and remainder when using long division. With mixed operand types, the rules for binary arithmetic operators apply. For plain and long integers, the result is the same as (a // b, a % b). For floating point numbers the result is (q, a % b), where *q* is usually math.floor(a / b) but may be 1 less than that. In any case q * b + a % b is very close to *a*, if a % b is non-zero it has the same sign as *b*, and 0 <= abs(a % b) < abs(b).

18

enumerate(*sequence*[, *start=0*])

Return an enumerate object. *sequence* must be a sequence, an *iterator*, or some other object which supports iteration. The next() method of the iterator returned by enumerate() returns a tuple containing a count (from *start* which defaults to 0) and the corresponding value obtained from iterating over *iterable*. enumerate() is useful for obtaining an indexed series: (0, seq[0]), (1, seq[1]), (2, seq[2])

19

eval(*expression*[, *globals*[, *locals*]])

The arguments are a string and optional globals and locals. If provided, *globals* must be a dictionary. If provided, *locals* can be any mapping object.

Changed in version 2.4: formerly *locals* was required to be a dictionary.

20

execfile(*filename*[, *globals*[, *locals*]])

This function is similar to the exec statement, but parses a file instead of a string. It is different from the import statement in that it does not use the module administration — it reads the file unconditionally and does not create a new module.

和 exec 很相似的函数

21

file(*filename*[, *mode*[, *bufsize*]])

Constructor function for the file type, described further in section *File Objects*. The constructor’ s arguments are the same as those of the open() built-in function described below.

When opening a file, it’ s preferable to use open() instead of invoking this constructor directly. file is more suited to type testing (for example, writing isinstance(f, file)).

22

filter(*function*, *iterable*)

> Construct a list from those elements of *iterable* for which *function* returns true. *iterable* may be either a sequence, a container which supports iteration, or an iterator. If *iterable* is a string or a tuple, the result also has

that type; otherwise it is always a list. If *function* is None, the identity function is assumed, that is, all elements of *iterable* that are false are removed.

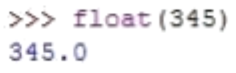
Note that filter(function, iterable) is equivalent to [item for item in iterable if function(item)] if function is not None and [item for item in iterable if item] if function is None.

See itertools.ifilter() and itertools.ifilterfalse() for iterator versions of this function, including a variation that filters for elements where the *function* returns false.

23、浮点数值转化

float([*x*])

用法：



```
>>> float(345)
345.0
```

24

format(*value*[, *format_spec*])

Convert a *value* to a “formatted” representation, as controlled by *format_spec*. The interpretation of *format_spec* will depend on the type of the *value* argument, however there is a standard formatting syntax that is used by most built-in types: *Format Specification Mini-Language*.

25

frozenset([*iterable*])

Return a frozenset object, optionally with elements taken from *iterable*. The frozenset type is described in *Set Types — set, frozenset*.

For other containers see the built in dict, list, and tuple classes, and the collections module.

26

getattr(*object*, *name*[, *default*])

Return the value of the named attribute of *object*. *name* must be a string. If the string is the name of one of the object’ s attributes, the result is the value of that attribute. For example, getattr(x, 'foobar') is equivalent to x.foobar. If the named attribute does not exist, *default* is returned if provided, otherwise AttributeError is raised.

27

globals()

Return a dictionary representing the current global symbol table. This is always the dictionary of the current module (inside a function or method, this is the module where it is defined, not the module from which it is called).

28

hasattr(*object*, *name*)

Return the hash value of the object (if it has one). Hash values are integers. They are used to quickly compare dictionary keys during a dictionary lookup. Numeric values that compare equal have the same hash value (even if they are of different types, as is the case for 1 and 1.0).

29

hash(*object*)

Return the hash value of the object (if it has one). Hash values are integers. They are used to quickly compare dictionary keys during a dictionary lookup. Numeric values that compare equal have the same hash value (even if they are of different types, as is the case for 1 and 1.0).

30、很重要的帮助函数方法

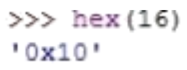
help([*object*])

31、十六进制转化

hex(*x*)

Convert an integer number (of any size) to a hexadecimal string. The result is a valid Python expression.

用法：



```
>>> hex(16)
'0x10'
```

32

id(*object*)

Return the “identity” of an object. This is an integer (or long integer) which is guaranteed to be unique and constant for this object during its lifetime. Two objects with non-overlapping lifetimes may have the same id() value.

33

input([*prompt*])

Equivalent to eval(raw_input(prompt)).

34

int([*x*[, *base*]])

Convert a string or number to a plain integer. If the argument is a string, it must contain a possibly signed decimal number representable as a Python integer, possibly embedded in whitespace. The *base* parameter gives the base for the conversion (which is 10 by default) and may be any integer in the range [2, 36], or zero. If *base* is zero, the proper radix is determined based on the contents of string; the interpretation is the same as for integer literals. (See *Numeric literals*.) If *base* is specified and *x* is not a string, TypeError is raised. Otherwise, the argument may be a plain or long integer or a floating point number. Conversion of floating point numbers to integers truncates (towards zero). If the argument is outside the integer range a long object will be returned instead. If no arguments are given, returns 0.

35

isinstance(*object*, *classinfo*)

Return true if the *object* argument is an instance of the *classinfo* argument, or of a (direct or indirect) subclass thereof. Also return true if *classinfo* is a type object (new-style class) and *object* is an object of that type or of a (direct or indirect) subclass thereof. If *object* is not a class instance or an object of the given type, the function always returns false. If *classinfo* is neither a class object nor a type object, it may be a tuple of class or type objects, or may recursively contain other such tuples (other sequence types are not accepted). If *classinfo* is not a class, type, or tuple of classes, types, and such tuples, a TypeError exception is raised.

36

issubclass(*class*, *classinfo*)

Return true if *class* is a subclass (direct or indirect) of *classinfo*. A class is considered a subclass of itself. *classinfo* may be a tuple of class objects, in which case every entry in *classinfo* will be checked. In any other case, a TypeError exception is raised.

37、导管，窗口，容器，数据的窗口化

iter(*o*[, *sentinel*])

Return an *iterator* object. The first argument is interpreted very differently depending on the presence of the second argument. Without a second argument, *o* must be a collection object which supports the iteration protocol (the __iter__() method), or it must support the sequence protocol (the __getitem__() method with integer arguments starting at 0). If it does not support either of those protocols, TypeError is raised. If the second argument, *sentinel*, is given, then *o* must be a callable object. The iterator created in this case will call *o* with no arguments for each call to its next() method; if the value returned is equal to *sentinel*, StopIteration will be raised, otherwise the value will be returned.

iter(*o*[, *sentinel*])返回一个迭代器对象。第一个参数根据第二个参数进行编译。第二参数为空，O 必须是支持迭代器的协议 (the __iter__() method)的集合对象,或者支持顺序协议(the __getitem__()method with integer arguments staring at 0).如果不支持其中任意一种协议，程序将会抛出类型异常。

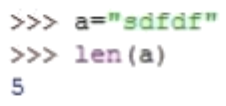
假如第二个参数被给出，然后 O 必须是一个可被调用的对象。迭代器被创建万一 will 掉用 O with 没有参数 for each call to its next() method; 如果返回值和初始值相同 *l*, StopIteration 将会抛出, 否则值会被返回！

38、计算长度（常用函数）

len(*s*)

Return the length (the number of items) of an object. The argument may be a sequence (string, tuple or list) or a mapping (dictionary).

用法：



```
>>> a="sdfdf"
>>> len(a)
5
```

39、转化成列表

list([*iterable*])

Return a list whose items are the same and in the same order as *iterable* 's items. *iterable* may be either a sequence, a container that supports iteration, or an iterator object. If *iterable* is already a list, a copy is made and returned, similar to iterable[:]. For instance, list('abc') returns ['a', 'b', 'c'] and list((1, 2, 3)) returns [1, 2, 3]. If no argument is given, returns a new empty list, [].

40

locals()

Update and return a dictionary representing the current local symbol table. Free variables are returned by locals() when it is called in function blocks, but not in class blocks.

41

long([*x*[, *base*]])

Convert a string or number to a long integer. If the argument is a string, it must contain a possibly signed number of arbitrary size, possibly embedded in whitespace. The *base* argument is interpreted in the same way as for int(), and may only be given when *x* is a string. Otherwise, the argument may be a plain or long integer or a floating point number, and a long integer with the same value is returned. Conversion of floating point numbers to integers truncates (towards zero). If no arguments are given, returns 0L.

42

map(*function*, *iterable*, *...*)

Apply *function* to every item of *iterable* and return a list of the results. If additional *iterable* arguments are passed, *function* must take that many arguments and is applied to the items from all iterables in parallel. If one iterable is shorter than another it is assumed to be extended with None items. If *function* is None, the identity function is assumed; if there are multiple arguments, map() returns a list consisting of tuples containing the corresponding items from all iterables (a kind of transpose operation). The *iterable* arguments may be a sequence or any iterable object; the result is always a list.

43、最大值

max(*iterable*[, *args...*][, *key*])

With a single argument *iterable*, return the largest item of a non-empty iterable (such as a string, tuple or list). With more than one argument, return the largest of the arguments.

The optional *key* argument specifies a one-argument ordering function like that used for list.sort(). The *key* argument, if supplied, must be in keyword form (for example, max(a,b,c,key=func)).

44

memoryview(*obj*)

Return a “memory view” object created from the given argument. See

memoryview type for more information.

45、最小值

min(*iterable*[, *args...*][, *key*])

With a single argument *iterable*, return the smallest item of a non-empty iterable

(such as a string, tuple or list). With more than one argument, return the smallest

of the arguments.

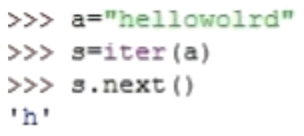
46、迭代以后的函数

next(*iterator*[, *default*])

Retrieve the next item from the *iterator* by calling its next() method. If *default* is

given, it is returned if the iterator is exhausted, otherwise StopIteration is raised.

用法：



```
>>> a="hellowolrd"
>>> s=iter(a)
>>> s.next()
'h'
```

47

object()

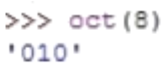
Return a new featureless object. object is a base for all new style classes. It has the methods that are common to all instances of new style classes.

48、八进制字符串的转化

oct(*x*)

Convert an integer number (of any size) to an octal string. The result is a valid Python expression.

用法：



```
>>> oct(8)
'010'
```

49

open(*filename*[, *mode*[, *bufsize*]])

Open a file, returning an object of the file type described in section *File Objects*. If the file cannot be opened, IOError is raised. When opening a file, it’ s preferable to use open() instead of invoking the file constructor directly.

50、字符转化成 ASCII码

ord(*c*)

Given a string of length one, return an integer representing the Unicode code point of the character when the argument is a unicode object, or the value of the byte when the argument is an 8-bit string. For example, ord('a') returns the integer

97, ord(u'\u2020') returns 8224. This is the inverse of chr() for 8-bit strings and of unichr() for unicode objects. If a unicode argument is given and Python was built with UCS2 Unicode, then the character’ s code point must be in the range [0..65535] inclusive; otherwise the string length is two, and a TypeError will be raised.

51

pow(*x*, *y*[, *z*])

Return *x* to the power *y*; if *z* is present, return *x* to the power *y*, modulo *z* (computed more efficiently than pow(x, y) % z). The two-argument form pow(x, y) is equivalent to using the power operator: x**y.

52、print 函数原来本身就是函数。

print([*object*, ...][, *sep=' '*][, *end='\n'*][, *file=sys.stdout*])

Print *object*(s) to the stream *file*, separated by *sep* and followed by *end*. *sep*, *end* and *file*, if present, must be given as keyword arguments.

53

property([*fget*[, *fset*[, *fdel*[, *doc*]]]])

Return a property attribute for *new-style class*es (classes that derive from object).

54

range([*start*], *stop*[, *step*])

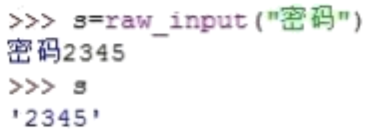
起始位置，终止位置，步长

55

raw_input([*prompt*])

If the *prompt* argument is present, it is written to standard output without a trailing newline.

用法：



```
>>> s=raw_input("密码")
密码2345
>>> s
'2345'
```

56

reduce(*function*, *iterable*[, *initializer*])

Apply *function* of two arguments cumulatively to the items of *iterable*, from left to right, so as to reduce the iterable to a single value. For example, reduce(lambda x, y: x+y, [1, 2, 3, 4, 5]) calculates ((((1+2)+3)+4)+5). The left argument, *x*, is the accumulated value and the right argument, *y*, is the update value from the *iterable*. If the optional *initializer* is present, it is placed before the items of the iterable in the calculation, and serves as a default when the iterable is empty. If *initializer* is not given and *iterable* contains only one item, the first item is returned.

57、重载模块，很重要的函数

reload(*module*)

58

repr(*object*)

Return a string containing a printable representation of an object. This is the same value yielded by conversions (reverse quotes). It is sometimes useful to be able to access this operation as an ordinary function. For many types, this function makes an attempt to return a string that would yield an object with the same value when passed to eval(), otherwise the representation is a string enclosed in angle brackets that contains the name of the type of the object together with additional information often including the name and address of the object. A class can control what this function returns for its instances by defining a __repr__() method.

59

reversed(*seq*)

Return a reverse *iterator*. *seq* must be an object which has a __reversed__() method or supports the sequence protocol (the __len__() method and the __getitem__() method with integer arguments starting at 0).

60

round(*x*[, *n*])

Return the floating point value *x* rounded to *n* digits after the decimal point. If *n* is omitted, it defaults to zero. The result is a floating point number. Values are rounded to the closest multiple of 10 to the power minus *n*; if two multiples are equally close, rounding is done away from 0 (so. for example, round(0.5) is 1.0 and round(-0.5) is -1.0).

61、去重，但是不改变原始数据

set([*iterable*])

Return a new set, optionally with elements taken from *iterable*. The set type is described in *Set Types — set, frozenset*.

62

setattr(*object*, *name*, *value*)

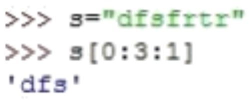
This is the counterpart of getattr(). The arguments are an object, a string and an arbitrary value. The string may name an existing attribute or a new attribute. The function assigns the value to the attribute, provided the object allows it. For example, setattr(x, 'foobar', 123) is equivalent to x.foobar = 123.

63、切片 起始位置，终止位置，步长

slice([*start*], *stop*[, *step*])

Return a *slice* object representing the set of indices specified by range(start, stop, step). The *start* and *step* arguments default to None. Slice objects have read-only data attributes start, stop and step which merely return the argument values (or their default). They have no other explicit functionality; however they are used by Numerical Python and other third party extensions. Slice objects are also generated when extended indexing syntax is used. For example: a[start:stop:step] or a[start:stop, i]. See itertools.islice() for an alternate version that returns an iterator.

用法



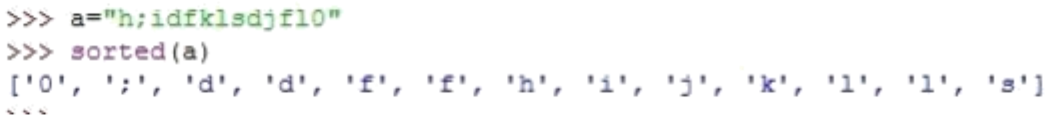
```
>>> s="dfsfrtr"
>>> s[0:3:1]
'dfs'
```

64、排序

sorted(*iterable*[, *cmp*[, *key*[, *reverse*]]])

Return a new sorted list from the items in *iterable*.

用法



```
>>> a="h;idfklsdjfl0"
>>> sorted(a)
['0', ';', 'd', 'd', 'f', 'f', 'h', 'i', 'j', 'k', 'l', 'l', 's']
```

65、静态方法函数 调用类方法的一种函数

staticmethod(*function*)

Return a static method for *function*.

66.字符串转化

str([*object*])

Return a string containing a nicely printable representation of an object. For strings, this returns the string itself. The difference with repr(object) is that str(object) does not always attempt to return a string that is acceptable to eval(); its goal is to return a printable string. If no argument is given, returns the empty string, ''.

67、求和

sum(*iterable*[, *start*])

Sums *start* and the items of an *iterable* from left to right and returns the total. *start* defaults to 0. The *iterable* 's items are normally numbers, and the start value is not allowed to be a string.

68

super(*type*[, *object-or-type*])

Return a proxy object that delegates method calls to a parent or sibling class of *type*. This is useful for accessing inherited methods that have been overridden in a class. The search order is same as that used by getattr() except that the *type* itself is skipped.

69、元组

tuple([*iterable*])

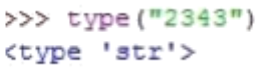
Return a tuple whose items are the same and in the same order as *iterable* 's items. *iterable* may be a sequence, a container that supports iteration, or an iterator object. If *iterable* is already a tuple, it is returned unchanged. For instance, tuple('abc') returns ('a', 'b', 'c') and tuple([1, 2, 3]) returns (1, 2, 3). If no argument is given, returns a new empty tuple, ().

70、类型

type(*object*)

Return the type of an *object*. The return value is a type object. The isinstance() built-in function is recommended for testing the type of an object.

用法：



```
>>> type("2343")
<type 'str'>
```

71

unichr(*i*)

Return the Unicode string of one character whose Unicode code is the integer *i*. For example, unichr(97) returns the string u'a'. This is the inverse of ord() for Unicode strings. The valid range for the argument depends how Python was configured – it may be either UCS2 [0..0xFFFF] or UCS4 [0..0x10FFFF]. ValueError is raised otherwise. For ASCII and 8-bit strings see chr().

72

unicode([*object*[, *encoding*[, *errors*]]])

Return the Unicode string version of *object* using one of the following modes:

73

vars([*object*])

Without an argument, act like locals().

74

xrange([*start*], *stop*[, *step*])

This function is very similar to range(), but returns an "xrange object" instead of a list. This is an opaque sequence type which yields the same values as the corresponding list, without actually storing them all simultaneously. The advantage of xrange() over range() is minimal (since xrange() still has to create the values when asked for them) except when a very large range is used on a memory-starved machine or when all of the range's elements are never used (such as when the loop is usually terminated with break).

75

zip([*iterable*, ...])

76

__import__(*name*[, *globals*[, *locals*[, *fromlist*[, *level*]]]])

Note

This is an advanced function that is not needed in everyday Python programming.

This function is invoked by the import statement. It can be replaced (by importing the __builtin__ module and assigning to __builtin__.__import__) in order to change semantics of the import statement, but nowadays it is usually simpler to use import hooks (see **PEP 302**). Direct use of __import__() is rare, except in cases where you want to import a module whose name is only known at runtime.