自带函数

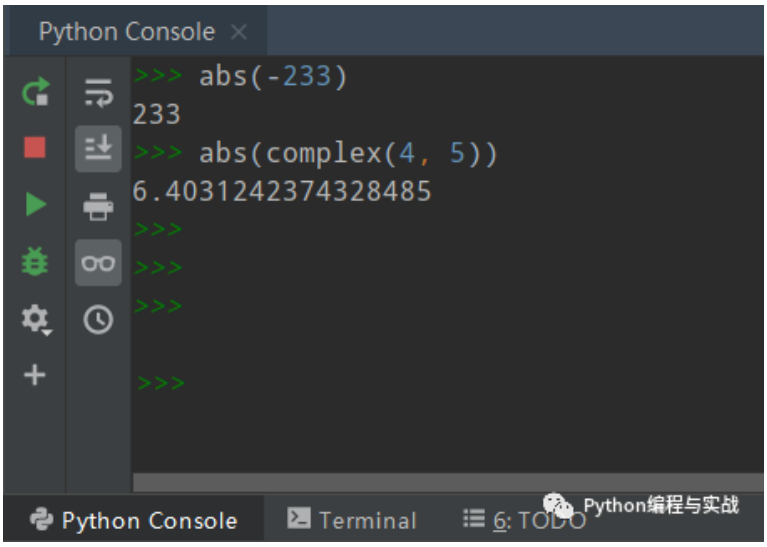
A	**E**		**R**
abs()	enumerate()		range()
aiter()	eval()	**L**	repr()
all()	exec()	len()	reversed()
anext()	**F**	list()	round()
any()	filter()	locals()	**S**
ascii()	float()	**M**	set()
B	format()	map()	setattr()
bin()	frozenset()	max()	slice()
bool()	**G**	memoryview()	sorted()
breakpoint()	getattr()	min()	staticmethod()
bytearray()	globals()	**N**	str()
bytes()	**H**	next()	sum()
C	hasattr()	**O**	super()
callable()	hash()	object()	**T**
chr()	help()	oct()	tuple()
classmethod()	hex()	open()	type()
compile()	**I**	ord()	**V**
complex()	id()	**P**	vars()
D	input()	pow()	**Z**
delattr()	int()	print()	zip()
dict()	isinstance()	property()	_import()
dir()	issubclass()		
divmod()	iter()		

1. abs()

语法

abs(x), 返回一个数的绝对值。参数可以是一个整数或浮点数。如果参数是一个复数，则返回它的模

示例



2. all()

语法

all(iterable), 如果 iterable 的**所有元素**均为 True（或 iterable 为空）则返回 True

等价代码如下:

```
1.  def all(iterable):
2.      for element in iterable:
3.          if not element:
4.              return False
5.      return True
```

3. any()

语法

any(iterable), 如果 iterable 的**任一元素**为 True, 则返回 True 如果可迭代对象为空，返回 False

等价代码如下:

```
1.  def any(iterable):
2.      for element in iterable:
3.          if element:
4.              return True
5.      return False
```

4. ascii()

语法

ascii(object), 返回对象的纯 ASCII 表示形式。

ascii() 函数类似 repr() 函数, 返回一个表示对象的字符串, 但是对于字符串中的非 ASCII 字符则返回通过 repr() 函数使用 \x, \u 或 \U 编码的字符。

生成字符串类似 Python2 版本中 repr() 函数的返回值。

5. bin()

语法

bin(x), 将一个整数转变为一个前缀为“0b”的二进制字符串

6.bool()

语法

返回一个布尔值，True 或者 False，如果没有参数，也是返回 False
bool 是 int 的子类

7. breakpoint()

语法

breakpoint(*args, **kws)，它调用 sys.breakpointhook()，直接传递 args 和 kws，进入 pdb 调试器
这个用的很少，几乎没用过..

8.bytearray()

语法

class bytearray([source[, encoding[, errors]]])
如果 source 为整数，则返回一个长度为 source 的初始化数组；
如果 source 为字符串，则必须提供 encoding 参数。并按照指定的 encoding 将字符串转换为字节序列；
如果 source 为可迭代类型，则元素必须为[0 ,255] 中的整数；
如果 source 为与 buffer 接口一致的对象，则此对象也可以被用于初始化 bytearray。
如果没有输入任何参数，则创建大小为 0 的数组。

9.bytes()

语法

bytes() 函数返回一个新的 bytes 对象，该对象是一个 0 <= x < 256 区间内的整数不可变序列。它是 bytearray 的不可变版本。

10.callable()

语法

callable(object), 用于检查一个对象是否可调用，可调用返回 True,否则返回 False
但是返回 True，调用对象 object 仍可能失败，但如果返回 False, 则调用 object 肯定不会成功
另外，类是可调用的，调用类将返回一个新的实例
如果实例所属的类有 __call__() 方法，则也是可调用的。

11.chr()

语法

chr(i), 返回参数对应的 ASCII 字符，i：可以是 10 进制也可以是 16 进制的形式的数字，数字范围为 0 到 1,114,111 (16 进制为 0x10FFFF)。

12.classmethod()

语法

将一个方法封装成类方法，该方法不需要实例化，不需要 self 参数，第一个参数是表示自身类的 cls 参数
可以用来调用类的属性，类的方法等

13.compile()

语法

compile(source, filename, mode, flags=0, dont_inherit=False, optimize=-1)
将 *source* 编译成代码或 AST 对象。代码对象可以被 exec()或 eval() 执行。
source :可以是常规的字符串、字节字符串，或者 AST 对象
filename：代码文件名称，如果不是从文件读取代码则传递一些可辨认的值。
mode：指定编译代码的种类。可以指定为 exec, eval, single。
flags：变量作用域，局部命名空间，如果被提供，可以是任何映射对象。

flags 和 *dont_inherit* 是用来控制编译源码时的标志。

14.complex()

语法

class complex([real[, imag]])

返回值为 real + imag*1j 的复数，或将字符串或数字转换为复数。

如果第一个形参是字符串，则它被解释为一个复数，并且函数调用时不能有第二个形参

参数

real：int, long, float 或字符串。

imag：int, long, float 不能为字符串

15. delattr()

语法

delattr(object, name)

实参是一个对象和一个字符串。该字符串必须是对象的某个属性。如果对象允许，该函数将删除指定的属性。

16. dict()

语法

1. class dict(**kwarg)
2. class dict(mapping, **kwarg)
3. class dict(iterable, **kwarg)

创建一个新的字典

参数

**kwargs：关键字 mapping：元素的容器。iterable：可迭代对象。

17. dir()

语法

dir([object]), 如果没有参数调用，则返回当前范围中的名称。

带参数时，返回参数的属性、方法列表

18.divmod()

语法

divmod(a, b)，函数接收两个数字类型（非复数）参数，返回一个包含商和余数的元组(a // b, a % b)。

19.enumerate()

语法

enumerate(iterable, start=0)，返回一个枚举对象。*iterable* 必须是一个序列，或 iterator，或其他支持迭代的对象

示例

1. >>> codes = ['Python', 'Java', 'GO', 'C++']
2. >>> list(enumerate(codes, start=2))
3. [(2, 'Python'), (3, 'Java'), (4, 'GO'), (5, 'C++')]

20.eval()

语法

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

参数

expression：Python 表达式。

globals：必须是一个字典对象。

locals：变量作用域，局部命名空间，如果被提供，可以是任何映射对象。
执行一个字符串表达式，并返回表达式的值

21.exec()
语法
exec(object[, globals[, locals]])
exec 执行储存在字符串或文件中的 Python 语句，相比于 eval，exec 可以执行更复杂的 Python 代码。
参数
object：必选参数，必须是字符串或 code 对象。如果 object 是一个字符串，该字符串会先被解析为一组 Python 语句，然后在执行（除非发生语法错误）。如果 object 是一个 code 对象，那么它只是被简单的执行。
globals：可选参数，表示全局命名空间（存放全局变量）必须是一个字典对象。
locals：可选参数，表示当前局部命名空间（存放局部变量）可以是任何映射对象。如果该参数被忽略，那么它将会取与 globals 相同的值。

22.filter()
语法
filter(function, iterable)
filter() 函数用于过滤序列，过滤掉不符合条件的元素，返回一个迭代器对象，如果要转换为列表，可以使用 list() 来转换。
该接收两个参数，第一个为函数，第二个为序列，序列的每个元素作为参数传递给函数进行判，然后返回 True 或 False，最后将返回 True 的元素放到新列表中。

23.float()
语法
将整数和字符串转换成浮点数。

24.format()
语法
format(value[, format_spec])，该函数主要作用是增强字符串格式化的功能，基本语法是通过 {} 和 : 来代替以前的 %
format 函数可以接受不限个参数，位置可以不按顺序。

25.frozenset()
语法
class frozenset([iterable])
frozenset() 返回一个冻结的集合，冻结后集合不能再添加或删除任何元素。

26.getattr()
语法
getattr(object, name[, default])
返回对象命名属性的值。*name* 必须是字符串。如果该字符串是对象的属性之一，则返回该属性的值。
例如， getattr(x, 'foobar') 等同于 x.foobar。如果指定的属性不存在，且提供了 *default* 值，则返回它，否则触发 AttributeError

27.globals()
语法
返回包含当前作用域的全局变量的字典。

28.hasattr()

语法

hasattr(object，name)，该实参是一个对象和一个字符串。如果字符串是对象的属性之一的名称，则返回 True，否则返回 False。

此功能是通过调用 getattr(object, name) 看是否有 AttributeError 异常来实现的。

29.hash()

语法

hash(object), 返回对象 object 的哈希值

hash() 函数可以应用于数字、字符串和对象，不能直接应用于 list、set、dictionary。

30.help()

语法

为你提供帮助的函数，查看某个函数的帮助信息

31.hex()

语法

hex(x), 将整数转换为以“0x”为前缀的小写十六进制字符串。

32.id()

语法

id(object), 返回该对象的内存地址

33.input()

语法

input() 函数接受一个标准输入数据，返回为 string 类型。

在 Python3.x 中 raw_input() 和 input() 进行了整合，去除了 raw_input( )，仅保留了 input( )函数，其接收任意输入，将所有输入默认为字符串处理，并返回字符串类型。

34.int()

语法

将一个字符串或数字转换为整型。

35.isinstance()

语法

isinstance(object, classinfo)

isinstance() 函数来判断一个对象是否是一个已知的类型，类似 type()。i sinstance() 与 type() 区别：type() 不会认为子类是一种父类类型，不考虑继承关系。

isinstance() 会认为子类是一种父类类型，考虑继承关系。

如果要判断两个类型是否相同推荐使用 isinstance()。

36.issubclass()

语法

issubclass(class, classinfo)

issubclass() 方法用于判断参数 class 是否是类型参数 classinfo 的子类。

37.iter()

语法

iter(object[, sentinel])

返回一个 *iterator* 对象

如果传递了第二个参数，则参数 object 必须是一个可调用的对象，此时，iter 创建了一个迭代器对象，每次调用这个迭代器对象的 next()方法时，都会调用 object。

38.len()
语法
返回对象的长度

39.list()
语法
将元组或字符串转换成列表

40.locals()
语法
locals() 函数会以字典类型返回当前位置的全部局部变量。

41.map()
语法
map(function, iterable, ...)
返回一个将 function 应用于 iterable 中每一项并输出其结果的迭代器

42.max()
语法
返回可迭代对象中最大的元素

43.memoryview()
语法
返回给定参数的内存视图

44. min()
语法
返回可迭代对象中最小的元素，或者返回两个及以上实参中最小的。

45.next()
语法
通过调用 *iterator* 的 `__next__()`[1] 方法获取下一个元素。如果迭代器耗尽，则返回给定的 *default*，如果没有默认值则触发 `StopIteration`[2]。

46.object()
语法
返回一个没有特征的新对象。`object`[3] 是所有类的基类。
它具有所有 Python 类实例的通用方法。这个函数不接受任何实参。

47. oct()
语法
返回整数的八进制表示形式

48.open()
语法
open(file, mode='r', buffering=-

1, encoding=None, errors=None, newline=None, closefd=True, opener=None)
open() 函数用于打开一个文件，并返回文件对象，在对文件进行处理过程都需要使用到这个函数，如果该文件无法被打开，会抛出 *OSError*

49.ord()
语法
对单个字符的字符串，返回它的 Unicode 编码的整数
例如 ord('a') 返回整数 97， ord('€') （欧元符号）返回 8364 。是 chr() 的逆函数。

50. pow()
语法
pow(base, exp[, mod])
函数是计算 base 的 exp 次方，如果 mod 存在，则再对结果进行取模，其结果等效于 pow(base,exp) %mod。

51.print()
语法
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
将 objects 打印到 file 指定的文本流, 默认为 *sys.stdout*

52.property()
语法
property() 函数的作用是在新式类中返回属性值。

53.range()
语法
range() 函数返回一个可迭代对象

54.repr()
语法
返回包含一个对象的可打印表示形式的字符串。对于大多数的类型，eval(repr(obj)) == obj

55.reversed()
语法
返回给定序列值的反向迭代器

56.round()
语法
返回 number 四舍五入到小数点后 ndigits 位精度的值。如果 ndigits 被省略或为 None，则返回最接近输入值的整数
对精度要求高的，不减少使用该函数

57.set()
语法
set() 函数创建一个无序不重复元素集，删除重复数据，可以用于计算交集、差集、并集等。

58.setattr()
语法
setattr(object, name, value)
其参数为一个对象、一个字符串和一个任意值，将给定对象上的命名属性设置为指定值。

例如，setattr(python, 'name', 123) 等价于 python.name= 123

59.slice()

语法

slice() 函数实现切片对象，主要用在切片操作函数里的参数传递。

60.sorted()

语法

sorted(iterable, key=None, reverse=False)

对所有可迭代的对象进行排序操作,默认为升序

sort 与 sorted 区别：sort 是应用在 list 上的方法，sorted 可以对所有可迭代的对象进行排序操作。

sort 方法返回的是对已经存在的列表进行操作

而 sorted 方法返回的是一个新的 list

61.staticmethod()

语法

将方法转换为静态方法, 该方法不要钱传递参数

62.str()

语法

返回一个对象的 string 格式

63.sum()

语法

sum(iterable[, start]), 从 start 开始自左向右对 iterable 的项求和并返回总计值

64.super()

语法

用于调用父类的一个方法, 用来解决多重继承问题的

示例

```
class Spider(object):
    """爬虫基类"""

    def __init__(self, *args, **kwargs):
        self.session = requests.session()
        self.session.headers.update({
            "User-Agent": UserAgent(default=True),
        })

    def request(self, url, session=None, method="get", proxy=None,
                headers=None, params=None, data=None, json=None, timeout=200,
                return_type=None, retry: int = 3, retry_condition=None, **kwargs):...


class TSpider(Spider):

    def __init__(self):
        super(TSpider, self).__init__()

    def crawl(self):
        response = self.request("https://www.baidu.com/", method="post")
        print(response.text)
```

65. tuple()

语法

将可迭代系列（如列表）转换为元组

66.type()

语法

传入一个参数时，返回 *object* 的类型, 传入三个参数时，返回一个新的 *type* 对象

```
>>> class X:
        a = 1

>>> X = type('X', (object,), dict(a=1))
>>> X
<class '__main__.X'>
```

67.vars()

语法

返回模块、类、实例或任何其它具有 `__dict__`[4] 属性的对象的 __dict__ 属性。

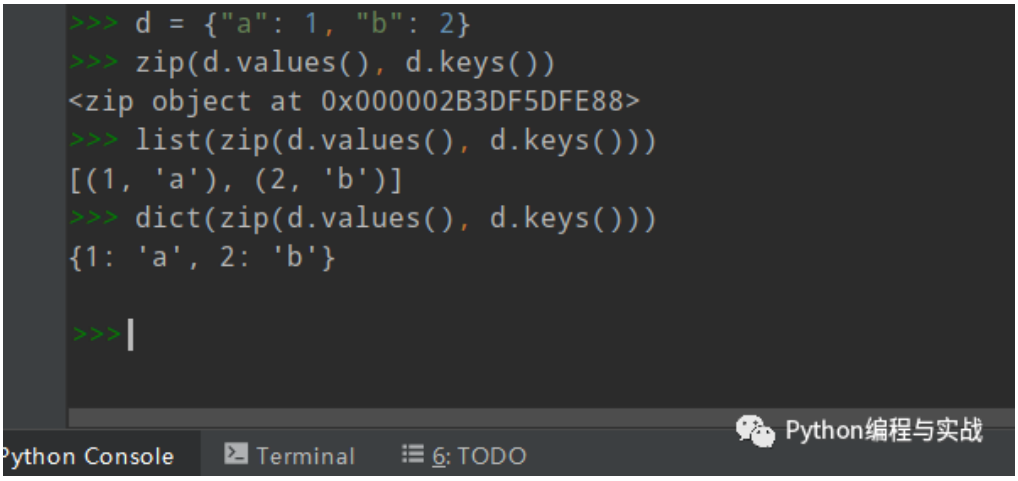
68. zip()

语法

用于将可迭代的对象作为参数，将对象中对应的元素打包成一个个元组，然后返回由这些元组组成的对象

可以使用 list() 转换来输出列表, 如果各个迭代器的元素个数不一致，则返回的列表长度以最短的对象为准

示例



69. *_import_*()

语法

__import__(name, globals=None, locals=None, fromlist=(), level=0)

import() 函数用于动态加载类和函数 。

如果一个模块经常变化就可以使用 _import()_ 来动态载入

以上便是 Python 全部的 69 个内置函数，语法规则基于 Python3.8.6