**0、两个变量值互换**

>>> a=1

>>> b=2

>>> a,b=b,a

>>> a

2

>>> b

1

**1、连续赋值**

a = b = c = 50

**2、自动解包**

>>> a,b,c = [1,2,3]

>>> a

1

>>> b

2

>>> c

3

>>>

>>>

>>> a, \*others = [1,2,3,4]

>>> a

1

>>> others

[2, 3, 4]

>>>

**4、链式比较**

a = 15

if (10 < a < 20):

print("Hi")

等价于

a = 15

if (a>10 and a<20):

print("Hi")

**5、重复列表**

>>> [5,2]\*4

[5, 2, 5, 2, 5, 2, 5, 2]

**6、重复字符串**

>>> "hello"\*3

'hellohellohello'

**7、三目运算**

age = 30

slogon = "牛逼" if age == 30 else "niubility"

等价于

if age == 30:

slogon = "牛逼"

else:

slogon = "niubility"

**8、字典合并**

>>> a= {"a":1}

>>> b= {"b":2}

>>> {\*\*a, \*\*b}

{'a': 1, 'b': 2}

>>>

**9、字符串反转**

>>> s = "i love python"

>>> s[::-1]

'nohtyp evol i'

>>>

**10、列表转字符串**

>>> s = ["i", "love", "pyton"]

>>> " ".join(s)

'i love pyton'

>>>

**11、for else 语句**

检查列表foo是否有0，有就提前结束查找，没有就是打印“未发现"

found = False

for i in foo:

if i == 0:

found = True

break

if not found:

print("未发现")

如果用 for else 语法来写可以省几行代码

for i in foo:

if i == 0:

break

else:

print("未发现")

**11、字典推导式**

>>> m = {x: x\*\*2 for x in range(5)}

>>> m

{0: 0, 1: 1, 2: 4, 3: 9, 4: 16}

>>>

**12、用Counter查找列表中出现最多的元素**

>>> content = ["a", "b", "c", "a", "d", "c", "a"]

>>> from collections import Counter

>>> c = Counter(content)

>>> c.most\_common(1)

[('a', 3)]

>>>

出现第1多的元素是a，一共出现3次， 你也可以用类似的方法找出第二多或者第三多的

**13、默认值字典**

给字典中的value设置为列表，普通方法

>>> d = dict()

if 'a' not in d:

d['a'] = []

d['a'].append(1)

使用defaultdict默认字典构建一个初始值为空列表的字典

from collections import defaultdict

d = defaultdict(list)

d['a'].append(1)

**14、赋值表达式**

这是3.8的新特性，赋值表达式又成为海象运算符:=， 可以将变量赋值和表达式放在一行，什么意思？ 看代码就明白

>>> import re

>>> data = "hello123world"

>>> match = re.search("(\d+)", data) # 3

>>> if match: # 4

... num = match.group(1)

... else:

... num = None

>>> num

'123'

第3、4行 可以合并成一行代码

>>> if match:=re.search("(\d+)", data):

... num = match.group(1)

... else:

... num = None

...

>>> num

'123'

**15、isinstance**

isinstance 函数可用于判断实例的类型，其实第二个参数可以是多个数据类型组成的元组。例如：

isinstance(x, (int, float))

# 等价于

isinstance(x, int) or isinstance(x, float)

类似的函数还有字符串的startswith，endswith，例如：

s.startswith(('"""', "'''"))

# 等价于

s.startswith("'''") or s.startswith('"""')

**16、用 http.server 共享文件**

# python3

python3 -m http.server

# python2

python -m SimpleHTTPServer 8000

效果如下，可以在浏览器共享文件目录，方便在局域网共享文件

**17、zip 函数实现字典键值对互换**

>>> lang = {"python":".py", "java":".java"}

>>> dict(zip(lang.values(), lang.keys()))

{'.java': 'java', '.py': 'python'}

**18、查找列表中出现次数最多的数字**

test = [1, 2, 3, 4, 2, 2, 3, 1, 4, 4, 4, 5]

>>> max(set(test), key=test.count)

4

**19、使用 slots 节省内存**

class MyClass(object):

def \_\_init\_\_(self, name, identifier):

self.name = name

self.identifier = identifier

self.set\_up()

print(sys.getsizeof(MyClass))

class MyClass(object):

\_\_slots\_\_ = ['name', 'identifier']

def \_\_init\_\_(self, name, identifier):

self.name = name

self.identifier = identifier

self.set\_up()

print(sys.getsizeof(MyClass))

# In Python 3.5

# 1-> 1016

# 2-> 888

**20、扩展列表**

>>> i = ['a','b','c']

>>> i.extend(['e','f','g'])

>>> i

['a', 'b', 'c', 'e', 'f', 'g']

>>>

**21、列表负数索引**

>>> a = [ 1, 2, 3]

>>> a[-1]

3

**22、列表切片**

>>> a = [0,1,2,3,4,5,6,7,8,9]

>>> a[3:6] # 第3个到第6个之间的元素

[3, 4, 5]

>>> a[:5] # 前5个元素

[0, 1, 2, 3, 4]

>>> a[5:] # 后5个元素

[5, 6, 7, 8, 9]

>>> a[::] # 所有元素（拷贝列表）

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

>>> a[::2] # 偶数项

[0, 2, 4, 6, 8]

>>> a[1::2] # 奇数项

[1, 3, 5, 7, 9]

>>> a[::-1] # 反转列表

[9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

**23、二维数组变一维数组**

import itertools

>>> a = [[1, 2], [3, 4], [5, 6]]

>>> i = itertools.chain(\*a)

>>> list(i)

[1, 2, 3, 4, 5, 6]

**24、有索引的迭代**

>>> a = ['Merry', 'Christmas ', 'Day']

>>> for i, x in enumerate(a):

... print '{}: {}'.format(i, x)

...

0: Merry

1: Christmas

2: Day

**25、列表推导式**

>>> le = [x\*2 for x in range(10)]

>>> le # 每个数乘以2

[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]

>>> le = [x for x in range(10) if x%2 == 0]

>>> le # 获取偶数项

[0, 2, 4, 6, 8]

**26、生成器表达式**

>>> ge = (x\*2 for x in range(10))

>>> ge

<generator object <genexpr> at 0x01948A50>

>>> next(ge)

0

>>> next(ge)

2

>>> next(ge)

4

...

>>> next(ge)

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

StopIteration

**27、集合推导式**

Python

>>> nums = {n\*\*2 for n in range(10)}

>>> nums

{0, 1, 64, 4, 36, 9, 16, 49, 81, 25}

**28、判断key是否存在字典中**

>>> d = {"1":"a"}

>>> d['2']

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

KeyError: '2'

>>> '1' in d

True

>>> d['1']

'a'

>>> d.get("1")

'a'

>>> d.get("2")

>>>

**29、装饰器**

from functools import wraps

def tags(tag\_name):

def tags\_decorator(func):

@wraps(func)

def func\_wrapper(name):

return "<{0}>{1}</{0}>".format(tag\_name, func(name))

return func\_wrapper

return tags\_decorator

@tags("p")

def get\_text(name):

"""returns some text"""

return "Hello " + name

print(get\_text("Python"))

>>><p>Hello Python</p>

**30、字典子集**

>>> def sub\_dicts(d, keys):

... return {k:v for k, v in d.items() if k in keys}

...

>>> sub\_dicts({1:"a", 2:"b", 3:"c"}, [1,2])

{1: 'a', 2: 'b'}

**31、反转字典**

>>> d = {'a': 1, 'b': 2, 'c': 3, 'd': 4}

>>>

>>> zip(d.values(), d.keys())

<zip object at 0x019136E8>

>>> z = zip(d.values(), d.keys())

>>> dict(z)

{1: 'a', 2: 'b', 3: 'c', 4: 'd'}

**32、具名元组**

>>> from collections import namedtuple

>>> Point = namedtuple("Point", "x,y")

>>> p = Point(x=1, y=2)

>>> p.x

1

>>> p[0]

1

>>> p.y

2

>>> p[1]

2

**33、设置字典默认值**

>>> d = dict()

>>> if 'a' not in d:

... d['a'] = []

...

>>> d['a'].append(1)

>>> d

{'a': [1]}

>>> d.setdefault('b',[]).append(2)

>>> d

{'a': [1], 'b': [2]}

>>>

**34、有序字典**

>>> d = dict((str(x), x) for x in range(10))

>>> d.keys() # key 无序

dict\_keys(['0', '1', '5', '9', '4', '6', '7', '8', '2', '3'])

>>> from collections import OrderedDict

>>> m = OrderedDict((str(x), x) for x in range(10))

>>> m.keys() # key 按照插入的顺序排列

odict\_keys(['0', '1', '2', '3', '4', '5', '6', '7', '8', '9'])

**35、列表中最大最小的前n个数**

>>> import heapq

a = [51, 95, 14, 65, 86, 35, 85, 32, 8, 98]

>>> heapq.nlargest(5,a)

[98, 95, 86, 85, 65]

>>> heapq.nsmallest(5,a)

[8, 14, 32, 35, 51]

>>>

**36、打开文件**

>>> with open('foo.txt', 'w') as f:

... f.write("hello")

...

**37、两个列表组合成字典**

list\_1 = ["One","Two","Three"]

list\_2 = [1,2,3]

dictionary = dict(zip(list\_1, list\_2))

print(dictionary)

**38、去除列表中重复元素**

my\_list = [1,4,1,8,2,8,4,5]

my\_list = list(set(my\_list))

print(my\_list)

**39、打印日历**

import calendar

>>> print(calendar.month(2021, 1))

January 2021

Mo Tu We Th Fr Sa Su

1 2 3

4 5 6 7 8 9 10

11 12 13 14 15 16 17

18 19 20 21 22 23 24

25 26 27 28 29 30 31

**40、匿名函数**

def add(a, b):

return a+b

等价于

>>> add = lambda a,b:a+b

>>> add(1,2)

3