\*args 和 \*\*kwargs

这两个魔法变量并不是一定要写成这样，星号是必须的，

也可以写成\*var,\*\*vars。

\*args 是用来发送一个非键值对的可变数量的参数列表给一个函数.

def test\_var\_args(f\_arg, \*argv):

print("first normal arg:", f\_arg)

for arg in argv:

print("another arg through \*argv:", arg)

test\_var\_args('yasoob', 'python', 'eggs', 'test')

\*\*kwargs 的用法：处理键值对

def test(arg1,arg2,arg3):

print("arg1",arg1)

print("arg2",arg2)

print("arg3",arg3)

args=("tw",3,5)

test(\*args)

生成器中的next（），iter()方法

def multiply(x):

return (x\*x)

def add(x):

return (x+x)

funcs=[multiply,add]

for i in range(5):

value=map(lambda x:x(i),funcs)

print(list(value))

my\_list = [1, 2, 3]

dir(my\_list)

# Output: ['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_delitem\_\_',

# '\_\_delslice\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_',

# '\_\_getitem\_\_', '\_\_getslice\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_iadd\_\_', '\_\_imul\_\_',

# '\_\_init\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mul\_\_', '\_\_ne\_\_',

# '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_reversed\_\_', '\_\_rmul\_\_',

# '\_\_setattr\_\_', '\_\_setitem\_\_', '\_\_setslice\_\_', '\_\_sizeof\_\_', '\_\_str\_\_',

# '\_\_subclasshook\_\_', 'append', 'count', 'extend', 'index', 'insert', 'pop',

# 'remove', 'reverse', 'sort']

lambda 参数:操作(参数)

open的第一个参数是文件名。第二个(mode 打开模式)决定了这个文件如何被打开。

+

* 如果你想读取文件，传入r
* 如果你想读取并写入文件，传入r+
* 如果你想覆盖写入文件，传入w
* 如果你想在文件末尾附加内容，传入a

协程：

def grep(pattern):

print("Searching for ",pattern)

while True:

line=(yield)

if pattern in line:

print(line)

search=grep('coroutine')

next(search)

search.send("I LOve you coroutine ")