# 二 Python函数

## 1 公共方法



## 2 内置函数



注意：

cmp在比较字典数据时，先比较键，再比较值，返回值有 -1 0 1 三种。

len在操作字典的时候返回的是键值对个数

del有两种用法，一种是 del加空格，一种是del(item)

## 3 函数简介

|  |
| --- |
| *#定义函数* **def** printInfo():  print(**'人生苦短，我用Python'**) *#调用函数* printInfo()  *#带参数的函数* **def** add(a,b):  print(a+b) add(2,3)  *#带返回值的函数* **def** addPlus(a,b):  **return** a+b |

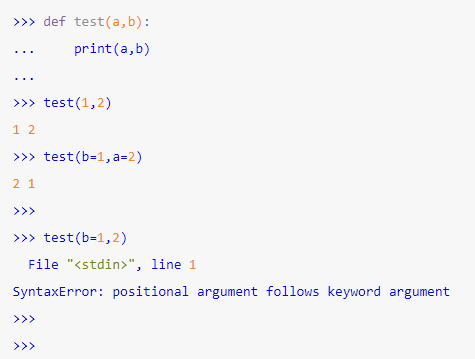


注意：Python的函数不能同名！

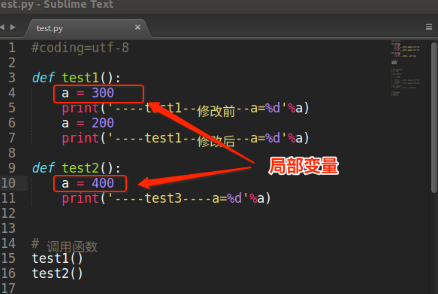
## 4 函数文档说明

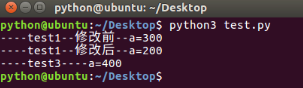


## 5 参数顺序



## 6 局部变量

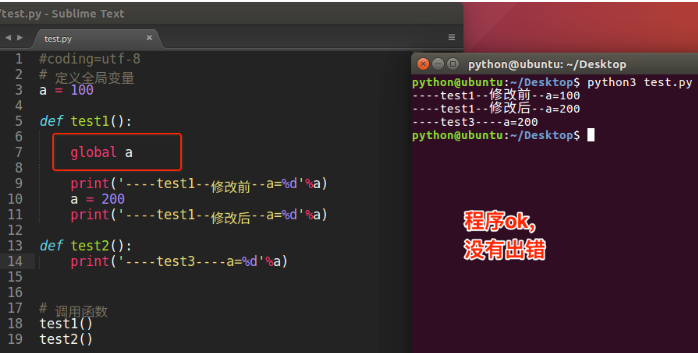




## 7 全局变量



修改全局变量



总结：

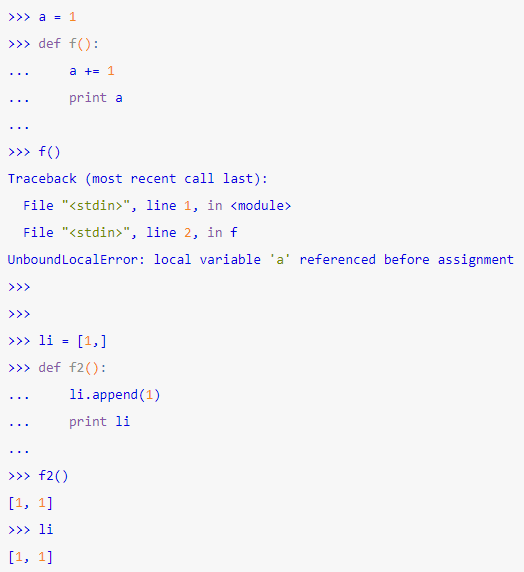
在函数外边定义的变量叫做全局变量

全局变量能够在所有的函数中进行访问

如果在函数中修改全局变量，那么就需要使用global进行声明，否则出错

如果全局变量的名字和局部变量的名字相同，那么使用的是局部变量的

## 8 可变类型全局变量



在函数中不使用global声明全局变量时不能修改全局变量的本质是不能修改全局变量的指向，即不能将全局变量指向新的数据。

对于不可变类型的全局变量来说，因其指向的数据不能修改，所以不使用global时无法修改全局变量。

对于可变类型的全局变量来说，因其指向的数据可以修改，所以不使用global时也可修改全局变量。

## 9 参数缺省、不定长、引用传参

### 9.1缺省参数

|  |
| --- |
| *#缺省参数的值如果没有传入，则被认为是默认值，下例会打印偶人的age，如果没有age被传入* **def** printInfo(name,age = **'35'**):  print(**'名字是'** + name)  print(**'年龄是'** + age) printInfo(**'lisi'**) printInfo(age=**'30'**,name=**'zs'**) *#注意：带有默认值的参数，一定要位于参数列表的最后，否则会报错* |

### 9.2 不定长参数

有时可能需要一个函数能处理比当初声明时更多的参数。这些参数叫做不定长参数，声明时不会命名。

def functionname([formal\_args,] \*args, \*\*kwargs):

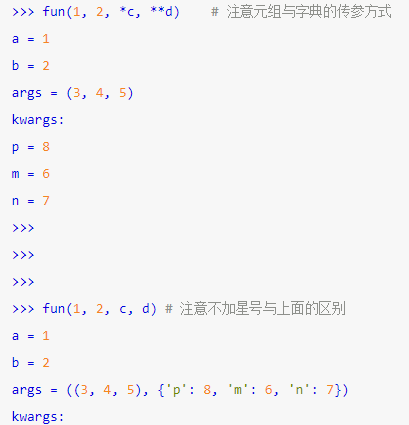
"函数\_文档字符串"

function\_suite

return [expression]

加了星号（\*）的变量args会存放所有未命名的变量参数，args为元组；而加\*\*的变量kwargs会存放命名参数，即形如key=value的参数， kwargs为字典。



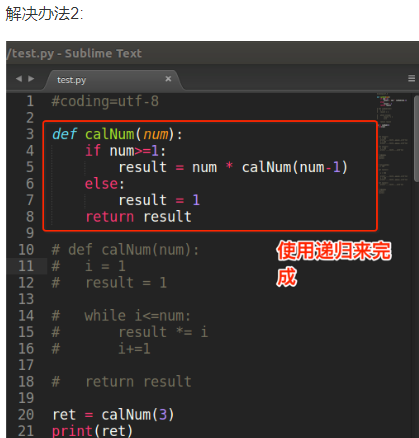
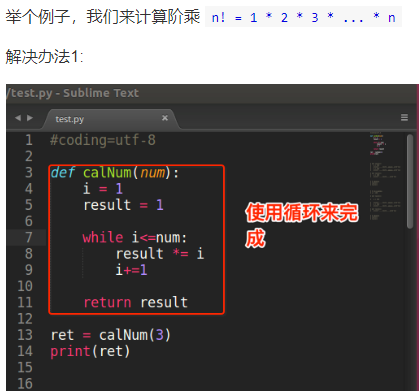


### 9.3 引用传参



## 10 递归函数

如果一个函数在内部不调用其它的函数，而是自己本身的话，这个函数就是递归函数。



## 11 匿名函数

用lambda关键词能创建小型匿名函数。这种函数得名于省略了用def声明函数的标准步骤。

lambda函数的语法只包含一个语句，如下：

lambda [arg1 [,arg2,.....argn]]:expression

|  |
| --- |
| sum = **lambda** arg1,arg2: arg1 + arg2 print(sum(10,20)) *#Lambda函数能接收任何数量的参数但只能返回一个表达式的值 #匿名函数不能直接调用print，因为lambda需要一个表达式* |





# 三 Python文件操作

## 1 文件打开与关闭

|  |
| --- |
| *#有打开就有关闭* f = open(**'1.txt'**,**'r'**) *#打开，如果是 r 方式，r可以省略不写* print(f.read()) *#读取，参数可以写入一个数字，表示读取多长* print(f.read()) *#输出空白：因为read是一次性读完，再读没有可以读的 #f.write('hi') #写入 这里由于权限问题会报错* |



## 2 readlines 与 readline

就像read没有参数时一样，readlines可以按照行的方式把整个文件中的内容进行一次性读取，并且返回的是一个列表，其中每一行的数据为一个元素。

即read读取的数据直接一次性展开显示，而readlines会按照行显示一个列表出来。

同样也有readline方法，每次读取只读取一行！

一般读取建议都要带个数值参数，因为如果数据太大，读取会导致程序崩溃。

## 3 案例-文件复制

|  |
| --- |
| *#coding=utf-8* oldFileName = input(**"请输入要拷贝的文件名字:"**)  oldFile = open(oldFileName,**'r'**)  *# 如果打开文件* **if** oldFile:   *# 提取文件的后缀* fileFlagNum = oldFileName.rfind(**'.'**)  **if** fileFlagNum > 0:  fileFlag = oldFileName[fileFlagNum:]   *# 组织新的文件名字* newFileName = oldFileName[:fileFlagNum] + **'[复件]'** + fileFlag   *# 创建新文件* newFile = open(newFileName, **'w'**)   *# 把旧文件中的数据，一行一行的进行复制到新文件中* **for** lineContent **in** oldFile.readlines():  newFile.write(lineContent)   *# 关闭文件* oldFile.close()  newFile.close() |

## 4 文件定位

|  |
| --- |
| f = open(**'1.txt'**,**'r'**) str = f.read(3) position = f.tell() *#tell函数获取当前位置* print(str) print(position) |

如果在读写文件的过程中，需要从另外一个位置进行操作的话，可以使用

seek(offset, from)

offset:偏移量

from:方向

0:表示文件开头

1:表示当前位置

2:表示文件末尾

|  |
| --- |
| f = open(**'1.txt'**,**'r'**) f.read(5) position = f.tell() print(**'当前位置是：%d'**%position ) position = f.seek(3,0) print(**'当前位置是：%d'**%position ) *#position = f.seek(-3,2) 这里的意思是 从文件末尾开始倒数，离末尾3个长度处 但是python3不支持负数了* |

## 5 文件重命名与删除

os模块中的rename()可以完成对文件的重命名操作

rename(需要修改的文件名, 新的文件名)

import os

os.rename("毕业论文.txt", "毕业论文-最终版.txt")

## 6 文件夹操作

|  |
| --- |
| **import** os os.mkdir(**"张三"**) *#创建文件* os.getcwd() *#获取当前目录* os.chdir(**"../"**) *#改变默认目录* os.listdir(**"./"**) *#获取目录列表* os.rmdir(**"张三"**) *#删除文件夹* |

示例代码：批量删除

#coding=utf-8

# 批量在文件名前加前缀

import os

funFlag = 1 # 1表示添加标志 2表示删除标志

folderName = './renameDir/'

# 获取指定路径的所有文件名字

dirList = os.listdir(folderName)

# 遍历输出所有文件名字

for name in dirList:

print name

if funFlag == 1:

newName = '[东哥出品]-' + name

elif funFlag == 2:

num = len('[东哥出品]-')

newName = name[num:]

print newName

os.rename(folderName+name, folderName+newName)