函数基础

什么是函数?
函数就是完成特定功能的一些语句的组合 这组语句可以作为一个单位使用，它不止一次的在程序中运行， 函数可以计算出一个返回值， 以函数的形式去编写一个操作， 可以使函数成为一个广泛应用的工具 不同的函数完成不同的功能 函数与函数之间的关系就是逻辑

函数可以通过传入的参数(也可以不传)计算出一个返回值(通过return),通过每次传入不同的参数 得到不同的返回值 函数是为实现代码最大程度的重用和最小化代码冗余最基本的程序结构

为什么使用函数
1. 最大化代码重用和最小化代码冗余

2. 流程的分解
函数将一个完整的任务分割为不同的部件， 将一个大任务分解成为一个小的任务，要比一次完成整个流程要容易的多

函数相关的语句和表达式

```
def         #定义函数
func()      #调用函数
return      #函数的返回值 没有return语句的函数返回None
global      #将函数内部的变量指定为全局变量(后面介绍变量作用域时细说)
nolocal     #应用在嵌套函数中 python3中才有
yield       #生成器
lambda      #定义匿名函数
```

在本节我们使用Python来进行函数的编写 我们编写的函数跟内置函数一样 通过表达式进行调用 传入一些值 返回结果

注意
1. 定义函数 通常使用def关键字

2. lambda 关键字也可以创建函数（匿名函数） 这个功能允许我们把函数定义到def 不能工作的地方 举例map()
3. return 将函数的结果返回给函数的调用者 当函数被调用时 只要遇到了return 则代码就认为函数的工作结束了 没有return 函数将自动返回一个none
4. global声明了一个模块级的变量并被赋值 默认情况下所有在函数中被赋值的变量是这个函数的本地变量 仅仅在这个函数的运行过程中存在 通过global可以将函数的本地变量 变成全局变量

def 语句将创建一个函数对象并将其赋值给一个变量名

def 函数名（arg1, arg2, args3,..）:
    coding….
函数的参数数量是0个以上 这里的参数是定义函数时候所需要传入的参数 这是形参

函数名(x, y, z) #调用函数在() 传入相应参数 （实际参数）
参数传入之后相当于 (arg1 = x, arg2 =y, arg3 = z)

代码示例 一个简单的加法函数

```
a = 100
b = 200
c = a + b
print c

def add(a, b):
    c = a + b
    print c

a = 100
def fun():
    if 1:
        print 'good'
        print a
```

函数的调用##
在程序中通过在函数名后增加括号来进行调用 括号中可以包含一个或多个对象参数 这些参数将会传递给函数头部的参数名

```
add(300, 400)
fun()
```

代码示例 2

寻找两个序列的交集

```
a = [1, 2, 3, 4]
b = [2, 3, 4, 5]

res = []
for j in a:
    if j in b:
        res.append(j)
print res


def intersect(l1, l2):
    res = []
    for j in l1:
        if j in l2:
            res.append(j)
    return res
print intersect(a, b)
```