# 函数

## 函数

函数是组织好的，可重复使用的，用来实现单一，或相关联功能的代码段。

函数能提高应用的模块性，和代码的重复利用率。

编写Python代码，可以直接调用内建函数，比如print()，也可以调用其他模块的函数。还可以自己创建函数，这被叫做**用户自定义函数**。

### 1.1 自定义函数

如何定义一个函数：

* 函数代码块以 **def** 关键词开头，后接函数标识符名称和圆括号**()**。
* 任何传入参数和自变量必须放在圆括号中间。圆括号之间可以用于定义参数。
* 函数的第一行语句可以选择性地使用文档字符串—用于存放函数说明或调用示例。
* 函数内容以冒号起始，并且缩进。
* **return [表达式]** 结束函数，选择性地返回一个值给调用方。不带表达式的return相当于返回 None。

**语法：**

def functionname( parameters ):

"函数\_文档字符串"

…

return [expression]

**参数可以没有，可以是1个参数或多个参数。如果没有参数，也需要带（），多个参数中间用逗号隔开。**

实例：

import math

def getArea(radius):

"根据半径获得面积"

area = math.pi \* radius \* radius

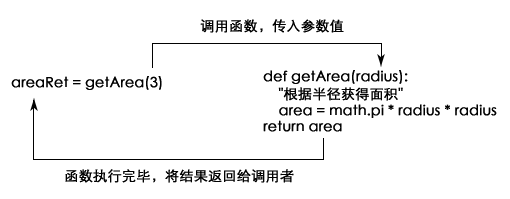
return area

利用print(getArea.\_\_doc\_\_)可以打印函数的文档字符串。

文档字符串也可以是多行。

调用函数

>>>getArea(3)



多次调用函数：

>>>2 \* (getArea(2.3) + getArea(3) + getArea(5))

如果不利用函数，代码就会很繁琐：

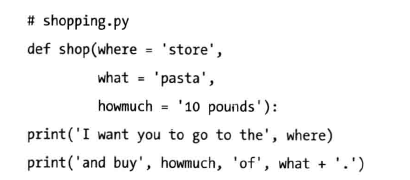
>>>2 \* (math.pi \* 2.3 \* 2.3 + math.pi \* 3 \* 3 + math.pi \* 5 \* 5))

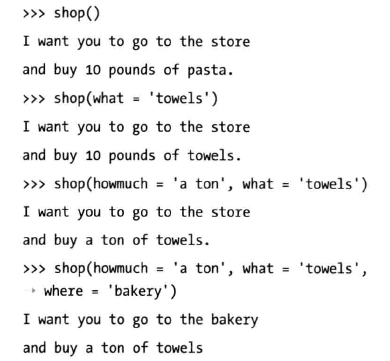
阅读代码也不直观。

### 1.2 函数参数默认值



### 1.3 关键字参数





### 1.4 不定长参数

# multiParam.py

# 可写函数说明

def printinfo( arg1, \*vartuple ):

"打印任何传入的参数"

print("输出: ")

print(arg1)

for var in vartuple:

print(var)

return

# 调用printinfo 函数

printinfo( 50 );

printinfo( 70, 80, 90 );

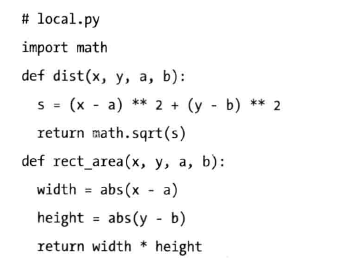
输出结果：

输出:  
50  
输出:  
70  
80  
90

## 将函数封装在模块里

将getArea封装在myMath模块里。这样，其它python程序就可以利用import语句更方便的调用。

## 局部变量、全局变量



S, width, height以及函数的参数都是局部变量。

理解全局变量，局部变量（包括参数）例子：

# var.py

def add(x, y):

z = x + y

return z

m = 10

n = 20

ret = add(m, n)

print(m)

print(n)

print(x)

print(z)

**全局变量和局部变量的名称没有直接关系**

上面的代码可以将m和n换成x和y。这样，程序就存在全局变量x，y以及add函数里的局部变量x, y，不会存在冲突。但实际编码最好区分开来。

### 1.3 访问全局变量 要访问全局变量，必须使用global语句声明一下。

# global\_visit.py

name = ‘tom’

def changeName(newName):

global name

name = newName

print(name)

changeName(‘tom2’)

print(name)

global name表示函数内部下面的代码块访问name的地方是访问全局变量name（包括读取和修改）