## 第七章 函数 章节时长：26min

第一节 数学函数 3min

第二节 函数简介+定义一个函数 5min

第三节 参数 4min

第四节 关键字参数 5min

第五节 返回语句 4min

第六节 创建可重用函数 5min

## 第一节 数学函数

1. round函数：

（1）返回值：将参数四舍五入后的结果，是一个整数

（2）举例：

**x = 2.9**

**print(round(x))**

* 输出结果：3

2. abs函数：

（1）返回值：将参数取绝对值后的结果，是一个非负数  
（2）举例：

**print(abs(-2.9))**

* 输出结果：2.9

**【模块 module】**

（1）概念：

- 在Python中的模块是一个包含可重用性代码的独立文件

- 我们使用模块将代码组织成不同的文件

- 在Python中我们有一个数学模块，里面包含了很多可重用性的可用于数学计算的函数

**【导入math模块】**

（1）格式（在最上方）：

**import math** # 都是小写形式

（2）使用点运算符调用数学函数：

**math.函数名**

* 输入点运算符之后，模块中的数学函数列表会自动显示，使得调用更方便
* 导入math模块后才能使用点运算符调用数学函数

3. ceil方法：

（1）返回值：参数的上限，即不小于该数字的最小整数

（2）举例：

**print(math.ceil(2.9))**

- 输出结果：3

4. floor方法：

（1）返回值：参数的下舍整数，即不大于该数字的最大整数

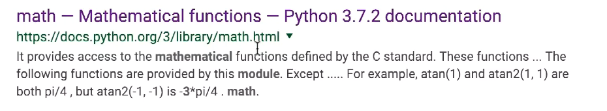
（2）举例：

**print(math.floor(2.9))**

* 输出结果：2

【怎样自己学习更多的math模块里的函数】

* 打开浏览器，搜索“”python 3 math module



* 点开该文档

## 第二节 函数简介+定义一个函数

1. 函数的概念：

随着程序的增长（chang），我们需要一个更好的方法来组织我们的代码，将它们拆分成更小、更易管理、更易维护的代码块，这些代码块就是函数

2. 函数的实质：

可以将函数视为一个盛放代码的容器，每个函数里的代码都能完成一个特定的任务

3. 定义一个函数

（1）思路：当有部分语句需要在其他程序中、或本程序的其他位置重复使用，可以用函数来“盛放”这些语句，形成一个能完成特定功能的代码块

（2）格式：

**def 函数名（）：** # 该函数无参数

**代码块**

（3）举例：

def greet\_user(): # 用一个末尾的冒号表示：我们定义了一个代码块  
 print('Hi there!') # 在这个缩进水平下的语句——属于这个函数  
 print('Welcome aboard')

4. 调用一个函数（无参数）：

- 格式：

**函数名（）**

5. 关于格式的注意事项：

（1）缩进：需要使用缩进来表示一个语句属于某个函数

（2）换行：在定义完一个函数后空两行会更好

（3）顺序：必须在定义一个函数之后才能调用它

## 第三节 参数

1. 作用：将信息传递给函数

2. 实质：可以在函数内部作为局部变量使用

3. 定义一个有参数的函数：

（1）格式：

**def 函数名（参数名）：** # 该函数有参数

**代码块**

（2）举例：

def greet\_user(name):   
 print(f'Hi {name}!')   
 print('Welcome aboard')

4. 调用一个有参数的函数：

（1）格式：

**函数名（待传递的值）**

（2）举例：

greet\_user(“John”) # 调用函数并传递一个值，name参数将被设置成“John”

5. 注意：当一个函数有参数时，我们调用该函数时就需要为它传递相应的值

【形参和实参的区别】

* 形参parameter：定义在函数中的占位符（place holder），用于接收信息

比如：def greet\_user(name): 里面的name，是形参

* 实参argument：我们调用函数时实际向函数传递的信息

比如：greet\_user(“John”)里面的“John”，是实参

6. 多个参数的函数

（1）定义：

**def 函数名(参数名1, 参数名2，参数名3，……):**

**代码块**

（2）调用：

**函数名（值1，值2，值3，……）**

## 第四节 关键字参数

1. 位置参数：

（1）调用一个带有参数的函数时，必须向其传递相应的参数，否则会报错

（2）在向函数传递形参时，参数的位置和顺序是很重要的，需要实参与形参的位置一一对应，这样的参数可称为位置参数

2. 关键字参数：

（1）概念：其位置和顺序不重要的参数，称为关键字参数

（2）特点：调用函数传参时可以不按照定义函数时参数列表的顺序

3. 关键字参数的格式：

def 函数名（参数名1，参数名2）： # 定义函数处

代码块

**函数名（参数名2=值2，参数名1=值1）** # 调用函数处

- 举例：

greet\_user(last\_name="Smith", first\_name="John")

4. 使用情况：

大部分时候我们都是使用位置参数，但是使用关键字参数可以帮助我们提升代码的可读性

- 如：传递数值参数，使用关键字参数可以帮助我们理解传递的每个数值的含义

- 举例：

cal\_cost（total=50，shipping=5，discount=0.1） # 一个计算订单总成本的函数

5. 注意：

在向函数传参的时候，若要同时使用关键字参数和位置参数，关键字参数总是要放在位置参数后面

## 第五节 return语句

1. 作用：将函数里的值传递回调用函数处

2. 格式：

**return 值**  # 第一种

**return 变量名** # 第二种

**return 表达式** # 第三种

- 举例：

def square(number):   
 return number \* number

3. 无返回值的情况：

若函数没有return语句，函数会默认返回一个None值

## 第六节 创建可重用函数

1. 函数命名原则：

每一个函数需要负责完成一个任务，并且只能负责这个任务；这个任务应该用函数名称来解释清楚

2. 创建可重用函数的原因：

某些代码块实现的功能，可能需要应用到其他不同的程序中，而这些代码没有必要重复写

3. 具体实现：见代码笔记