# 第四阶段 Python数据分析

# Python 基础5

## 网校相关视频

Python语言基础

第二章Python初级基础知识

## 明确本次课知识点，明确重点难点

### 【知识点目标】

* Python函数
* Python面向对象

### 【重点】

* Python函数
* Python面向对象

### 【难点】

* 函数的参数传递
* 变量的作用域
* 匿名函数
* 面向对象：封装和继承

## 复习巩固作业讲解

* Python程序流程控制之循环结构

## 本次课程任务讲解

### 【知识点1】Python函数

1. 函数的定义和使用
   1. 什么是函数

函数是将一些语句集合在一起的程序结构，是组织好的，可重复使用的，用来实现某些功能的代码段。

* 1. 内置函数与自定义函数

Python函数分为内置函数和自定义函数。

内置函数：Python已经定义好的函数，如print()

自定义函数：用户自己定义的函数。

* 1. 函数的定义

def <函数名>(arg1,arg2,...argN):

<statements>

return <expression>

arg1,arg2,...argN是函数的参数（非必须），但即使函数没有参数，()也不可省略

statements是函数中要执行的语句，必须缩进

return是结束函数并返回表达式的值给调用函数，非必须

* 1. 函数的调用

无返回值的调用：函数名(args)

有返回值的调用：变量 = 函数名(args)

函数要先定义再调用

1. 参数传递
   1. 函数的参数
      1. 形式参数

定义函数时，只是建立函数对象，并未运行，函数的参数，也并未赋值，此时参数是空的，称为“形式参数”

* + 1. 实际参数

调用函数时，会将一些对象传递到函数内部，这些传进函数的参数 称为“实际参数”，并与形式参数绑定，之后函数内部才能对形式 参数进行存取，函数才会进行操作。

* 1. 值传递

值传递也称为不可变类型，如整数、字符串、元组。如myfunc（a），传递的只是a的值，没有影响a对象本身。在函数中生成了当前对象的一个镜像，在函数中修改的是镜像。

* 1. 引用传递

引用传递也称为可变类型：列表，字典属于引用传递，也称为传址（地址传递）。如 funcB（mylist），则是将 mylist 真正的传过去，修改后funcB外部的mylist也会受影响。

由于列表是引用传递，所以常以列表作为在整个程序中都可以进行使用的变量

1. 参数类型
   1. 必备参数

函数中不可省略的参数

调用时不给予函数该参数会报错

调用方式：

funcA(arg1,arg2)

funcA(arg1=表达式,arg2=表达式)

后一种调用方式参数的顺序可以变化：关键字调用

* 1. 默认参数

调用函数时，默认参数的值如果没有传入，则被认为是默认值。也称为缺省参数

缺省参数在声明函数时就要赋值

* 1. 不定长参数

有时候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，不定长参数在声明函数时不用命名，用\*和\*\*表示

def funcName(arg1,arg2...,\*args,\*\*kwargs):

用\*args收集到的参数会生成一个元组

用\*\*kwargs收集到的参数会生成一个字典，接受形如x=y结构的变量，x成为key,y成为value

\*args和\*\*kwargs接受的参数位置不能交换，否则会出错

1. 函数返回值
   1. return

函数利用return语句返回值给调用函数的对象，如果该语句不存在，则返回None(没有参数的return语句也返回None)

1. 变量作用域
   1. 什么是变量的作用域

变量能够起作用的范围

一个程序的所有的变量并不是在哪个位置都可以访问的。访问权限决定于这个变量是在哪里赋值的。变量的作用域决定了在哪一部分程序你可以访问哪个特定的变量名称。

* 1. 全局变量

在整个程序中都可以访问的变量，定义在函数外

* 1. 局部变量

在定义函数的语句块中可以访问的变量

局部变量在函数外不可访问

在函数外定义的全局变量在函数内实际上是不可访问的

* 1. 在函数内使用函数外定义的为全局变量

如果在函数内也定义了和全局变量相同的变量，这时函数内的同名变量是一个局部变量，和函数外的全局变量不同。

在函数中使用global关键字，可以在函数内改变全局变量的值

* 1. 变量作用域和传址/传值的区别

变量的作用域在不进行参数传递时也起作用

在实际的操作中，往往会利用列表在函数中传递时是传址这种方法，将列表作为一个伪全局变量来使用

1. 匿名函数
   1. 匿名函数概念

定义函数中，没有给定名称的函数就叫匿名函数

* 1. 匿名函数语法

lambda 参数列表：表达式

* 1. 匿名函数使用
     1. 无参数形式
     2. 有参数形式

### 【知识点2】Python面向对象

1. 常见的编程方式
   1. 面向过程

根据业务逻辑从上到下书写代码

* 1. 函数式

将相关的业务逻辑代码封装到程序块中，后续只需要调用即可

* 1. 面向对象

对函数进行分类和封装，做为对象的内置属性或方法，在调用时通过对象进行调用

1. 面向对象的基本概念

什么是面向对象

对象

对象有哪些特征、哪些行为、对象和对象之间有什么关系

对象的属性，对象的方法，**封装，继承**

**属性：对象的特征，用来描述对象（笔的材质、长短）**

**对属性可以做的操作：获取属性/设置属性**

**方法：对象的行为，用来处理对象可以实现的功能（笔有写的方法）,方法其实就是封装在对象内部的函数**

**pen.write(“hello”)**

1. 类与对象
   1. 什么是类

类是用来描述具有相同属性和方法的对象的集合

* 1. 什么是对象

对象是对类的实例化

* 1. 类和对象的关系

类是对象的类型，对象是类的实例，类是抽象概念，对象是具体实体

1. 类的声明和实例化
   1. 创建类

class语句

class 类名称(继承类):

类体

* 1. 实例化类

和函数的调用方式类似

asset1 = Asset()

* 1. 如何调用类
     1. 类属性的访问

用点语法进行属性的访问

变量 = 对象.属性 （获取类的属性）

对象.属性 = 表达式 （设置类的属性）

dict.keys()

* + 1. 类方法的调用

使用点语法进行方法的调用

对象.方法(参数)

* 1. 类的构造函数：创建类属性

构造函数是一个属于类的特殊方法，\_\_init\_\_(),前后都有两个下划线，类的构造函数。

每当用类名称实例化创建对象时，都会执行\_\_init\_\_()方法，其作用是初始化已经实例化的对象。

在\_\_init\_\_()方法中，可以设置类的属性，在初始化类的时候，需要为这些属性赋值

* 1. 封装：
     1. 在类中定义方法

通过设定自定义方法名称，在类中创建方法

* + 1. 调用类的方法

通过对象.方法名()来调用

* + 1. 属性私有化

将属性限制在类的内部进行访问，不暴露到类外面

通过在属性前加上\_\_（两个下划线）,将属性转为private属性

私有化属性的调用可以通过类中的方法（函数）来实现

* 1. 继承：子类继承父类的属性和方法

继承的类具有父类的属性和方法，也可以有自己的属性和方法

## 自主学习作业讲解

无

## 课程总结

#### Python中的自定义函数

Python面向对象

## 下次自主学习任务布置