Python Statistics Analysis Lecture Notes

Daniel

[Python Statistics Analysis Lecture Notes 1](#_Toc109400262)

[1 Python与环境配置 3](#_Toc109400263)

[1.1 Python语言 3](#_Toc109400264)

[1.2 Python语言的下载 3](#_Toc109400265)

[1.3 开发环境搭建——常用的IDE 4](#_Toc109400266)

[1.3-1 使用PyCharm开发Python 4](#_Toc109400267)

[1.3-2 使用VS code开发Python 4](#_Toc109400268)

[1.3-3 使用Jupyter Notebook开发Python 5](#_Toc109400269)

[1.3-4 简单粗暴：在命令行中开发Python 5](#_Toc109400270)

[1.4 数据科学的好帮手 Anaconda 5](#_Toc109400271)

[1.4-1 使用spyder开发Python 5](#_Toc109400272)

[1.5 常见问题 6](#_Toc109400273)

[1.5-1 多版本Python共存小技巧 6](#_Toc109400274)

[1.5-2 创建python虚拟环境，解决包冲突问题 6](#_Toc109400275)

[2 Python基础语法 6](#_Toc109400276)

[2.1 第一个Python程序：hello world 6](#_Toc109400277)

[2.1-1 小技巧：使用python手册 7](#_Toc109400278)

[2.2 来自手册的例子 7](#_Toc109400279)

[2.3 Python变量与数据类型 7](#_Toc109400280)

[2.3-1 数值型 8](#_Toc109400281)

[2.3-2 字符型 9](#_Toc109400282)

[2.3-3 数值型和字符型的转换 10](#_Toc109400283)

[2.4 Python数据结构 10](#_Toc109400284)

[2.4-1 列表 10](#_Toc109400285)

[2.4-2 元组 11](#_Toc109400286)

[2.4-3 集合 11](#_Toc109400287)

[2.4-4 字典 11](#_Toc109400288)

[2.5 Python判断结构 11](#_Toc109400289)

[2.5-1 单分支if语句 11](#_Toc109400290)

[2.5-2 双分支if语句 11](#_Toc109400291)

[2.5-3 match-case语句（Python 3.10引入） 11](#_Toc109400292)

[2.6 Python循环结构 11](#_Toc109400293)

[2.6-1 基于条件判断的while循环 11](#_Toc109400294)

[2.6-2 基于固定次数的for循环 12](#_Toc109400295)

[2.6-3 循环的控制 12](#_Toc109400296)

[2.7 python函数的使用与定义 12](#_Toc109400297)

[2.8 引用python库的四种方式 12](#_Toc109400298)

[2.9 Python文件操作 13](#_Toc109400299)

[2.9-1 文件的打开与关闭 13](#_Toc109400300)

[2.9-2 文件的读取和写入 13](#_Toc109400301)

[2.9-3 文件读取乱码的情况 13](#_Toc109400302)

[2.10 Python类编程 13](#_Toc109400303)

[2.10-1 类继承 13](#_Toc109400304)

[3 Python常用库 14](#_Toc109400305)

[3.1 Python库的安装 14](#_Toc109400306)

[3.2 数学math库 14](#_Toc109400307)

[3.3 随机random库 14](#_Toc109400308)

[3.4 面向数据清洗 14](#_Toc109400309)

[3.4-1 格式化数据处理的json库 14](#_Toc109400310)

[3.4-2 正则表达式re库 14](#_Toc109400311)

[3.5 用于爬虫的selenium和requests库 14](#_Toc109400312)

[3.6 用于科学计算的NumPy、SciPy库 14](#_Toc109400313)

[3.7 用于数据分析的Pandas库 15](#_Toc109400314)

[3.8 用于绘图的Matplotlib库 15](#_Toc109400315)

[4 Python统计学 15](#_Toc109400316)

[4.1 对样本的描述性统计 15](#_Toc109400317)

# Python与环境配置

## Python语言

Python语言是一门面向对象编程（Object-Oriented Programming, OOP）语言。Python语言是解释型的，源代码无需编译（compile）即可运行；C语言等其他面对对象编程语言需要对不同平台使用不同的编译器进行编译。

Python支持多个操作系统，在Windows、Linux、MacOS等系统上都有较好的支持。Python语言的源代码可以直接在不同系统上运行，实际应用中非常方便。

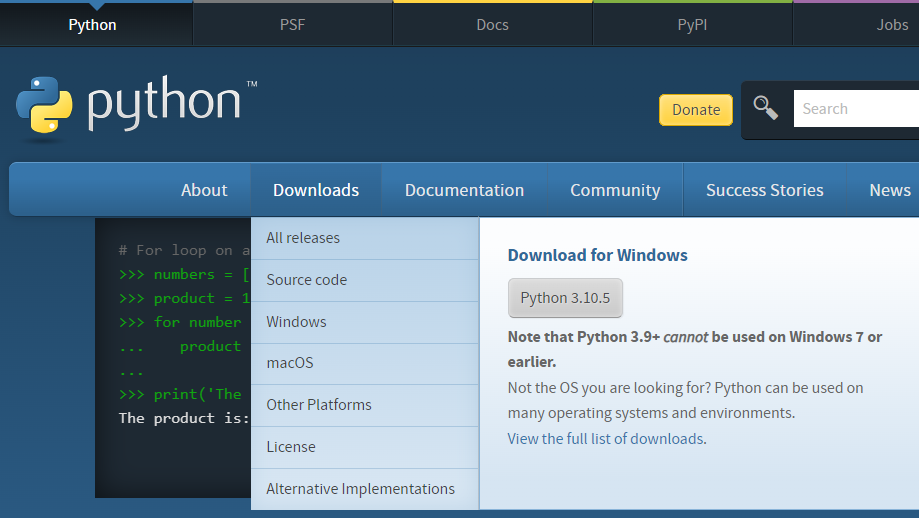
Python语言简便，第三方库非常多，应用非常广泛，在数据分析、人工智能领域具有很多应用。在互联网上，例如GitHub等网站提供了很多开源（open source）的项目，可供参考学习。

Python的官网是[www.python.org](http://www.python.org)。我们可以在Python官网上下载最新版本的Python，也可以查询到相关资料。

Python3是Python2的下一个版本，后者已经停止更新，二者语法存在着一定的差异。本课程将介绍目前使用更为广泛且支持度更好的Python3。

## Python语言的下载

打开Python官网，根据操作系统选择对应的安装包进行下载。Python提供了基于windows、



## 开发环境搭建——常用的IDE

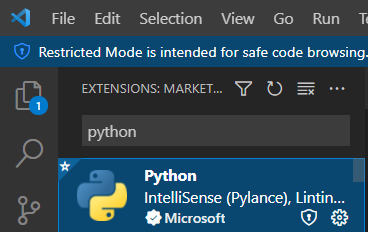
集成开发环境（Integrated development environment, IDE）

### 使用PyCharm开发Python

PyCharm是JetBrains公司开发的用于Python的集成开发环境。优点是在创建项目的时候，可以非常方便的使用pipenv虚拟环境。

### 使用VS code开发Python

Visual Studio code（VS code）是微软开发的一款轻量级的IDE。可以同时配置多个语言的开发环境。安装Python拓展后，对于打开的python文件，按下F5即可运行。



### 使用Jupyter Notebook开发Python

使用Jupyter Notebook开发的好处在于同一个文件中的代码分块，可以执行部分代码。此外，运行结果还可以保留。

在安装了Anaconda的情况下，无需再安装Jupyter Notebook，只需在运行中输入“jupyter notebook”命令即可执行。

如果是安装了Python，需要在命令窗口中输入“pip install jupyter”。具体可以参考3.1节的python库的安装。

### 简单粗暴：在命令行中开发Python

在windows系统“运行”中，输入cmd并回车，或打开linux系统的终端，键入“python”。如下图所示，我已经运行了在我电脑上的Python。接下来，就可以进行编程了。在这里的代码是随时编写、随时运行的，与IDE中开发有所不同，在需要简单编程的时候，可以考虑这种方法。



## 数据科学的好帮手 Anaconda

### 使用spyder开发Python

## 常见问题

### 多版本Python共存小技巧

计算机中可以同时安装多个版本的python。但在调用的时候，会从环境目录中寻找一个。如果多个python版本都是用的是同一个文件名，那么自然无法区分开来。

可以将每一个目录下的python.exe和script文件夹下的pip.exe文件分别复制一份在原目录。例如我的是python 3.9版本，就可以重命名为”python39”和“pip39”。同样的，python 3.10版本可以重命名为”python310”或”python3a”，可以依照自己的习惯来。

以后，就可以使用python39 xxx.py这样的命令，用指定的版本运行啦~

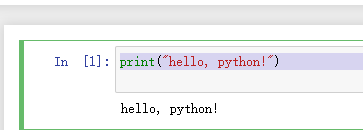
### 创建python虚拟环境，解决包冲突问题

# Python基础语法

## 第一个Python程序：hello world

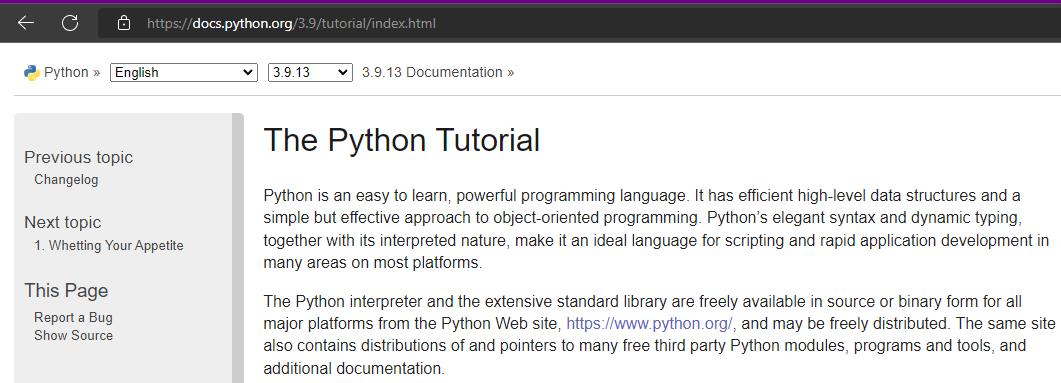
print("hello, python!")

运行结果



### 小技巧：使用python手册

使用官网提供的Python手册**（https://docs.python.org/3.9/tutorial/index.html）**可以获取一些使用指南。



## 来自手册的例子

*# this is the first comment*

spam = 1 *# and this is the second comment*

*# ... and now a third!*

text = "# This is not a comment because it's inside quotes."

Python注释语句：注释部分相当于做个笔记，说明代码的用途。这一部分不作为代码本体运行。有时也可以将部分需要调试的代码进行注释，这部分代码不运行。

单行注释，使用 # 开头。多行注释借用多行文本形式，”””多行注释””” 或者’’’多行注释’’’。

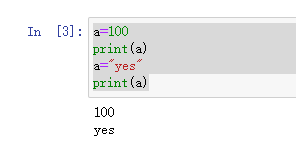
第四行是一个失败的注释例子，因为将其放置在引号中，作为一个文本变量的内容，并赋值给text。

## Python变量与数据类型

Python为变量提供多种数据类型：数值类、字符串等。

在Python3中，定义

Python语言对变量类型不敏感，可以为数值类变量赋值字符。



### 数值型

数值型包括：整数、布尔、浮点数、复数。

整数int：不同于其他语言，整数无大小上限。

布尔bool：bool表示逻辑型的变量，取值为True（1），False（0）。

浮点float：表示小数。由于二进制存储关系，在财务等要求精度的计算中，往往采用Decimal十进制存储的的小数用以计算。（思考：Python中0.1+0.2==0.3）

复数complex：虚数，在python中用j表示虚数单位i。例如3+4j。

#### 数值运算

基本运算：加减乘除

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 运算名 | | Python | 赋值给X | 简写 |
| 加 | add | X + Y | X = X + Y | X += Y |
| 减 | minus | X - Y | X = X – Y | X -= Y |
| 乘 | time | X \* Y | X = X \* Y | X \*= Y |
| 除 | divide | X / Y | X = X / Y | X /= Y |
| 整除 | X // Y | X = X // Y | X //= Y |
| 取余 | remainder | X % Y | X= X % Y | X %= Y |

扩展的基本运算

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 运算名 | | Python | 赋值给X | 简写 |
| 乘幂 | power | X \*\* Y | X = X \*\* Y | X \*\*= Y |
| 根号 | root | X \*\* (1/Y) | X = X \*\* (1/Y) | X \*\*= 1/Y |

位运算：且、或、异或

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 运算名 | | Python | 赋值给X | 简写 |
| 且 | and | X & Y | X = X & Y | X &= Y |
| 或 | or | X | Y | X = X | Y | X |= Y |
| 异或 | xor | X ^ Y | X = X ^ Y | X ^= Y |

#### 求解一元二次方程：使用math库进行数据计算

a,b,c=[float(x) for x in input().split()]

delta=b\*\*2-4\*a\*c

x1=(-b-delta\*\*0.5)/2/a

x2=(-b+delta\*\*0.5)/2/a

改用数学库：

import math

a,b,c=[float(x) for x in input().split()]

delta=b\*\*2-4\*a\*c

x1=(-b-math.sqrt(delta))/2/a

x2=(-b+math.sqrt(delta))/2/a

### 字符型

在python中，字符型可以用英文单引号或双引号标记。如 “字符串” 或 ‘字符串’。

多行字符串的实现：在单行字符串中使用\n换行符，或者使用三个单双引号进行标记。（多行字符串也可用作多行注释）。此外，如果希望字符串中包含引号，而使用引号会被当作字符串结束，那么可以使用\”或\’来标记。

其中\是用作转义字符，后面往往搭配一个字符。如果希望字符串中包含\，那么可以用两个\\来表示。

#### 字符串的运算

字符串可以进行拼接。两个字符串可以使用加法，合成一个新字符串。

字符串的“乘法”：字符串重复多次出现。

字符串的索引：正索引和副索引。

例如字符串，“MISAKAMIKOTO”

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 正索引 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 字符串 | M | I | S | A | K | A | M | I | K | O | T | O |
| 负索引 | -12 | -11 | -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 |

x[i:j]表示索引[i,j)字符。这里正负索引均可兼容。

>>> x[1:-1]

'isakamikot'

>>> x[1:11]

'isakamikot'

步长也是可选项，例如对字符串取反：

>>> x="MISAKAMIKOTO"

>>> x[::-1]

'OTOKIMAKASIM'

前面的两个冒号表示第一个参数和第二个参数都省略。

### 数值型和字符型的转换

使用int、float、str函数。

使用int函数：转换成整型数值变量。

使用float函数：转换成浮点数值变量。

使用complex(x,y=0)函数：转换成复数x+yj。

使用str函数：转换成字符型变量。

## Python数据结构

不同于C++等语言，Python的集合结构中没有要求变量具有相同的数据类型。这意味着，Python语言在编程中可以更加灵活。

### 列表

列表(list)是Python中的元素集合。可以使用list()函数，或[]创建一个列表。

### 元组

元组(tuple)也是Python中的元素集合。相比较于列表，元组中的元素是不可变的。通常用tuple()函数或者()来创建一个元组。当元组中元素只有一个时，要在后面加一个逗号。例如(test,)表示这是一个元组，否则会当作一个变量来处理。

### 集合

集合(set)也是Python中的元素集合。相比较于列表，集合中的元素是唯一的。通常用set()函数来创建一个集合。

### 字典

字典是Python中的键值对的集合。每一个键都是唯一的，每一个键都对应着一个值。通常用dict()函数或{}来创建一个字典。

## Python判断结构

### 单分支if语句

### 双分支if语句

### match-case语句（Python 3.10引入）

## Python循环结构

### 基于条件判断的while循环

### 基于固定次数的for循环

### 循环的控制

## python函数的使用与定义

我们初步使用了一些函数。通过“函数名(参数)”的方式调用函数，并将返回值打印出来。

函数的定义使用了def 函数名(参数):，其中返回值使用return语句返回。

我们也可以使用yield语句返回值，与return相同的是，此时函数都会停止运行。而在下一次调用的时候，return返回后的子程序在调用后会重新开始执行；yield语句返回后被再次调用，会从上一次yield语句后继续执行。

## 引用python库的四种方式

方法一：导入import 紧跟库名。调用库内函数，需要**“库名.函数名”**。这是最简单的方法，但在库名较长的时候，调用函数长度繁琐。

# case 1

import math

print(math.sqrt(2))

方法二：导入+别名。在导入后，加上as m，给数学库起别名m。调用库内函数，需要**“别名.函数名”**。由于起了别名，在库名字较长的时候，可以有效减少代码繁琐度。缺点在于不同的程序中，对同一个库可能有不同的别名。对于不同的库，起的别名要有区分度。

# case 2

import math as m

print(m.sqrt(2))

方法三：直接导入库中指定函数：from 库 import 函数名，此时函数可以直接使用，可直接**“函数名”**。此时，库中其他函数不可使用，例如sin。

# case 3

from math import sqrt

print(sqrt(2))

print(sin(30))

方法四：直接导入库中所有函数：from 库 import \*，此时库中所有函数都可以直接使用，可直接**“函数名”**。使用此方法时，需要注意函数重名的情况。

# case 4

from math import \*

print(sqrt(2))

print(sin(30))

## Python文件操作

### 文件的打开与关闭

通过open()函数创建一个文件操作对象。

### 文件的读取和写入

### 文件读取乱码的情况

## Python类编程

Python中创建一个类。

### 类继承

# Python常用库

## Python库的安装

使用pip install package-name命令安装。如果速度较慢，可以使用清华源进行安装

pip install -i https://pypi.tuna.tsinghua.edu.cn/simple package-name

## 数学math库

## 随机random库

random库提供了Python的随机计算

## 面向数据清洗

### 格式化数据处理的json库

### 正则表达式re库

## 用于爬虫的selenium和requests库

## 用于科学计算的NumPy、SciPy库

NumPy(Numerical Python) 是 Python 语言的一个扩展程序库，支持大量的维度数组与矩阵运算，此外也针对数组运算提供大量的数学函数库。NumPy 通常与 SciPy（Scientific Python）和 Matplotlib（绘图库）一起使用

## 用于数据分析的Pandas库

## 用于绘图的Matplotlib库

# Python统计学

## 对样本的描述性统计

# 用于数值计算的库

import numpy as np

import scipy as sp

# 设置浮点数打印精度（jupyter）

%precision 3

len(fish\_data) #样本容量

np.mean(fish\_data) #样本均值

np.var(fish\_data, ddof = 0) #样本方差 （MLE，有偏的，二阶中心距）

np.var(fish\_data, ddof = 1) #样本方差 （无偏）

np.std(fish\_data, ddof = 1) #样本标准差 （无偏）

np.amax(fish\_data) #样本最大值

np.amin(fish\_data) #样本最小值

np.median(fish\_data) #样本中位数

from scipy import stats #分位数

stats.scoreatpercentile(fish\_data\_3, 25) #0.25分位数 或 上0.75分位数

stats.scoreatpercentile(fish\_data\_3, 75) #0.75分位数 或 上0.25分位数