

# 实验课

## python安装与简单语法

何鸿荣

数据科学与计算机学院

中山大学

# 环境搭建

## ➤ Python版本及操作系统选择

- Python是跨平台的语言，因此脚本可以跨平台运行，然而不同的平台运行效率不一样，一般来说Linux下的速度会比Windows快，而且对于数据分析和挖掘任务。此外，在Linux下搭建Python环境相对来说容易一些，很多Linux发行版自带了Python程序，并且在Linux下更容易解决第三方库的依赖问题。当然，Linux的操作门槛较高，可以先在Windows熟悉，然后再考虑迁移到Linux下。

# 环境搭建

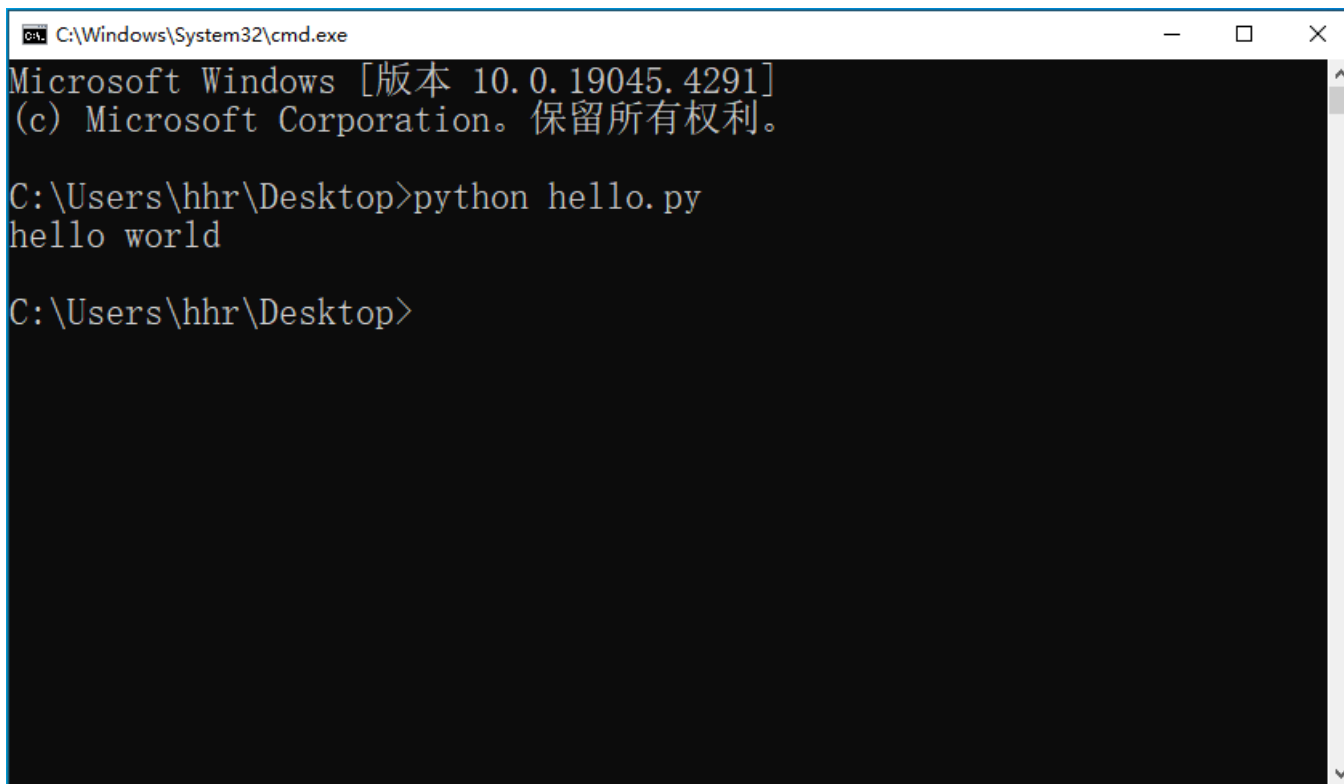
---

## ➤ Python的安装

- 在Windows下安装Python比较容易，直接到官方网站下载相应的msi安装包安装即可，和一般软件的安装无异，在此不赘述。安装包选择64位版本。
- Python的官网：<https://www.python.org/>

# 环境搭建

- 运行Python代码有两种方式，一种方式是启动Python，然后在命令窗口下直接输入相应的命令；另外就是将完整的代码写成.py脚本，如hello.py，然后通过python hello.py执行。



```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.19045.4291]
(c) Microsoft Corporation。保留所有权利。

C:\Users\hhr\Desktop>python hello.py
hello world

C:\Users\hhr\Desktop>
```

# 环境搭建

## ➤ 高效编辑器-VSCode

- 扩展商店安装python扩展



# 环境搭建

## ➤ 库的安装

- 修改配置源

- 在cmd输入

- ```
pip config set global.index-url https://mirrors.aliyun.com/pypi/simple/
```

- 使用pip install 安装库

```
(base) PS C:\Users\hhr\Desktop\作业\模式识别\hw1> pip config set global.index-url https://mirrors.aliyun.com/pypi/simple/
Writing to C:\Users\hhr\AppData\Roaming\pip\pip.ini
(base) PS C:\Users\hhr\Desktop\作业\模式识别\hw1> pip install numpy
Defaulting to user installation because normal site-packages is not writeable
Looking in indexes: https://mirrors.aliyun.com/pypi/simple/
```

# 环境搭建

---

- 可能会使用到的库
- **numpy, cv2**
- **pip install numpy**
- **pip install opencv-python**
- 库的使用
- **import numpy**

# 基本命令

可以把python当作一个方便的计算器来看待。

```
a = 2
```

```
a * 2
```

```
a ** 2
```

以上是Python几个基本的运算，第一个是赋值运算，第二是乘法，最后是一个是幂（即a的平方），这些基本上是所有编程语言通用的。

Python支持多重赋值：

```
a, b, c = 2, 3, 4
```

这句命令相当于：

```
a = 2
```

```
b = 3
```

```
c = 4
```



# 基本命令

## ➤ 字符串运算

- **Python支持对字符串的灵活操作:**

```
s = 'I like python'
```

```
s + ' very much' #将 s 与' very much'拼接, 得到'I like python very much'
```

```
s.split(' ') #将 s 以空格分割, 得到列表['I', 'like', 'python']
```

# 基本命令

## ➤ 判断

判断和循环是所有编程语言的基本命令，Python的判断语句与c++比较相似：

```
if 条件1:  
    语句2  
elif 条件3:  
    语句4  
else:  
    语句5
```

需要特别指出的是，Python一般不用花括号{}，也没有end语句，它是用缩进对齐作为语句的层次标记。同一层次的缩进量要一一对应，否则报错。

# 基本命令

## ➤ 循环与 in 的使用

**for 与 while 都属于循环的关键字。**

**其中while与c++的使用比较相似**

**for通常搭配 in 一起使用**

**in作为一个关键字，能够比较方便的取出列表中的内容。**

```
Sum = 0
for i in range(10):
    Sum += i
print(i)
```

# 基本命令

## ➤ 函数

**Python用def来自定义函数：**

```
def harris_corner_detection (img, window_size):  
    ...  
    return corners
```

**Python的函数返回值可以是各种形式，比如返回列表，甚至返回多个值**

:

```
def add2(x = 0, y = 0): #定义函数，同时定义参数的默认值  
    return [x+2, y+2] #返回值是一个列表  
def add3(x, y):  
    return x+3, y+3 #双重返回  
a, b = add3(1,2) #此时a=4,b=5
```

# 数据分析工具

- **NumPy**是一个用于进行科学计算的**Python**库。它提供了高性能的多维数组对象（**ndarray**），以及处理这些数组的数据。
- 数组切片是一种在**NumPy**中常用的技术，用于获取数组的子集。
- `array[start:stop:step]`
- 条件筛选可以选择在**numpy**数组中符合条件的数据。

```
import numpy as np

arr = [1,2,3,4,5,6,7]
arr = np.array(arr)

arrstep = arr[1:5:2] # [2,4]

arrgt4 = arr[arr>4] # [5,6,7]
```

# 数据分析工具

- **NumPy**是一个用于进行科学计算的**Python**库。它提供了高性能的多维数组对象（**ndarray**），以及处理这些数组的数据。
- 切片和条件筛选

| 函数                                    | 说明              | 函数                                        | 说明          |
|---------------------------------------|-----------------|-------------------------------------------|-------------|
| <code>np.array(list)</code>           | 从列表创建数组。        | <code>np.sum(array, axis)</code>          | 计算数组元素的和。   |
| <code>np.zeros(shape)</code>          | 创建指定形状的全零数组。    | <code>np.mean(array, axis)</code>         | 计算数组元素的平均值。 |
| <code>np.ones(shape)</code>           | 创建指定形状的全一数组。    | <code>np.max(array, axis)</code>          | 找到数组中的最大值。  |
| <code>ndarray.shape</code>            | 获取数组的形状         | <code>np.min(array, axis)</code>          | 找到数组中的最小值   |
| <code>np.argwhere(condition)</code>   | 返回满足条件的数组元素的索引。 | <code>np.concatenate(arrays, axis)</code> | 沿指定轴连接数组    |
| <code>numpy.linalg.norm(array)</code> | 计算L2距离          |                                           |             |

# 数据分析工具

- **cv2 (OpenCV)** 是一个广泛应用于计算机视觉领域的开源计算机视觉库，它提供了用于图像处理、计算机视觉和机器学习的各种函数和工具。

| 函数                                                                        | 说明                    |
|---------------------------------------------------------------------------|-----------------------|
| <code>cv2.imread(path, flag)</code>                                       | 从文件中读取图像。             |
| <code>cv2.imwrite(filename, image)</code>                                 | 将图像保存到文件中             |
| <code>cv2.cvtColor(image, code)</code>                                    | 将图像从一个颜色空间转换为另一个颜色空间。 |
| <code>cv2.resize(image, dsize)</code>                                     | 调整图像的大小。              |
| <code>cv2.Sobel(image, ddepth, dx, dy)</code>                             | 计算图像的梯度。              |
| <code>cv2.drawMatches(img1, keypoints1, img2, keypoints2, matches)</code> | 将两个图像中的关键点及其之间的匹配关系   |
| <code>cv2.warpPerspective(img, M, dsize)</code>                           | 对图像进行透视变换。            |

# 数据分析工具

- **cv2 (OpenCV)** 是一个广泛应用于计算机视觉领域的开源计算机视觉库，它提供了用于图像处理、计算机视觉和机器学习的各种函数和工具。

| 函数                                       | 说明             |
|------------------------------------------|----------------|
| <code>cv2.SIFT_create()</code>           | 创建一个SIFT特征检测器。 |
| <code>sift.compute(img, kp)</code>       | 计算灰度图中的关键点的特征  |
| <code>cv2.HOGDescriptor()</code>         | 创建一个HOG特征检测器   |
| <code>hog.compute(img, locations)</code> | 计算某个位置的hog特征   |