



Introduction to Python for Data Science

数据科学入门 – 使用Python程序语言

Aug 2020

Microsoft Reactor | Ryan Chung

```
led by player to  
s.load_image("kg.png")  
  
[self]:  
    initialize Dog object and create Text o  
g, self).__init__(image = Dog.image  
x = games.mouse.x  
bottom = games.sc  
  
re = games.Text(value = 0, size = 24  
top = 5, right = gam  
  
reen.add(self.score)  
1 = games.Text(value = 0, size = 24  
top = 5, left = gam
```



Ryan Chung

Instructor / DevelopIntelligence
Founder / MobileDev.TW

@ryanchung403 on WeChat
Ryan@MobileDev.TW





Reactor



developer.microsoft.com/reactor/
@MSFTReactor on Twitter

Data Science Workshop agenda 数据科学在线研讨会议程

Intro to Python

Python 程序语言入门

19:30	Welcome 开场
19:35	Introduction to Data Science 数据科学介绍
20:10	Arithmetic and numeric types 数值运算与数据类型
20:30	5-minute break 中场休息
20:35	Strings 字符串操作
20:50	Other data types 其他数据类型介绍
21:00	Event end 研讨会结束



制造业瑕疵检测



农业资源管理

人工智能应用广泛，取得资料是研究的第一步



医疗影像辨识

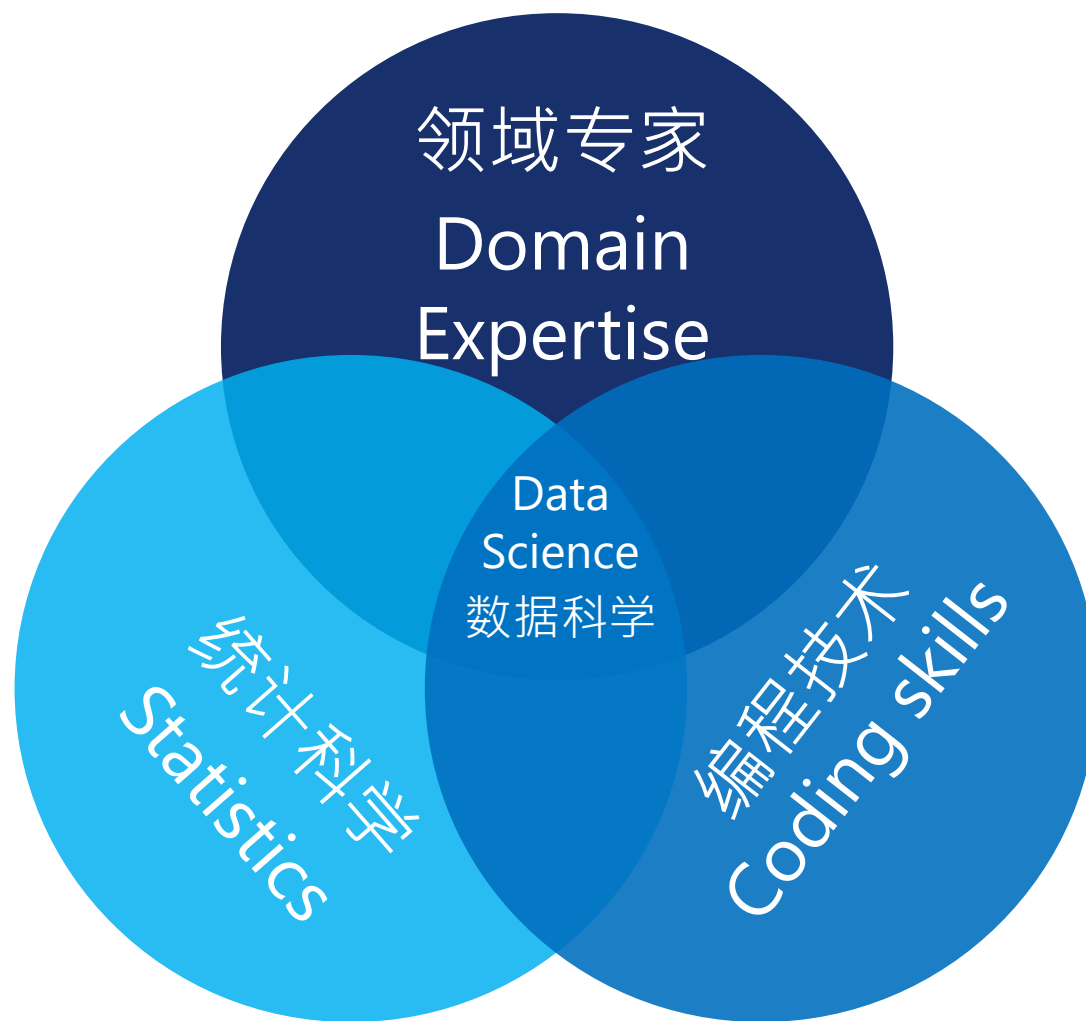


视觉障碍辅助



What is Data Science?

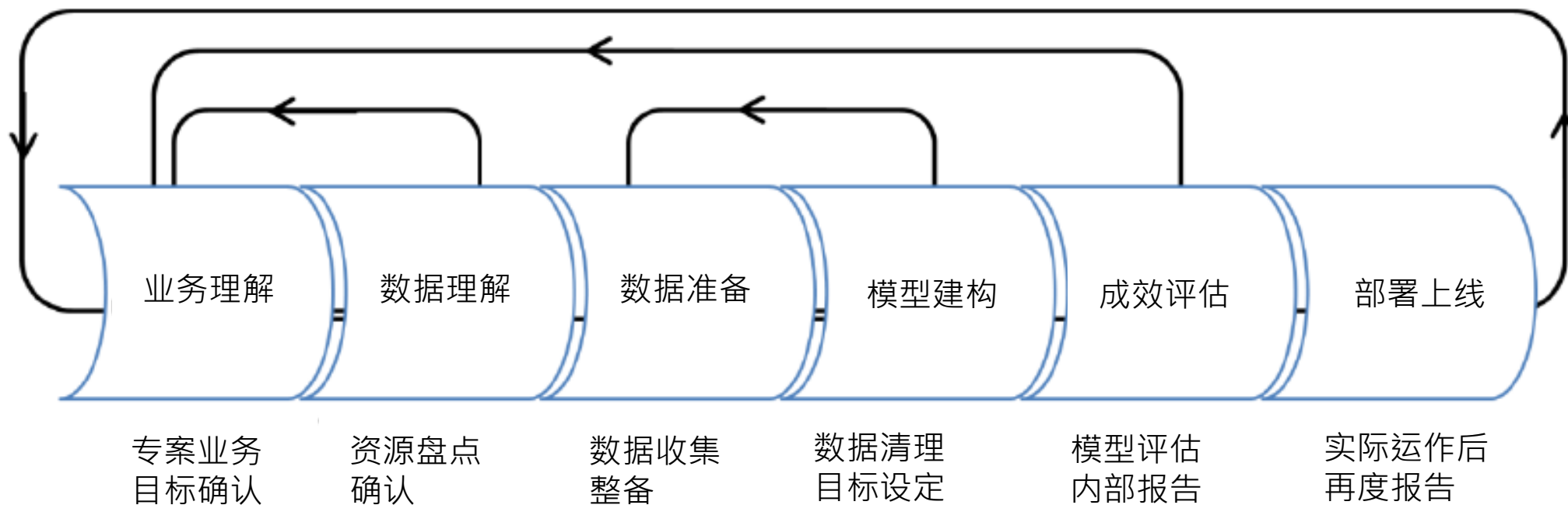
什么是数据科学?



Data Science Process 数据科学运作流程

首要挑战：从大数据中挖掘有意义的线索

Extracting value from large amounts of data and making human sense of it is the primary challenge of data science.



- from *Introduction to Data Science* on Microsoft Learn: <https://docs.microsoft.com/learn/modules/intro-to-data-science-in-azure/2-data-science-process>
- Helpful beginners' series: <https://docs.microsoft.com/en-us/azure/machine-learning/studio/data-science-for-beginners-the-5-questions-data-science-answers>

Specialized roles in Data Science 数据科学领域职业发展

技术导向

Data Scientist
Data Analyst
Data Architect
Data Engineer
Developer

数据科学家
数据分析师
数据架构师
数据工程师
开发者

领域导向

数据分析5个关键职务



(image source:managertoday)

關鍵職務	2015從業人數	2018需求人數	三年累計增幅	備註
領域專家	1,962人	3,201人	63%	-
資料科學家	1,092人	6,296人	477%	增幅最大
資料分析師	8,159人	14,607人	79%	-
資料工程師	21,763人	45,843人	110%	缺口最多
軟體工程師	8,001人	30,576人	282%	-

资料来源：104资讯科技



最佳职业选择 – 资料科学家 #1 2018 & 2019

薪水高
职缺多
持续成长
升职机会大

数据科学家

Harvard
Business
Review

DATA

Data Scientist: The Sexiest Job of the 21st Century

by Thomas H. Davenport and D.J. Patil

FROM THE OCTOBER 2012 ISSUE

2012年哈佛商业评论
数据科学家：21世纪最性感的职业

2015人力资源点评网Glassdoor调查
工作生活兼具薪水又高，数据科学家荣登最梦幻工作

「用数据解决真实问题的人」

工作生活平衡度排名前 25 职业薪资对照表
(Source : Glassdoor)

Job Title	Ranking	Salary
Data Scientist	4.2	114,808
SEO Manager	4.1	45,720
Talent Acquisition Specialist	4	63,504
Social Media Manager	4	40,000
Substitute Teacher	3.9	24,380
Recruiting Coordinator	3.9	44,700
UX Designer	3.9	91,440
Digital Marketing Manager	3.9	70,052
Marketing Assistant	3.8	32,512
Web Developer	3.8	66,040
Risk Analyst	3.8	69,088
Civil Engineer	3.8	65,532
Client Manager	3.8	71,120
Instructional Designer	3.8	66,040
Marketing Analyst	3.8	60,000
Software QA Engineer	3.8	91,440
Web Designer	3.8	53,848

生活工作平衡度前10名 - 2016

- ① Corporate Recruiter 人力资源招聘员
- ② UX Designer 用户体验设计师
- ③ Data Scientist 数据科学家
- ④ Strategy Manager 策略经理
- ⑤ UI Designer 用户接口设计师
- ⑥ Recruiting Coordinator 招聘协调员
- ⑦ Technical Account Manager 科技业务经理
- ⑧ Mobile Developer 移动开发程序设计师
- ⑨ DevOps Engineer DevOps工程师
- ⑩ Research Engineer 研究工程师



2020 工作与招聘趋势 - Glassdoor

- ① 人工智能将在管理政策上占有一席之地
- ② 打造公司文化，招聘相同气息的人
- ③ 因应经济紧缩，重新改写招聘规则
- ④ 更高的兼容，容许更多的不同
- ⑤ 高龄就职不是问题，只要能为公司产生价值
- ⑥ 越来越多人 在行动装置上找寻工作

AI in Management

智能指导系统

运用大数据产生实时回馈与
建议给予员工

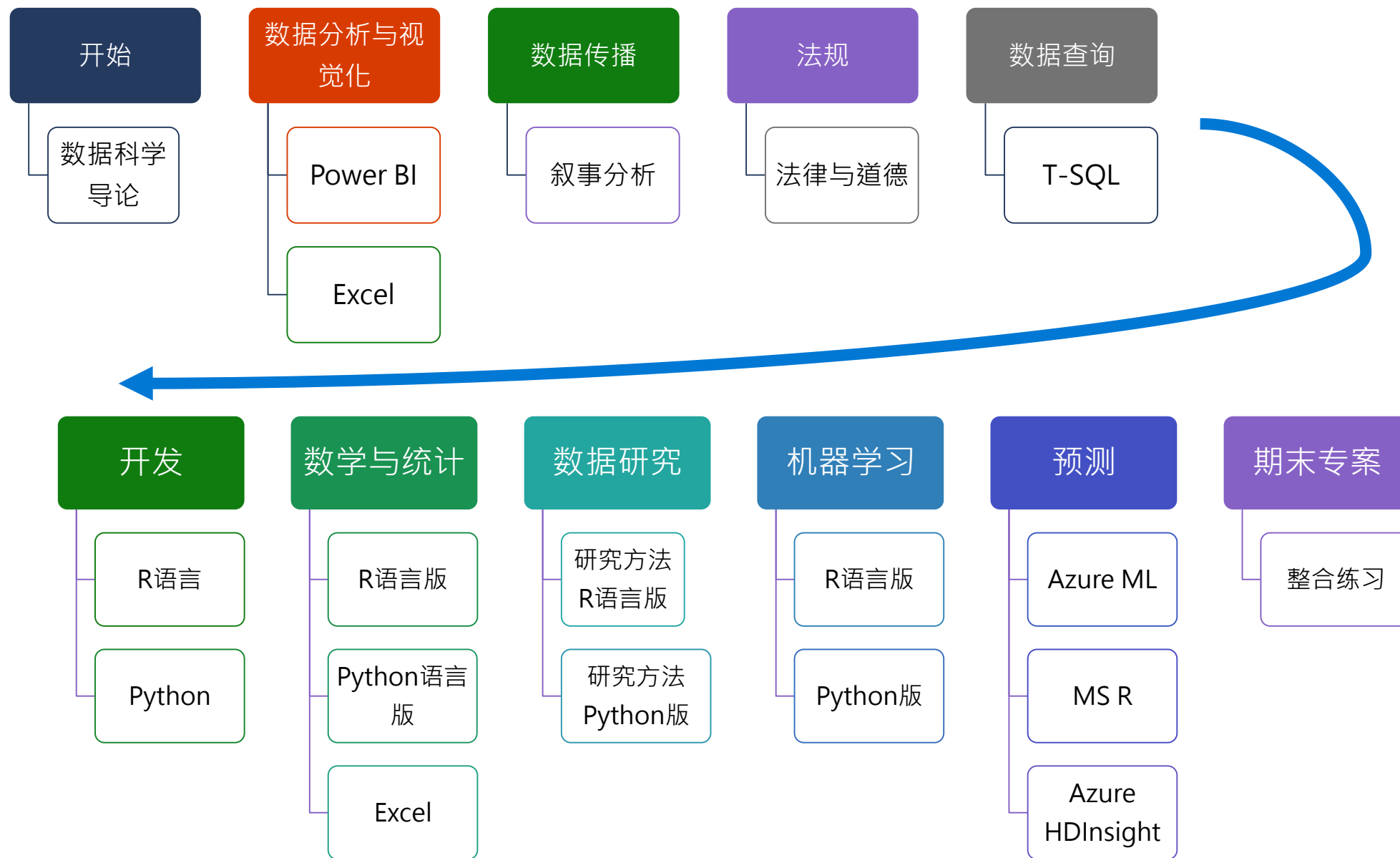
业务员工作流程优化

工作排程建议/活动监视

客户服务介入

实时提供更周到的响应建议
情绪指数、处理速度监视

数据科学学习路径



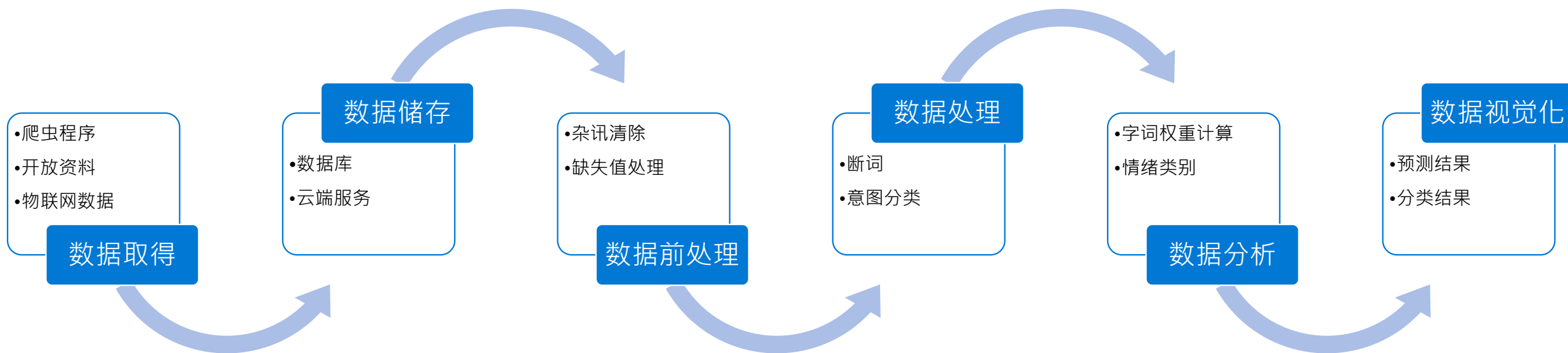
数据科学家应具备的能力

- 统计学、数学
- 程序语言
 - R、Python
- 视觉化工具呈现与讲解能力
 - Power BI, Tableau, Qlik, Excel
- 建模技术、汇整工具
 - Azure machine learning, Spark



Source: 微软MPP资料科学导论

数据处理 – 以文字探勘流程为例



Data
数据

搜集

储存

分析

呈现

Answer
答案

Python 程序设计基础

Section 1 第一节

云端练习环境

- DataCamp - Microsoft On-line Training Partner

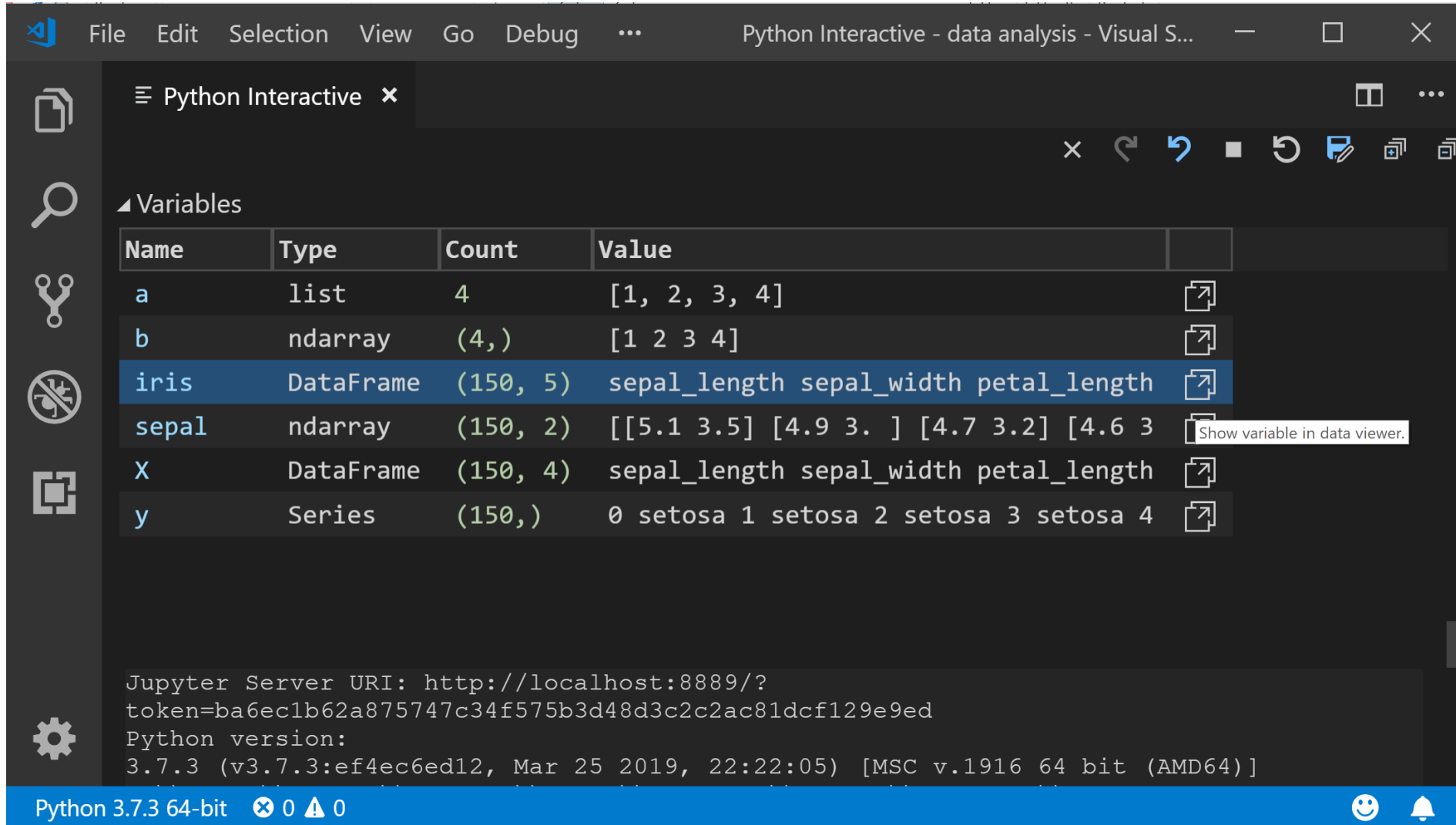
The screenshot shows the DataCamp web interface for a Python exercise. On the left, a sidebar contains the 'DataCamp' logo, a 'Course Outline' button, and an 'Exercise' section titled 'The Python Interface'. This section explains that users can type Python code to solve exercises, hit 'Run Code' or 'Submit Answer' to execute the script (script.py), and receive feedback. It also mentions that users can hit 'Run Code' and 'Submit Answer' as often as they want, or click 'Get Hint' or 'Get Solution' if stuck. At the bottom of the sidebar, there is a checkmark icon, the word 'Instructions', and a '100 XP' badge. The main area on the right is divided into two parts. The top part is a code editor for 'script.py' containing the following Python code:

```
1 # Example, do not modify!
2 print(5 / 8)
3
4 # Print the sum of 7 and 10
5
```

Below the code editor are three buttons: a circular arrow icon, a 'Run Code' button, and a green 'Submit Answer' button. The bottom part of the main area is the 'IPython Shell', which shows 'In [1]:' followed by a cursor. A 'Slides' tab is also visible next to the 'IPython Shell' tab.

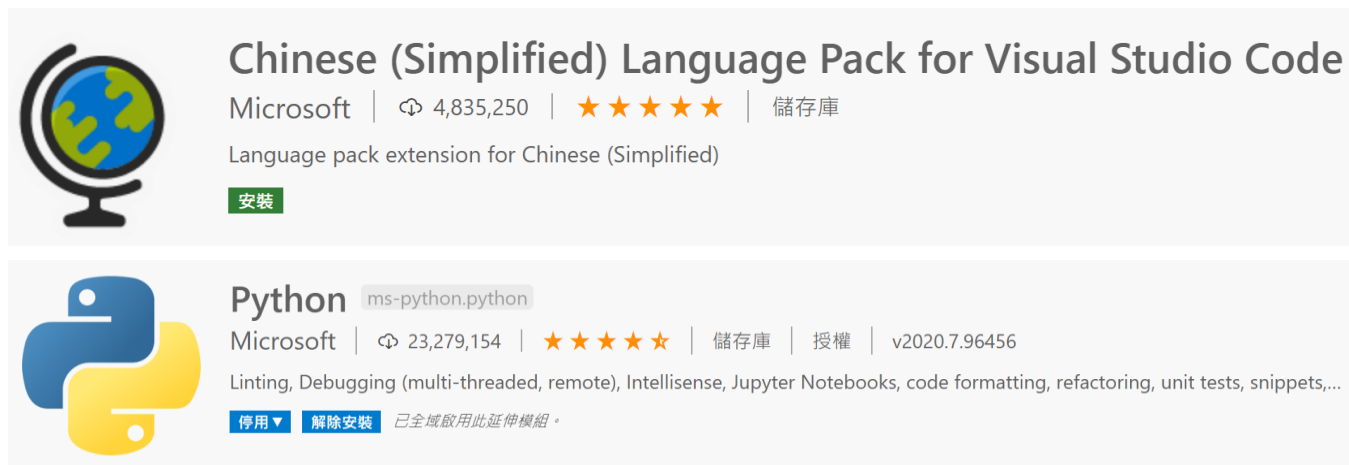
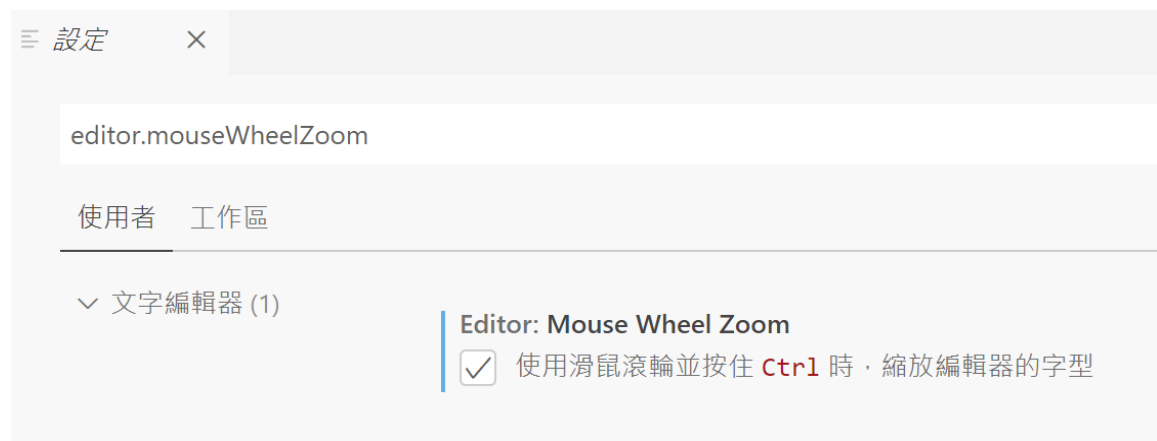
本机开发环境选择

- Microsoft Visual Studio Code



安装扩充套件与设定

- 按下左边 Extensions图示或 Ctrl + Shift + X
 - Chinese (Simplified) Language Pack for Visual Studio Code
 - Python
- 设定Ctrl+ 鼠标滚轴控制编辑器字号
 - editor.mouseWheelZoom
- 设定编辑时自动储存
 - 档案 -> 自动储存



Section 1 overview 第一节 综览

- Python的数值运算
- 字符串操作
- 其他资料型态
 - 串行 List 、序对 Tuple 、集合 Set 、字典 Dictionary
- 是否在群里
- 串行建构
- 模组汇入

Section 1 overview 第一节 综览

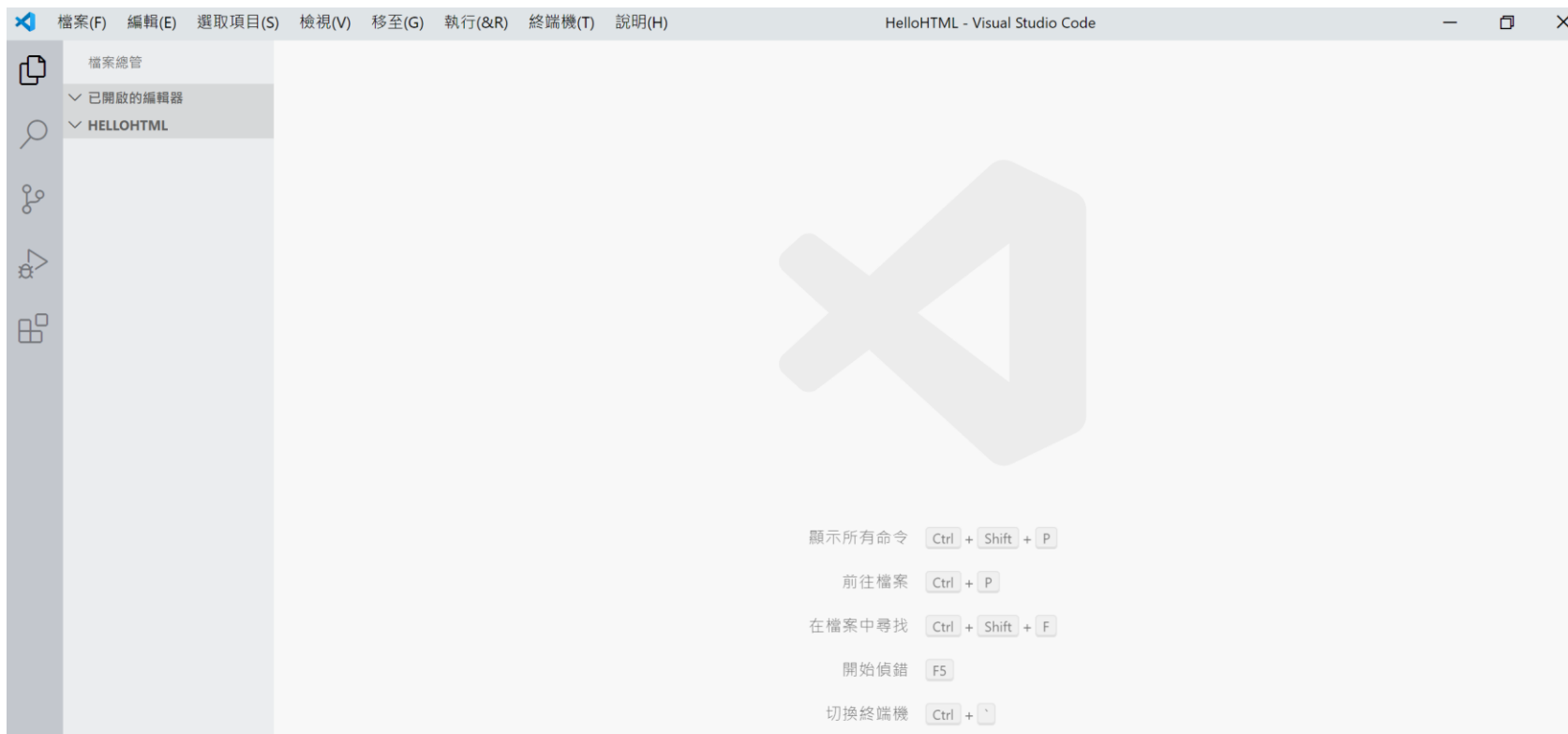
为什么使用Python在数据科学？

- Easy to learn 容易学习
- Flexible 弹性大
- Powerful libraries 强大的函式库

知名社群网路、影片平台以及搜寻引擎公司都大量使用**Python**在他们的核心技术，数据科学也是其中一个应用项目。

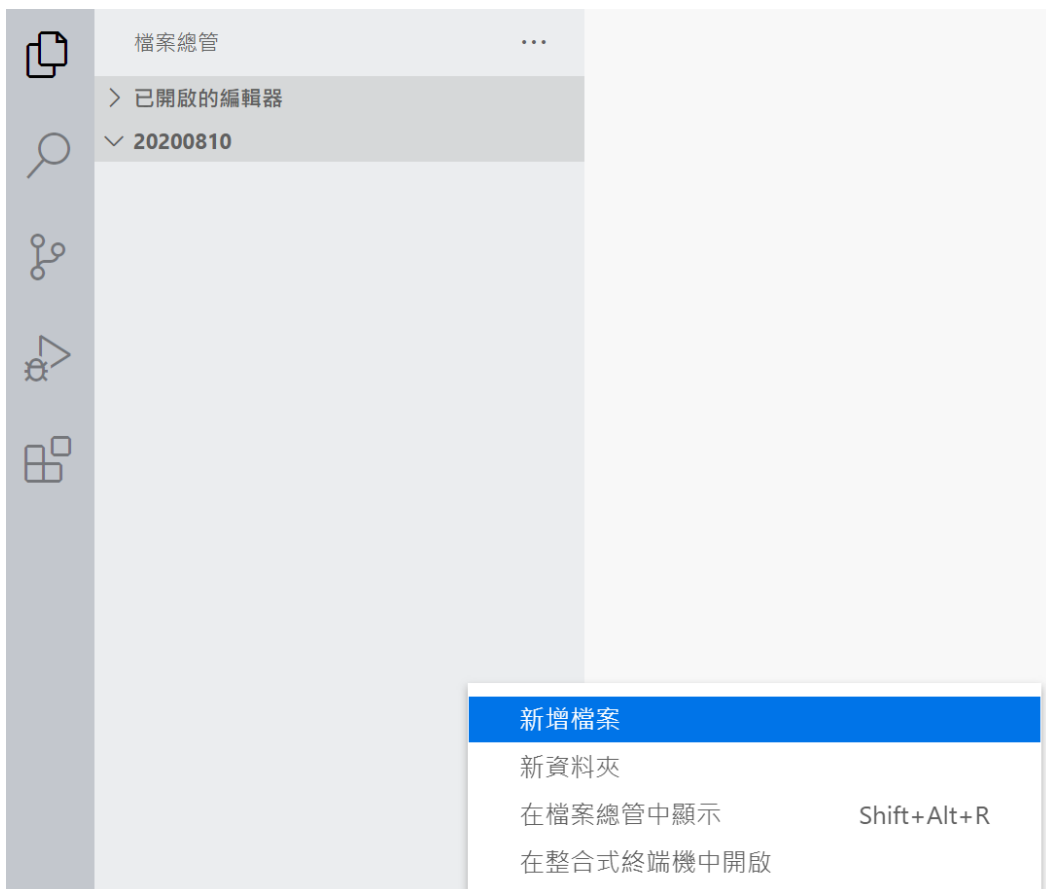
第一个 HelloWorld.py


- 在计算机中新增一个文件夹：HelloPython
- 打开VS Code, 档案(F) -> 开启文件夹...
- 选择刚才建立的文件夹
- 关闭「开始使用」分页




第一个 HelloWorld.py

- 左边档案总管区，按下鼠标右键 -> 新增档案
- 输入HelloWorld.py
- 档案前会出现Python图案




 Python Extension




Create a Jupyter Notebook

- Run "[Create New Blank Jupyter Notebook](#)" in the Command Palette (*Shift + Command + P*)
- Explore our [sample notebook](#) to learn about notebook features




Create a Python File

- Create a [new file](#) with a `.py` extension



Open a Folder or Workspace

- Open a [Folder](#)
- Open a [Workspace](#)

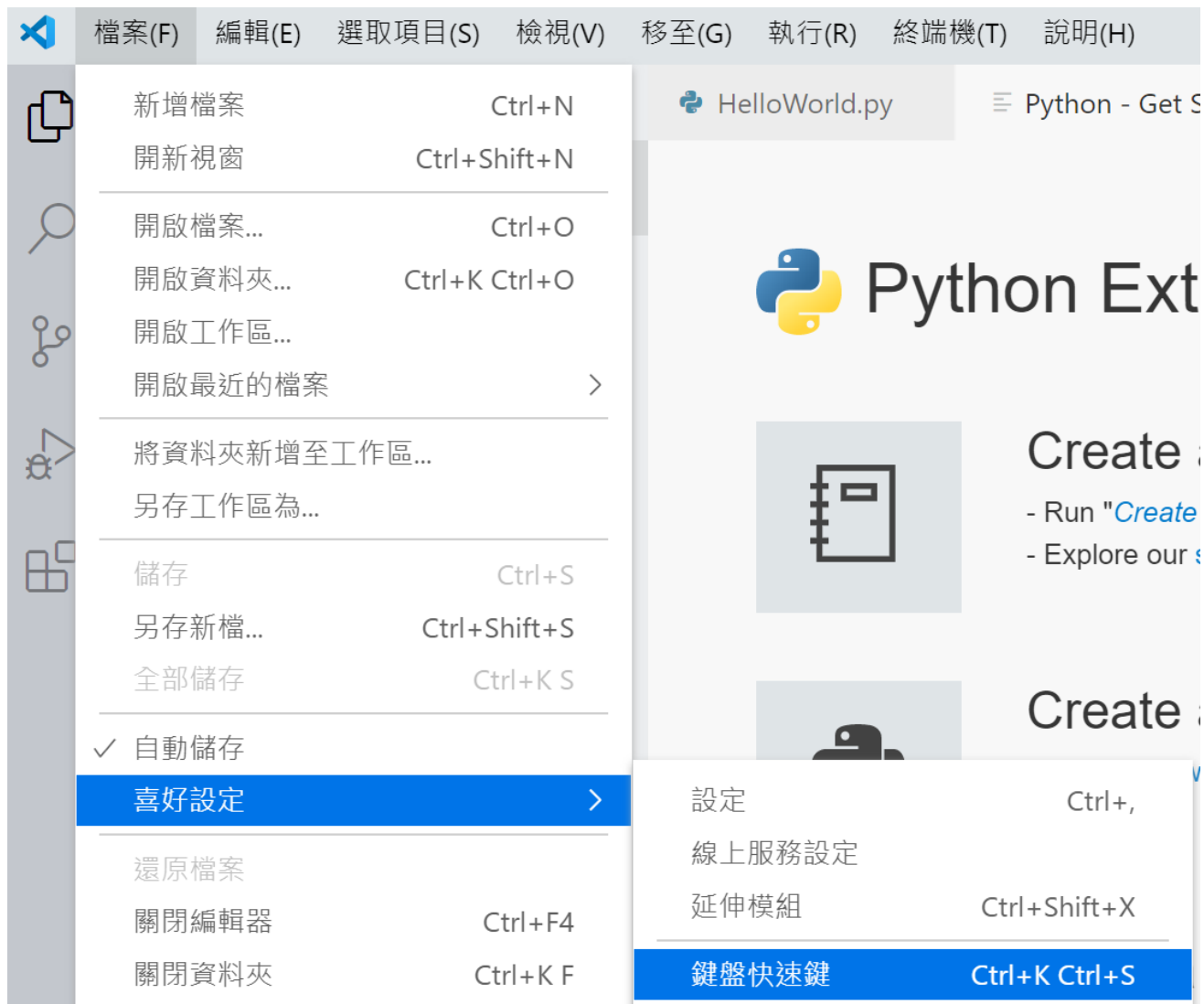


Use the Interactive Window to develop Python Scripts

- You can create cells on a Python file by typing `#%%`
- Use "*Shift + Enter*" to run a cell, the output will be shown in the interactive window

設定執行快捷鍵

- 檔案 -> 喜好設定 -> 鍵盤快捷方式



设定执行快捷键

- 输入Python进行搜索

Python			
命令	按键繫结關係	當	來源
Python: 在 Python 終端機中執行選定內容 / 行 Run Selection/Line in Python Terminal <code>python.execSelectionInTerminal</code>	Shift + Enter	editorTextFocus && !findInputFocussed && !python.datas...	預設
Python: Run Current Cell <code>python.datascience.runcurrentcell</code>	Ctrl + Enter	editorTextFocus && python.datascience.featureenabled &...	預設
Python: Run Current Cell And Advance <code>python.datascience.runcurrentcelladvance</code>	Shift + Enter	editorTextFocus && python.datascience.featureenabled &...	預設
Python: Run Current File in Python Interactive Window Run Current File in Python Interactive Window <code>python.datascience.runFileInteractive</code>	Shift + Alt + Enter	—	使用者
Python: Run Selection/Line in Python Interactive Window Run Selection/Line in Python Interactive Window <code>python.datascience.execSelectionInteractive</code>	Shift + Enter	editorTextFocus && python.datascience.featureenabled &...	預設
<code>python.datascience.runcurrentcellandadddb...</code>	Alt + Enter	editorTextFocus && python.datascience.featureenabled &...	預設

数值运算

- Python numeric operators: + - * / // ** %
数值运算符 商数 指数 余数

- Variables:
变数

```
length = 15  
width = 3 * 5  
length * width
```

225

The screenshot shows a Python IDE with two windows. The 'HelloPython.py' window contains the following code:

```
1 a = 5 / 2  
2 b = 5 // 2  
3 c = 5 ** 2  
4 d = 5 % 2
```

The 'Python Interactive' window shows the 'Variables' section with the following table:

Name	Type	Count	Value
a	float		2.5
b	int		2
c	int		25
d	int		1

- Expressions:
运算式/表达式

```
1 < 2 or 1 > 2
```

True

批注方式

- 单行

#

- 多行

Python会忽略没有指定给变量的字符串

'''

这边放批注

'''

String 字符串

• String literals: `'"Isn\'t," she said.'`

文字处理

`'"Isn\'t," she said.'`

```
HelloPython.py × 单引号、双引号、三引号
HelloPython.py > ...
1  userName = "王小明"
2  userNameWithLocation = '上海來的"王小明"'
3  userNameWithLocationAndCompany = """
4  上海來的王小明,
5  他很會寫Python."""
```

单引号、双引号皆可

单引号可以包着双引号

三引号可以换行

Python Interactive ×

× ↶ ↷ □ ↺ 📄 📋 📄

Variables

Name	Type	Count	Value
userName	str	3	王小明
userNameWithLocation	str	9	上海來的"王小明"
userNameWithLocationAndCompany	str	21	上海來的王小明, 他很會寫Python.

String 字符串

- Concatenating strings:

```
3 * 'un' + 'ium'
```

```
'unununium'
```

String.py > ...

```
1 badPersonName = "李大同"
2 hateWord = "我討厭你"
3 wantToTell = badPersonName + 3*hateWord
```

Python Interactive ×



Variables

Name	Type	Count	Value
badPersonName	str	3	李大同
hateWord	str	4	我討厭你
wantToTell	str	15	李大同我討厭你我討厭你我討厭你

String 字符串

- String indices:

取得单一字符，用[位置]，从0开始

```
word = 'Python'  
word[0]  # Character in position 0.
```

'P'

- Slicing strings:

取得部分字符串，[开始位置, 结束位置-不包含]

```
word[0:2]  # Characters from position 0 (included) to 2 (excluded).
```

'Py'

String 字符串

StringArray.py > ...

```
1 hostSay = "下雨天留客天留我不留"
2 mean1 = hostSay[0:3]+", "+hostSay[3:6]+", "+hostSay[6:9]+"? "+hostSay[9:]
3 mean2 = hostSay[0:2]+", "+hostSay[2:5]+", "+hostSay[5:7]+", "+hostSay[7:]
```

Python Interactive X



Jupyter Server:

Variables

Name	Type	Count	Value
hostSay	str	10	下雨天留客天留我不留
mean1	str	16	下雨天，留客天，留我不？ 留
mean2	str	16	下雨，天留客，天留，我不留

Other data types 其他资料型态

资料型态	中文	符号	是否有顺序性	内容是否可改变	概念
List	串行	[]	O	O	类似其他语言的阵列(数组)，但内容的资料型态可以不同
Tuple	序对	()	O	X	用在一组固定顺序的常数集合上
Set	集合	{ }	X	O	没有顺序概念，一堆资料放在一起
Dictionary	字典	{ }	X	O	Key-Value成对，中间是冒号

dataType.py > ...

```
1 roomGuest = ["王明", "柳宇", "陳尚"]
2 roomKey = (1111, 2222, 3333)
3 breakfastChoice = {"中式", "西式", "法式"}
4 guestBreakfast = {
5     "王明": "中式",
6     "柳宇": "法式",
7     "陳尚": "中式"
8 }
```

Python Interactive X

Variables

Name	Type	Count	Value
breakfastChoice	set	3	{'西式', '法式', '中式'}
guestBreakfast	dict	3	{'王明': '中式', '柳宇': '法式', '陳尚': '中式'}
roomGuest	list	3	['王明', '柳宇', '陳尚']
roomKey	tuple	3	(1111, 2222, 3333)

Membership testing 是否在群里

dataType.py > ...

```
1 roomGuest = ["王明", "柳宇", "陳尚"]
2 roomKey = (1111, 2222, 3333)
3 breakfastChoice = {"中式", "西式", "法式"}
4 guestBreakfast = {
5     "王明": "中式",
6     "柳宇": "法式",
7     "陳尚": "中式"
8 }
9 print("王明有住房嗎? " + str("王明" in roomGuest))
10 print("陳尚應該沒來住房吧? " + str("陳尚" not in roomGuest))
```

Python Interactive X

X ↶ ↷ □ ↻ 田 窗 窗 窗

[11] ▶ roomGuest = ["王明", "柳宇", "陳尚"]...



X 王明有住房嗎? True
陳尚應該沒來住房吧? False

List comprehension 串行建构

Programmatically create lists: 程序化自动建立串行内容

ListComprehension.py > ...

```
1 numbers = [x for x in range(1,11)] #注意不包含range結束
2 odd_numbers = [x for x in range(1,11) if x %2 != 0]
3 even_numbers = [x for x in range(1,11) if x %2 == 0]
4 odd_squares = [x*x for x in range(1,11) if x % 2 != 0]
```

Python Interactive ×



Variables

Name	Type	Count	Value	
even_numbers	list	5	[2, 4, 6, 8, 10]	
numbers	list	10	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	
odd_numbers	list	5	[1, 3, 5, 7, 9]	
odd_squares	list	5	[1, 9, 25, 49, 81]	

Importing modules 模組匯入

匯入整個模組進行使用 `import ...`

```
importModules.py > ...  
1 import math  
2 x = math.factorial(5) # 5!=5*4*3*2*1=120
```

Python Interactive X

Variables

Name	Type	Count	Value
x	int		120

匯入部分模組進行使用 `from ... import ...`

```
importModules.py > ...  
1 from math import factorial  
2 x = factorial(5) # 5!=5*4*3*2*1=120
```

Python Interactive X

Variables

Name	Type	Count	Value
x	int		120

07.import_modules.py

import 慣例寫法

```
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
import statsmodels as sm
```

自訂匯入模組名稱 `import ... as ...`

```
importModules.py X  
importModules.py > ...  
1 from math import factorial as stepDown  
2 x = stepDown(5) # 5!=5*4*3*2*1=120
```

Python Interactive X

Variables

Name	Type	Count	Value
x	int		120

产生一个乱数

randomNumbers.py > ...

```
1 import random
2 #產生一個亂數                                不包含尾数
3 x = random.randrange(10) #產生介於0~10的亂數
4 y = random.randrange(20,40) #產生介於20~40的亂數
5 z = random.randrange(0,60,6) #產生介於0~60的亂數，除了0以外只會是6的倍數
```

Python Interactive X



Jupyter Server: local

Variables

Name	Type	Count	Value
x	int		2
y	int		22
z	int		18

综合练习：家庭**BMI**值计算

案例：BMI值计算

1. 直接计算BMI值
2. 显示BMI值对应的结果
3. 用List建立全家人的BMI资料
4. 帮全家人检验，判断是否落于正常范围
5. 用BMI资料画图，观察一年的体重变化
6. 引用儿童的BMI指标来进行比对

使用Python计算BMI值

- 变数直接命名、直接给值
- `**n`：乘以自己n次

```
BMI > 01.HelloWorld.py > ...  
1  height = 1.78  
2  weight = 72.0  
3  bmi = weight / height **2
```

Python Interactive X



Variables

Name	Type	Count	Value
bmi	float		22.724403484408533
height	float		1.78
weight	float		72.0

BMI值标准

成人的体重分级与标准	
分 级	身体质量指数
体重过轻	$BMI < 18.5$
正常范围	$18.5 \leq BMI < 24$
过 重	$24 \leq BMI < 27$
轻度肥胖	$27 \leq BMI < 30$
中度肥胖	$30 \leq BMI < 35$
重度肥胖	$BMI \geq 35$
资料来源：食品资讯网 / 肥胖及体重控制	

显示判断结果

- if
- elif
- else

```
BMI > 01.HelloWorld.py > ...
1 height = 1.78
2 weight = 72.0
3 bmi = weight / height **2
4 print('你的BMI值为'+("%.2f" % bmi))
5 if bmi <18.5:
6     print('體重過輕')
7 elif 18.5 <= bmi < 24.0:
8     print('正常範圍')
9 elif 24.0 <= bmi < 27.0:
10    print('體重過重')
11 elif 27.0 <= bmi < 30.0:
12    print('輕度肥胖')
13 elif 30.0 <= bmi < 35.0:
14    print('中度肥胖')
15 else:
16    print('重度肥胖')
```

小数点后留两位数的浮点数

可以直接用一个范围来写

所有条件式不用加()
最后面打上冒号

```
Python Interactive X
[8] height = 1.78...
你的BMI值为22.72
正常範圍
```

用List建立全家的BMI资料

- 名称、身高、体重、BMI值

BMI > 02.FamilyBMI.py > ...

```
1 family_data = [  
2     ['Dad',178,72],  
3     ['Mom',155,44],  
4     ['Kid',117,19]  
5 ]  
6  
7 for each_one in family_data:  
8     this_bmi = each_one[2] / ((each_one[1]/100) **2)  
9     each_one.append(this_bmi)  
10  
11 family_data
```

计算完之后，加回原本的List

Python Interactive X

× ↶ ↷ □ ↻ 📊 📄 📁 📂

📄

×

```
[['Dad', 178, 72, 22.724403484408533],  
 ['Mom', 155, 44, 18.314255983350673],  
 ['Kid', 117, 19, 13.879757469501062]]
```

用List建立全家的BMI资料

BMI/02.FamilyBMI.py

- 名称、身高、体重、BMI值、BMI指标

```
10
11 for each_one in family_data:
12     if each_one[3] < 18.5:
13         bmi_index = '體重過輕'
14     elif 18.5 <= each_one[3] < 24.0:
15         bmi_index = '正常範圍'
16     elif 24.0 <= each_one[3] < 27.0:
17         bmi_index = '體重過重'
18     elif 27.0 <= each_one[3] < 30.0:
19         bmi_index = '輕度肥胖'
20     elif 30.0 <= each_one[3] < 35.0:
21         bmi_index = '中度肥胖'
22     else:
23         bmi_index = '重度肥胖'
24     each_one.append(bmi_index)
25
26 family_data
```

把判断完的结果也放回List

```
[[ 'Dad', 178, 72, 22.724403484408533, '正常範圍'],
 [ 'Mom', 155, 44, 18.314255983350673, '體重過輕'],
 [ 'Kid', 117, 19, 13.879757469501062, '體重過輕']]
```

用List建立全家的BMI资料

- 名称、身高、体重、BMI值、BMI指标、是否落于正常范围

```
25
26 for each_one in family_data:
27     if each_one[4] == '正常範圍':
28         is_normal = True
29     else:
30         is_normal = False
31     each_one.append(is_normal)
32
33 family_data
```

增加一个布尔值，来记录是否落在正常范围

```
[[ 'Dad', 178, 72, 22.724403484408533, '正常範圍', True],
 [ 'Mom', 155, 44, 18.314255983350673, '體重過輕', False],
 [ 'Kid', 117, 19, 13.879757469501062, '體重過輕', False]]
```

前面用到的语法

- 资料型态
 - 字符串、整数、浮点数、布尔值、list
- 运算符
 - 等于、大于、大于等于、小于等于、小于
- 流程控制
 - 回圈
 - if..else if..else
- List 运算
 - 新增元素

检查资料型态

```
32
33 for each_one in family_data[0]:
34     print(type(each_one))
```

用type() 检查资料型态

0	1	2	3	4	5
Dad	178	72	22.7244034844	正常範圍	true

```
<class 'str'>
<class 'int'>
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
```

观察list中的部分元素

- 起始 : 结束
- 取出元素不包含结束
- 不写开头，代表从最前面开始
- 不写结尾，代表从那一项开始走到全部结束

family_data

全部的资料

```
[[ 'Dad', 178, 72, 22.724403484408533, '正常範圍', True],  
 [ 'Mom', 155, 44, 18.314255983350673, '體重過輕', False],  
 [ 'Kid', 117, 19, 13.879757469501062, '體重過輕', False]]
```

family_data[0:2]

第0笔、第1笔

```
[[ 'Dad', 178, 72, 22.724403484408533, '正常範圍', True],  
 [ 'Mom', 155, 44, 18.314255983350673, '體重過輕', False]]
```

family_data[1:]

第1笔、第2笔

```
[[ 'Mom', 155, 44, 18.314255983350673, '體重過輕', False],  
 [ 'Kid', 117, 19, 13.879757469501062, '體重過輕', False]]
```

family_data[:2]

第0笔、第1笔

```
[[ 'Dad', 178, 72, 22.724403484408533, '正常範圍', True],  
 [ 'Mom', 155, 44, 18.314255983350673, '體重過輕', False]]
```

函数/函式 Function

- 关键词def
- 注意缩排、冒号

```
def functionName([parameters]):  
    statements  
    [return | return value]  
    [statements]
```

练习：温度单位转换运算

- 输入：温度、单位
- 输出：摄氏温度、华氏温度

练习：温度单位转换运算

```
def DegreeTransfer(degree, unit):  
    if unit == "C" or unit == "c":  
        degreeF = degree * (9/5) + 32  
        degreeC = float(degree)  
    elif unit == "F" or unit == "f":  
        degreeC = (5/9) * (degree - 32)  
        degreeF = float(degree)  
    else:  
        print("输入值有误，请重新输入")  
        return  
    print("华氏"+str(degreeF)+"°F、摄氏"+str(degreeC)+"°C")  
  
inputDegree, inputUnit = input("请输入温度与单位，例如摄氏25度c可输入'25 c'").split()  
DegreeTransfer(eval(inputDegree), inputUnit)
```



Reactor



developer.microsoft.com/reactor/
[@MSFTReactor](#) on Twitter

议程结束 感谢聆听



请记得填写课程回馈问卷
<https://aka.ms/ReactorFeedback>

© 2019 Microsoft Corporation. All rights reserved. The text in this document is available under the Creative Commons Attribution 3.0 License, additional terms may apply. All other content contained in this document (including, without limitation, trademarks, logos, images, etc.) are not included within the Creative Commons license grant. This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it. Some examples are for illustration only and are fictitious. No real association is intended or inferred. Microsoft makes no warranties, express or implied, with respect to the information provided here.