



AI Development Fundamentals

Python - 太空探索应用

**Intro to Python for
Space Exploration**

Dec 2020

Microsoft Reactor | Ryan Chung

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



5000 XP

了解 Python 在太空探索中扮演的角色

2 小时 34 分钟 剩余 • 学习路径 • 已完成 0 个模块，共 5 个

初级

学生

Visual Studio Code

此学习路径介绍了 Python 的世界。但目标不是学习 Python，而是了解 Python 如何在 NASA 创建的创新型解决方案中发挥作用。该学习路径通过太空探索镜头来激发持之以恒地学习、探索和创建的热情，让你有一天也能帮助我们所有人更多地了解一点外太空。

通过这些模板，你将：

- 了解和安装学习编程所需的工具
- 了解核心编程概念并在实际的 NASA 问题中运用它们
- 了解机器学习和人工智能等领先技术
- 观看真实的 NASA 员工谈论他们的工作并给出建议

先决条件

无

[继续 >](#)[书签](#)[+ 添加到集合](#)

学习目标

- 了解程式设计所需工具
- 将程式设计套用于NASA面临的真实问题
- 使用机器学习与人工智能知识
- 初步了解NASA员工工作内容

学习路径



面向太空探索的 Python 简介

4 分钟 剩余 • 模块 • 已完成 6 个单元, 共 8 个

★★★★★ 4.8 (705)

了解 Python 和数据科学可以影响的太空探索问题的类型。



安装用于 Python 开发的编码工具

40 分钟 剩余 • 模块 • 已完成 0 个单元, 共 11 个

★★★★★ 4.8 (451)

了解什么是编码, 并安装工具以帮助编码。

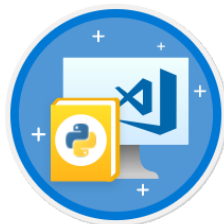


使用 Python 按类型统计月球岩石数

32 分钟 剩余 • 模块 • 已完成 0 个单元, 共 8 个

★★★★★ 4.8 (223)

使用 Python 和 Visual Studio Code 编写一个简单程序来统计每种类型的太空岩石数。



在 Visual Studio Code 中的笔记本中编写基本 Python

37 分钟 剩余 • 模块 • 已完成 0 个单元, 共 9 个

★★★★★ 4.8 (322)

了解 Python 的基础知识。



Python 中的代码控制语句

41 分钟 剩余 • 模块 • 已完成 0 个单元, 共 9 个

★★★★★ 4.8 (231)

详细了解 Python 的高级主题。

阿尔忒弥斯计划 Artemis program











- 人员登月探勘
 - 探索基础系统
 - 太空发射系统
 - 耐高温设计
 - 太空通讯
 - 登陆系统
 - 太空服



岩石研究

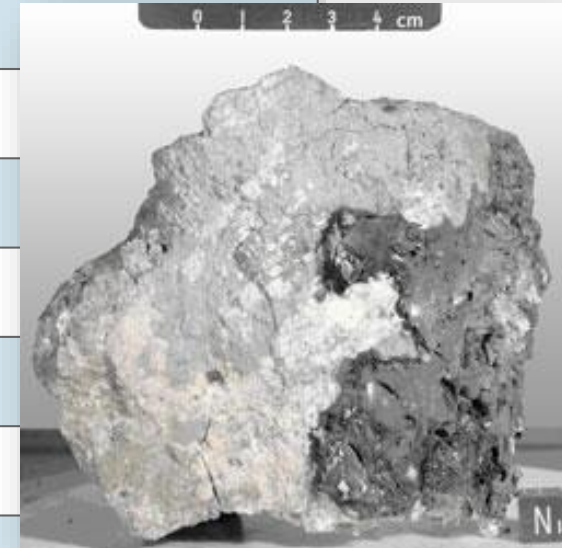
Show 10 ▾ entries

Search all columns:

Generic	Mission	Collection Site	Rock Type	Weight	% Pristine	Display Samples
10001	 Apollo 11		Soil » Unsieved	125.80	88.36	
10002	 Apollo 11		Soil » Unsieved	5629.00	93.73	
10003	 Apollo 11		Basalt » Ilmenite	213.00	65.56	
10004	 Apollo 11	Station LM	Core » Unsieved	44.80	71.76	
10005	 Apollo 11	Station LM	Core » Unsieved	53.40	40.31	
10006	 Apollo 11		Unclassified	0.00	0.00	
10007	 Apollo 11		Unclassified	0.00	0.00	
10008	 Apollo 11		Soil » Unsieved	89.00	5.75	
10009	 Apollo 11		Breccia » Regolith	112.00	97.27	
10010	 Apollo 11		Soil » Unsieved	491.00	91.03	

Showing 1 to 10 of 2511 entries

First Previous 1 2 3 4 5 Next Last



<https://curator.jsc.nasa.gov/lunar/>

向下扎根的太空科研

- 月球上的下一步挑战
- NASA STEM @ Home
- 月球到火星



知识检查

1. NASA 主题的 Microsoft Learn 模块的目标是什么？

- ☐ 激励下一代都成为宇航员。
- ☐ 激励科学家专注于太空探索。
- ☐ 激励各行各业解决问题的人坚持和创新。
- ☐ 激励 Artemis 计划的创新。

2. 人类上次登上月球是什么时候？

- ☐ 300 多年前
- ☐ 在 2015 年
- ☐ 大约 50 年前
- ☐ 在 1999 年

3. 下一次人类登上月球是什么时候？

- ☐ 到 2024 年。
- ☐ 在 3000 年。
- ☐ 还没有计划。
- ☐ 现在有人在月球上。

本机开发环境设置

- 安装Python

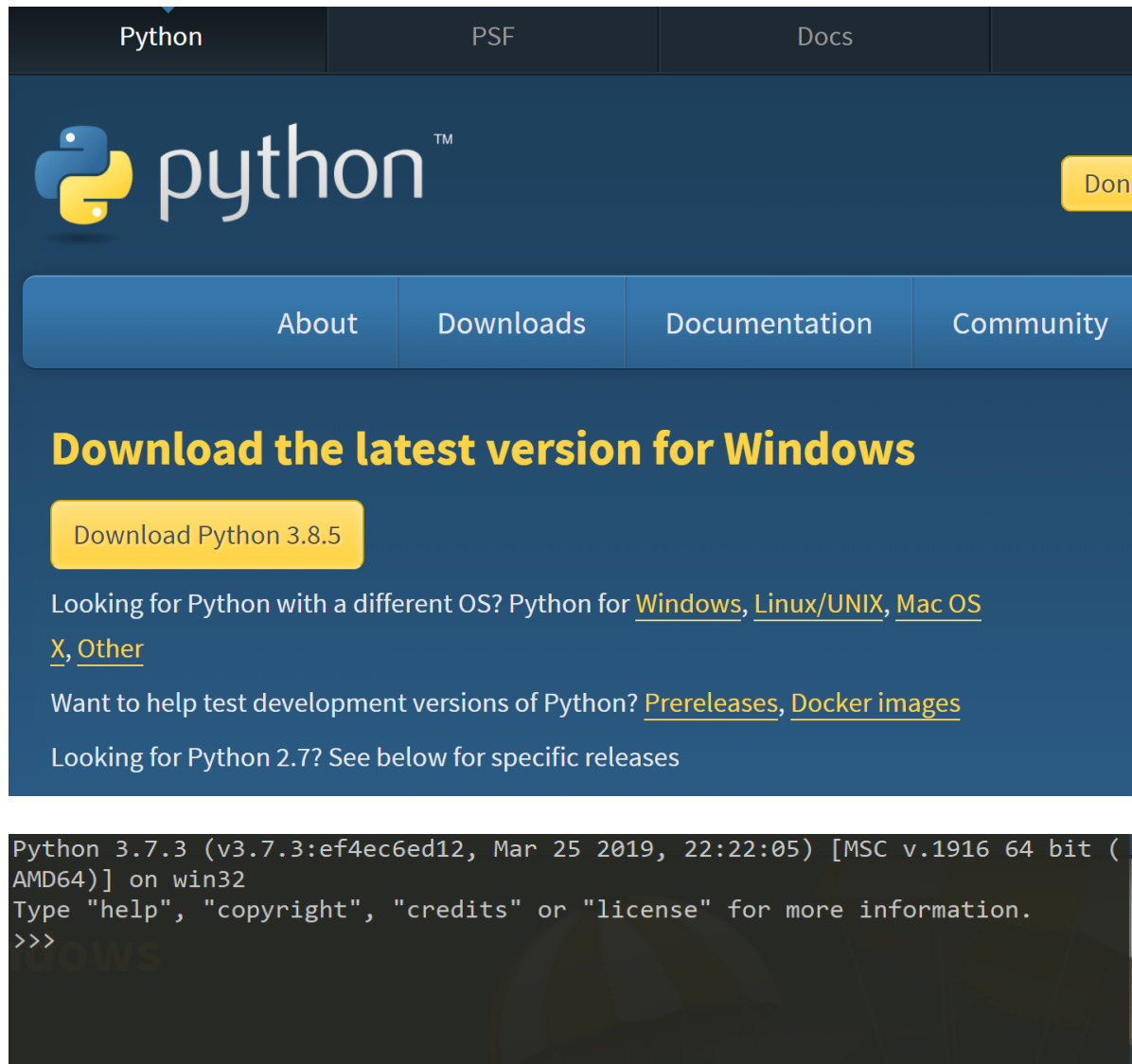
- 下载对应操作系统的版本

- 确认安装

- 开启命令提示字符
 - 输入python
 - 如有出现执行环境代表确认安装完成
 - 按下Ctrl+Z退出

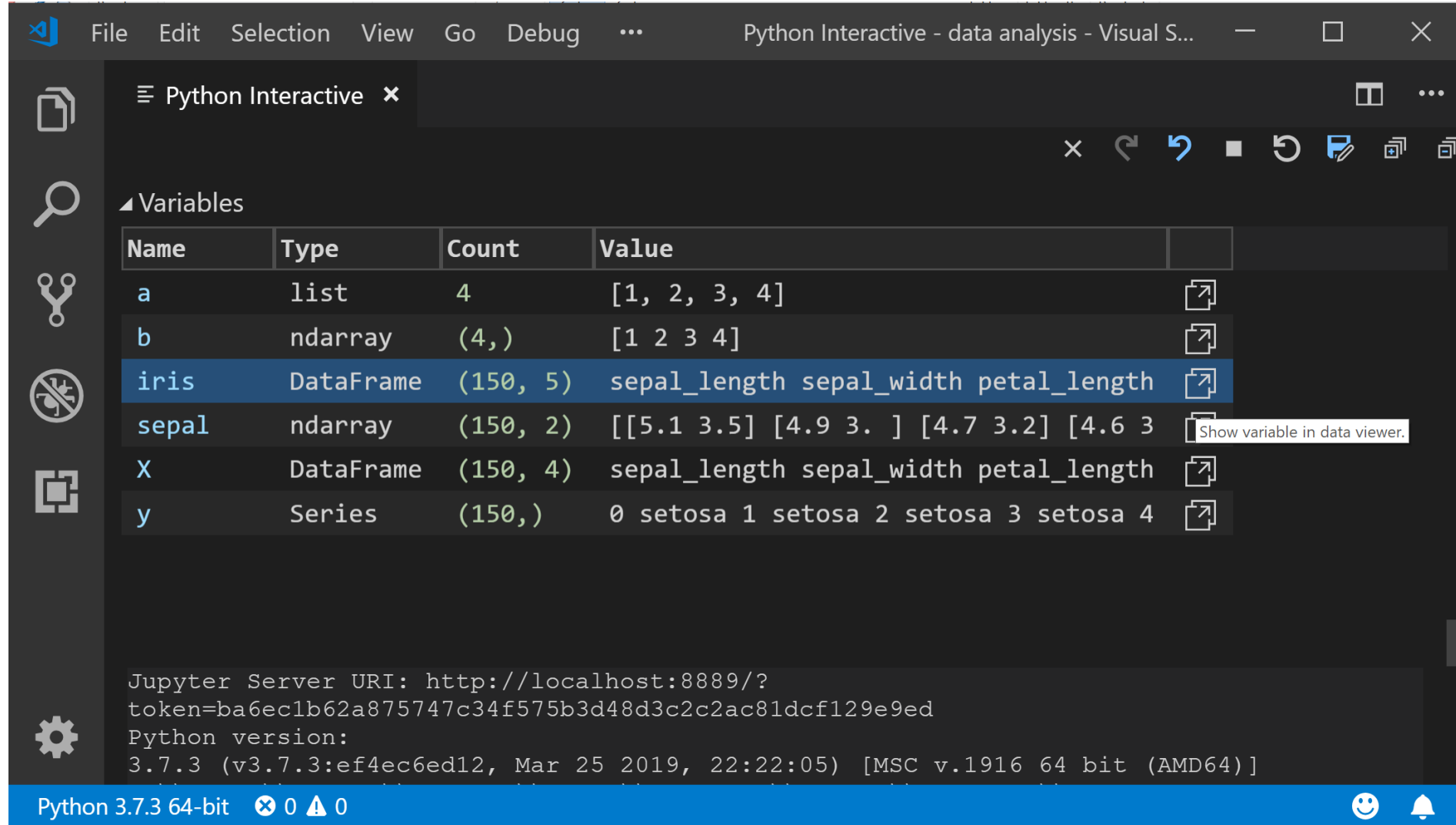
- 版本确认

- 开启命令提示字符
 - 输入py --version



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
 - Pylance
 - Visual Studio IntelliCode
- 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




Chinese (Simplified) Language Pack for Visual Studio Code

Microsoft | 4,835,250 | ★★★★★ | 儲存庫

Language pack extension for Chinese (Simplified)

安裝




Python

ms-python.python | Microsoft | 23,279,154 | ★★★★★ | 儲存庫 | 授權 | v2020.7.96456

Linting, Debugging (multi-threaded, remote), Intellisense, Jupyter Notebooks, code formatting, refactoring, unit tests, snippets,...

停用 ▼ **解除安裝** 已全域啟用此延伸模組。



Pylance

ms-python.vscode-pylance | Microsoft | 194,470 | ★★★★★☆ | 儲存庫 | 授權 | v2020.9.8

A performant, feature-rich language server for Python in VS Code

停用 ▼ **解除安裝** 已全域啟用此延伸模組。

Overview 综览

为什么使用Python?

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

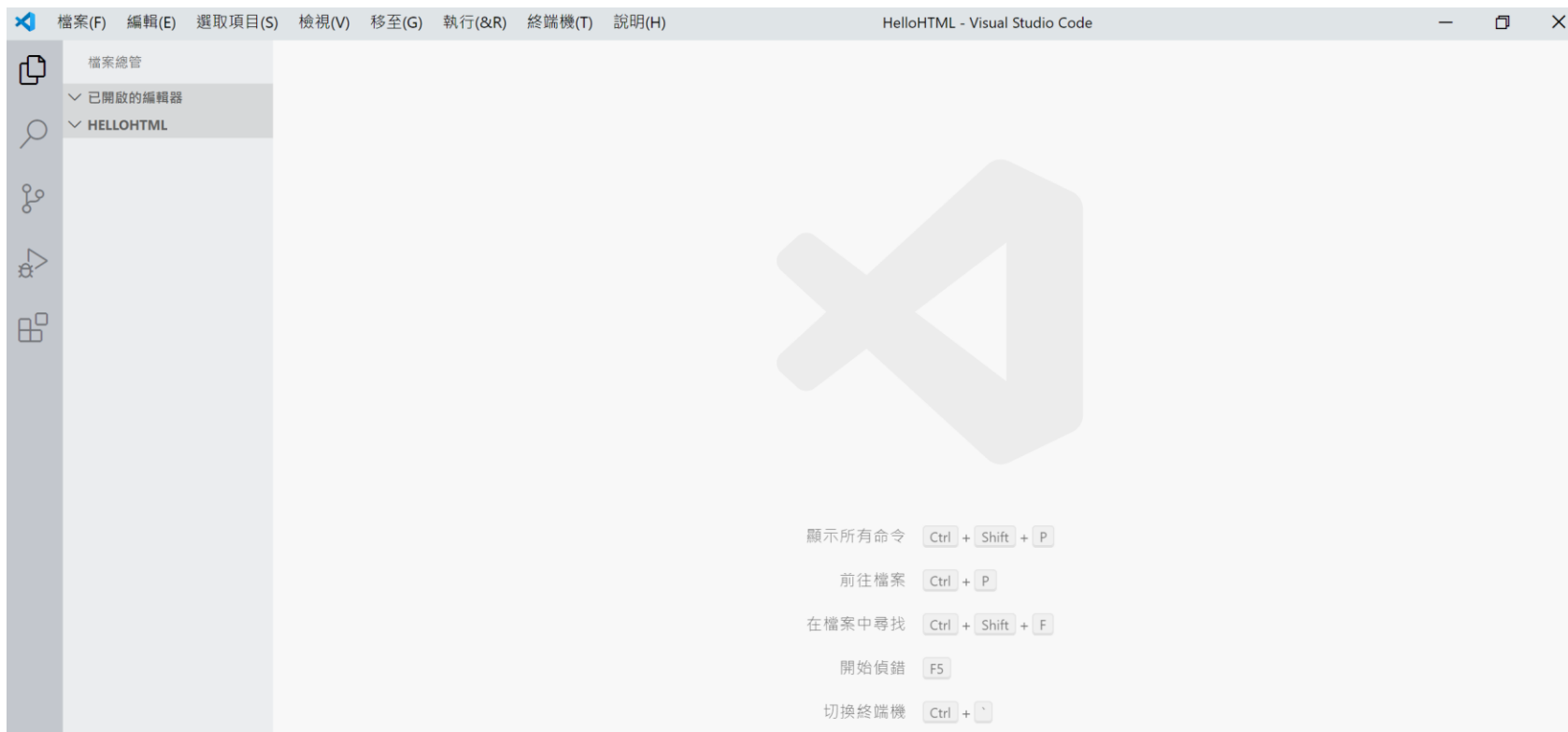
- 处理数据的业务应用
- 动态 Web 应用
- 2D 和 3D 游戏
- 金融和科研应用
- 基于云的应用
- 移动应用

影片平台、云平台以及搜寻引擎公司都大量使用**Python**在他们的核心技术，**NASA**也有一个**Python**的开源代码项目站。

<https://code.nasa.gov/>

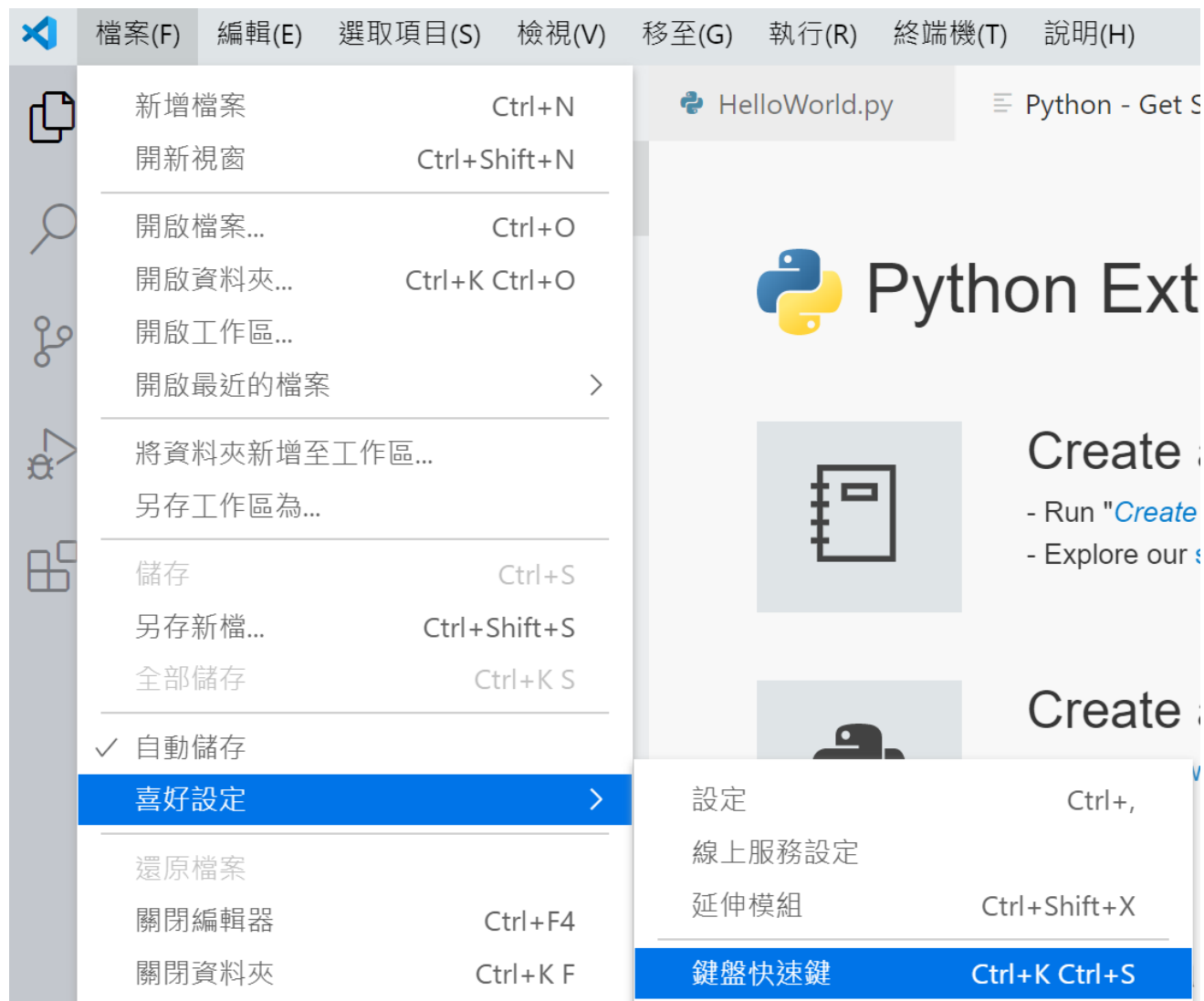
第一个 HelloWorld.py

- 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



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进行搜寻

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 V 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 Run Selection/Line in Python Interactive Window <code>python.datascience.execSelectionInteractive</code>	Shift + Enter	editorTextFocus && python.datascience.featureenabled &...	預設
<code>python.datascience.runcurrentcellandaddb...</code>	Alt + Enter	editorTextFocus && python.datascience.featureenabled &...	預設

错误提示

```
Print("Hello")
```

Print: Any

**"Print" is not defined Pylance
(reportUndefinedVariable)**

瞄孔問題 (Alt+F8) 沒有可用的快速修正

知识检查

1. 人类如何与计算机交流？

- ☐ 英语
- ☐ 编程语言
- ☐ 技术语言
- ☐ 数字

2. 软件开源意味着

- ☐ 你需要朋友为你提供其访问权限。
- ☐ 付费可以获得更多功能。
- ☐ 没有人可以查看代码。
- ☐ 任何人都可以对其进行开发并向其添加内容。

3. Visual Studio Code 不符合以下哪一项描述？

- ☐ 代码编辑器
- ☐ 开源
- ☐ 售价 50 美元

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

'''

数值运算

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

- Variables:
变数/变量

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

225

The screenshot shows a Python IDE with a file named 'HelloPython.py' open. The code in the editor is:

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

To the right of the editor is a 'Python Interactive' window. It contains a 'Variables' table with the following data:

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

建立变量

```
numberOfRocks = 5
tempInSpace = -457.87
roverName = "Artemis Rover"
rocketOn = False
```

Variables

	Name	Type	Size	Value
	numberOfRocks	int		5
	rocketOn	bool		False
	roverName	str	13	Artemis Rover
	tempInSpace	float		-457.87

岩石数量计算

```
basaltRockCount = 0  
basaltRockCount = 3  
basaltRockCount = basaltRockCount + 1  
basaltRockCount = 5  
basaltRockCount += 3  
basaltRockCount -= 2
```

Variables

	Name	Type	Size	Value
	basaltRockCount	int		6

[9] basaltRockCount



6

字符串 String 操作

- upper() 大写
- lower() 小写
- capitalize() 首字大写

```
astronaut = "Remy Morris"
upperCase = astronaut.upper()
upperCase
lowerCase = astronaut.lower()
lowerCase
rocketOutput = "rOckEt iS A lAUnCh!"
rocketOutput.capitalize()
```

Variables

	Name	Type	Size	Value
	astronaut	str	11	Remy Morris
	lowerCase	str	11	remy morris
	rocketOutput	str	19	rOckEt iS A lAUnCh!
	upperCase	str	11	REMY MORRIS

```
[2] rocketOutput.capitalize()
✖ 'Rocket is a launch!'
```

字符串 String 操作

- 串接 +
- 重复 *

```
launchLocationCity = "Cape Canaveral, "  
launchLocationState = "Florida"  
launchLocationCity + launchLocationState
```

```
artemisRoverSounds = "beep beep "  
artemisRoverSounds * 3
```

```
[5] launchLocationCity + launchLocationState  
    'Cape Canaveral, Florida'
```

```
[7] artemisRoverSounds * 3  
    'beep beep beep beep beep beep '
```

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)

List 串行 操作

Create a list of common moon rocks

```
rockTypes = ["basalt", "highland", "breccia"]  
rockTypes
```

A list with rock names and the number of that rock found

```
rockTypeAndCount = ["basalt", 1, "highland", 2.5, "breccia", 5]  
rockTypeAndCount
```

```
rockTypes.append("soil")  
rockTypes
```

```
rockTypes.pop()  
rockTypes
```

```
rockTypes[0]  
rockTypes[2]
```

```
rockTypes[2] = "soil"  
rockTypes[2]
```

Variables

	Name	Type	Size	Value
	rockTypeAndCount	list	6	['basalt', 1, 'highland', 2.5, 'breccia', 5]
	rockTypes	list	3	['basalt', 'highland', 'breccia']

[5] rockTypes

['basalt', 'highland', 'breccia', 'soil']

[8] rockTypes[0]

'basalt'

[9] rockTypes[2]

'breccia'

[14] rockTypes[2]

'soil'

Print 打印方法

```
numBasalt = 4  
print("The number of Basalt rocks found:", numBasalt)
```

```
date = "February 26"  
numRocks = 15  
print("On", date, "number of rocks found:", numRocks)
```

```
[16] print("The number of Basalt rocks found:", numBasalt)
```



```
The number of Basalt rocks found: 4
```

```
[19] print("On", date, "number of rocks found:", numRocks)
```



```
On February 26 number of rocks found: 15
```

知识检查

1. 以下哪一项不是变量类型？

- ☐ 整数
- ☐ 浮点型
- ☐ 字符串
- ☐ 小数

2. 哪项陈述是正确的？

- ☐ 列表只能存储一种类型的数据。
- ☐ 列表可以存储所有类型的数据。

3. 以下哪种方法是输出变量和文本的值的正确方法？

- ☐ `print("This is a variable"; variable)`
- ☐ `print("This is a variable", variable)`
- ☐ `print("This is a variable": variable)`
- ☐ `print("This is a variable"~ variable)`

真伪判断

```
temp = 50
print(temp >= 32)
print(temp < 32)
```

```
rock = "basalt"
print("highland" == rock)
print("basalt" == rock)
```

```
rock = "basaltrock"
print("highland" in rock)
print("basalt" in rock)
```

```
[20] temp = 50
```



```
[21] print(temp >= 32)
```



```
True
```

```
[22] print(temp < 32)
```



```
False
```

```
[26] rock = "basaltrock"
```



```
[27] print("highland" in rock)
```



```
False
```

```
[28] print("basalt" in rock)
```



```
True
```

```
[23] rock = "basalt"
```



```
[24] print("highland" == rock)
```



```
False
```

```
[25] print("basalt" == rock)
```



```
True
```

条件判断

```
basalt = 0
if(basalt == 0):
    print("We have found no basalt rocks.")
print("Done checking basalt rocks.")
```

```
basalt = 1
if(basalt == 0):
    print("We have found no basalt rocks.")
print("Done checking basalt rocks.")
```

```
[31] ▶ if(basalt == 0):...
```



```
We have found no basalt rocks.
Done checking basalt rocks.
```

```
[32] ▶ basalt = 1...
```



```
Done checking basalt rocks.
```

条件判断

```
basalt = 1
if(basalt == 0):
    print("We have found no basalt rocks.")
else:
    print("We have found some basalt rocks!")
print("Done checking basalt rocks.")
```

```
[33] ▶ basalt = 1...
```



```
We have found some basalt rocks!
Done checking basalt rocks.
```

条件判断

```
basalt = 1
if(basalt == 0):
    print("We found no basalt rocks.")
elif(basalt == 1):
    print("We found exactly 1 basalt rock.")
else:
    print("We found more than 1 basalt rock!")
print("Done checking basalt rocks.")
```

```
[35] ▶ basalt = 1...
```



```
We found exactly 1 basalt rock.
Done checking basalt rocks.
```

while 回圈 – 火箭发射倒数

```
countdown = 5
```

```
while countdown >= 0:  
    print(countdown)  
    countdown = countdown - 1  
print("Lift Off")
```

Q. 请问执行完成后，
countdown的值为何？

```
[36] ▶ countdown = 5...
```



5
4
3
2
1
0
Lift Off

for 回圈 - 巡访所有星球

```
planets = "Mars", "Saturn", "Jupiter"
```

```
for planet in planets:  
    print(planet)
```

```
[37] ▶ planets = "Mars", "Saturn", "Jupiter"...
```



```
Mars  
Saturn  
Jupiter
```

函数 - 重构火箭发射

```
def launchRocket():  
    countdown = 5  
    while countdown >= 0:  
        print(countdown)  
        countdown = countdown - 1  
    print("Lift Off")
```

```
launchRocket()
```


档案读取

```
strPath = "text.txt"
fileObject = open(strPath)
textList = fileObject.readlines()
fileObject.close()
```

```
for line in textList:
    print(line)
```



First Astronaut on the moon
Neil Armstrong

text.txt

```
[43] ▶ for line in textList:...
```



First Astronaut on the moon

Neil Armstrong

全域 VS. 区域

- 函数内可读取全域变量

```
rocketText = "We will launch in"  
def OutputRocketText():  
    print(rocketText + " two days")  
    return
```

OutputRocketText()

全域 VS. 区域

- 函数内可读取全域变量，但是不能改写

```
rocketText = "We will launch in"  
def OutputRocketText():  
    rocketText = rocketText + " two days."  
    print(rocketText)  
    return
```

OutputRocketText()

```
2  
3 def OutputRocketText():  
----> 4     rocketText = rocketText + " two days."  
5     print(rocketText)  
6     return
```

UnboundLocalError: local variable 'rocketText' referenced before assignment

全域 VS. 区域

- 函数内可读取全域变量，但是不能改写
- 解决方式：告知为全域变量

```
rocketText = "We will launch in"  
def OutputRocketText():  
    global rocketText  
    rocketText = rocketText + " two days."  
    print(rocketText)  
    return
```

OutputRocketText()

```
[1] ▶ rocketText = "We will launch in"...
```



```
× We will launch in two days.
```

全域 VS. 区域

- 函数内可读取全域变量，但是不能改写
- 另一种解决方式：传入值

```
def OutputRocketText(textInput):  
    textInput = textInput + " two days."  
    return textInput
```

```
rocketText = "We will launch in"  
newRocketText = OutputRocketText(rocketText)  
print(newRocketText)
```

```
[3] ▶ def OutputRocketText(textInput):...
```



```
× We will launch in two days.
```

知识检查

1. 函数的用途是什么？

- ☐ 使其他人更难阅读你的代码
- ☐ 使代码看起来更复杂
- ☐ 降低代码冗余度并提升简洁性

2. 有哪两种方法可以更改不是在函数中创建的变量？

- ☐ 将其设置为全局变量或将其作为参数传递。
- ☐ 在函数外部创建变量或将其作为参数传递。

3. if 语句的正确代码是什么？

- ☐

```
if(condition):  
    do something
```
- ☐

```
if(condition);  
    do something
```
- ☐

```
if(condition):  
do something
```
- ☐

```
if(condition);  
do something
```

计算各种岩石数量

- 读档 -> 逐行取得数据 -> 关档

```
print("Artemis Rover Rock Scanner Starting")
```

```
basalt = 0
```

```
breccia = 0
```

```
highland = 0
```

```
regolith = 0
```

```
rockList = []
```

```
strPath = "rocks.txt"
```

```
fileObject = open(strPath)
```

```
rockList = fileObject.readlines()
```

```
for rock in rockList:
```

```
    print(rock)
```

```
fileObject.close()
```

```
[7] rockList
[8] for rock in rockList:...
```

```
['basalt\n',
 'breccia\n',
 'highland\n',
 'regolith\n',
 'highland\n',
 'breccia\n',
 'highland\n',
 'regolith\n',
 'regolith\n',
 'regolith\n',
 'basalt\n',
 'highland\n',
 'basalt\n',
 'breccia\n',
 'breccia\n',
 'regolith\n',
 'breccia\n',
 'highland\n',
 'highland\n',
 'breccia\n',
 'basalt']
```

```
basalt
breccia
highland
regolith
highland
breccia
highland
regolith
regolith
basalt
highland
basalt
breccia
breccia
regolith
breccia
highland
highland
breccia
basalt
```


计算各种岩石数量

- 逐项呼叫函式->比对查找->纪录

```
def countMoonRocks(rockToID):  
    global basalt,breccia,highland,regolith  
    rockToID = rockToID.lower()  
    if("basalt" in rockToID):  
        print("Found a basalt\n")  
        basalt += 1  
    elif("breccia" in rockToID):  
        print("Found a breccia\n")  
        breccia += 1  
    elif("highland" in rockToID):  
        print("Found a highland\n")  
        highland += 1  
    elif("regolith" in rockToID):  
        print("Found a regolith\n")  
        regolith += 1  
    return  
  
for rock in rockList:  
    countMoonRocks(rock)
```

```
[10]> def countMoonRocks(rockToID):...
```



Found a basalt

Found a breccia

Found a highland

Found a regolith

Found a highland

Found a breccia

Found a highland

Found a regolith

Found a regolith

Found a basalt

Found a highland

Found a basalt

Found a breccia

Found a breccia

Found a regolith

Found a breccia

Found a highland

Found a highland

Found a breccia

Found a basalt

计算各种岩石数量

- 确认结果、找出最大与最小

```
print("Number of Basalt: ", basalt)
print("Number of breccia: ", breccia)
print("Number of highland: ", highland)
print("Number of regolith: ", regolith)

print("The max number of one type of rock was:",
      max(basalt, breccia, highland, regolith))
print("The minimum number of one type of rock was:",
      min(basalt, breccia, highland, regolith))
```

```
[12] ▶ print("Number of Basalt: ", basalt)...
```



```
Number of Basalt: 4
Number of breccia: 6
Number of highland: 6
Number of regolith: 4
The max number of one type of rock was: 6
The minimum number of one type of rock was: 4
```

计算各种岩石数量

· 重构

```
rock_category = {"basalt":0, "breccia":0, "highland":0, "regolith":0}
```

```
def countMoonRocksV2(rock_list):  
    for thisRock in rock_list:  
        for is_this_category in rock_category.keys():  
            if is_this_category in thisRock:  
                rock_category[is_this_category]+=1  
    return
```

```
countMoonRocksV2(rockList)
```

```
import pandas as pd  
rock_df = pd.DataFrame({'count':rock_category})  
rock_df.sort_values(by='count',ascending=False)
```

```
[46] rock_category
```



```
{'basalt': 4, 'breccia': 6, 'highland': 6, 'regolith': 4}
```

```
[49] rock_df.sort_values(by='count',ascending=False)
```



	count
breccia	6
highland	6
basalt	4
regolith	4

知识检查

1. 我们如何修改我们创建的函数中的岩石计数变量？

- ☐ 我们将它们作为参数传递。
- ☐ 我们将其设置为全局变量。

2. 如何调用 countMoonRocks 函数？

- ☐ for rock in rockList: countMoonRocks(rock)
- ☐ for rockList in rock: countMoonRocks(rockList)
- ☐ while rock: countMoonRocks(rock)
- ☐ while rockList: countMoonRocks(rock)

3. 从 rocks.txt 文件读取完数据后，我们调用了什么代码？

- ☐ `.finish()`
- ☐ `.close()`
- ☐ `.end`
- ☐ `.done()`

获得奖杯

<https://docs.microsoft.com/zh-cn/users/XXXXXX/achievements>



Microsoft

Docs

概述

活动

挑战

书签

集合

关注

成就

设置



奖杯

了解 Python 在太空探索中扮演的角色

小结

- Python程序在各行各业都有应用机会，本次探讨于太空探索相关应用
- 在函式中若无法直接变更全局变量，需进行全局指定宣告
- 尝试重构、尝试套用别种方法进行比较





Reactor



developer.microsoft.com/reactor/
[@MSFTReactor](https://twitter.com/MSFTReactor) on Twitter

议程结束 感谢聆听



请记得填写课程回馈问卷 (Event ID : **11892**)
<https://aka.ms/Reactor/Survey>

© 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.