

Problem-Based Learning

Python 3

Lecture: 01-variable

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



Problem

- Write a program which prompts the user for a Celsius temperature, convert the temperature to Fahrenheit, and print out the converted temperature.

- 顯示提示
- 輸入數字
- 計算

$$F = C \times \frac{9}{5} + 21$$

- 顯示結果

顯示

- `print(3.14)`

- 3.14

- `print("Temperature C")`

- Temperature C

- `print(2*3.14)`

- 6.28

- `print(30*9/5+32)`

- 86.0

變數使用

- `a = 3.14`

- `print(a)`

- 3.14

- `a = "Temperature C"`

- `print(a)`

- Temperature C

- `a = 30*9/5+32`

- `print(a)`

- 86.0

變數使用

- `a = "Temperature:"`
- `b = 30`
- `c = b * 9/5 + 32`
- `print(a+"C=",b)`
- `print(a+"F=",c)`

- `Temperature:C=30`
- `Temperature:F=86.0`

輸入

- `input("Temperature C:")`
 - `a = "Temperature C:"`
 - `C = input(a)`
 - `print(C)`
 - `print(C+32)`
 - `print(type(C))`
- Temperature C:
 - Temperature C:
 - ERROR!
 - `<class 'str'>`

術語

電腦：

- 字串(string)
- 浮點數(float)

人：

- 文字(text)
- 實數(real number)

輸入

- `a = "Temperature C"`
- `C = input(a)`
- `C = float(C)`
- `print(C+32)`

- Temperature C
- 30 ← 輸入
- 62.0

輸入

- `a = "Temperature C"`
- `C = input(a)`
- `try:`
 - `C = float(C)`
 - `print(C+32)`
 - `except:`
 - `print("ERROR!")`

4個空格或是Tab

- Temperature C
- 30
- 62.0

Problem

- Write a program which prompts the user for a Celsius temperature, convert the temperature to Fahrenheit, and print out the converted temperature.

- 顯示提示
- 輸入數字
- 計算

$$F = C \times \frac{9}{5} + 21$$

- 顯示結果

Problem-Based Learning

Python 3

Lecture: 02-if & while

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



條件句

- Write a program to prompt for a score. Print a grade using the following table:

Score Grade ≥ 0.9 A

≥ 0.8 B

≥ 0.7 C

≥ 0.6 D

< 0.6 F

- 顯示提示
- 輸入數字
- 判斷範圍
- 顯示結果

判斷 if

- `a = 0.95` `> A`
- `if a >= 0.9:`
- `print("A")`

判斷 if ... else ...

- `a = 0.85` > F
- `if a >= 0.9:`
- `print("A")`
- `else:`
- `print("F")`

判斷 if ... else ...

- `a = 0.85` > CORRECT
- `if a > 0.0 and a < 1.0:`
- `print("CORRECT")`
- `else:`
- `print("ERROR")`

Problem (自我練習)

- Write a program to prompt for a score between 0.0 and 1.0. **First check: if the score is out of range, print an error message.** Then if the score is between 0.0 and 1.0, print a grade using the following table:
Score Grade ≥ 0.9 A
 < 0.9 F

- 顯示提示
- 輸入數字
- 第一步先判斷範圍
 $x \geq 0.9$
- 第二步判斷成績
A, F
- 顯示結果

Solution

- `C = input("Enter score := ")`
- `try:`
- `a = float(C)`
- `if a > 0.0 and a < 1.0:`
- `if a >= 0.9:`
- `print("A")`
- `else:`
- `print("F")`
- `else:`
- `print("Bad score")`
- `except:`
- `print("Not a number!")`

判斷 if ... elif ... else ...

- `a = 0.85`
- `if a >= 0.9:`
- `print("A")`
- `else:`
- `if a >= 0.8:`
- `print("B")`
- `else:`
- `print("F")`

- `a = 0.85`
- `if a >= 0.9:`
- `print("A")`
- `elif a >= 0.8:`
- `print("B")`
- `else:`
- `print("F")`

判斷 if ... elif ... else ...

- a = 0.75 > C
- if a >= 0.9:
 - print("A")
- elif a >= 0.8:
 - print("B")
- elif a >= 0.7:
 - print("C")
- else:
 - print("F")

Problem

- Write a program to **repeatedly** prompt for a score between 0.0 and 1.0. until the user enters “done”. If the score is out of range, print an error message. If the score is between 0.0 and 1.0, print a grade using the following table:

Score	Grade
≥ 0.9	A
≥ 0.8	B
≥ 0.7	C
≥ 0.6	D
< 0.6	F

- 重覆顯示提示

條件式迴圈 while

```
• a = 0.95 > A
• while a >= 0.9: > A
    print("A") > A
    > A
    > A
    > A
    > A
    > A
    > A
    > A
```

條件式迴圈 while ... break

- `a = 0.95` `> A`
- `while a >= 0.9:`
- `print("A")`
- `break`

條件式迴圈 while

- `while True:`
- `C = input("Enter score (or 'done' to finish) := ")`
- `if C.lower() == "done":`
- `break`
- `C = float(C)`
- `print(C)`

Problem

- Write a program to repeatedly prompt for a score between 0.0 and 1.0. until the user enters “done”. Once “done” is entered, print out the total, count, and average of the numbers. If the score is out of range, print an error message. If the score is between 0.0 and 1.0, print a grade using the following table:

Score	Grade
≥ 0.9	A
≥ 0.8	B
≥ 0.7	C
≥ 0.6	D
< 0.6	F

Problem-Based Learning

Python 3

Lecture: 3-function

Online Python

- <https://python-fiddle.com>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



變數(variable)和函數(function)

- 變數

```
c = 1.0
```

- 一般函數

```
y = input(1.0)  
print(y)
```

- 數學函數

```
y = float(60)  
y = int(3.14)  
y = round(3.14159265, 2)
```

數學函數(function)

- `import numpy as np`
- `x = np.linspace(-np.pi, np.pi, 100)`
- `y = np.cos(x)`
- `print(y)`

x 在 $-\pi$ 到 π 之間取100個元素

將產生的 x 元素，代入`cos()`產生對應 y 值

數學函數(function)

- `import matplotlib.pyplot as plt`
- `import numpy as np`
- `x = np.linspace(-np.pi, np.pi, 100)` # x 在 $-\pi$ 到 π 之間取100個元素
- `y = np.cos(x)` # 將產生的 x 元素，代入`cos()`產生對應 y 值
- `plt.plot(x, y)`
- `plt.show()`

Matplotlib

- `import matplotlib.pyplot as plt`
- `import numpy as np`
- `x = np.linspace(-np.pi, np.pi, 100)`
- `y = np.cos(x)`
- `plt.plot(x, y, label="Cosine")` `# Legend "Cosine"`
- `plt.legend(loc="upper left")`
- `plt.show()`

Matplotlib

- `import matplotlib.pyplot as plt`
- `import numpy as np`
- `x = np.linspace(-np.pi, np.pi, 100)`
- `y = np.cos(x)`
- `plt.plot(x, y, label="Cosine")`
- `plt.legend(loc="upper left")`
- `plt.xlabel(r"$\theta$")`
- `plt.ylabel(r"$\cos(\theta)$")`
- `plt.show()`

Matplotlib

- `import matplotlib.pyplot as plt`
- `import numpy as np`
- `y =`
`[1320,1283,1274,1291,1339,1412,1502,1602,1699,1782,1839,1866`
`,1853,1805,1729,1637,1541,1446,1363,1305]`
- `x = range(0, len(y), 1)`
- `plt.scatter(x, y, label="Data")`
- `plt.legend(loc="best", fontsize=14)`
- `plt.xlabel("Frame number", fontsize=14)`
- `plt.ylabel("Pixel position", fontsize=14)`
- `plt.show()`

Matplotlib

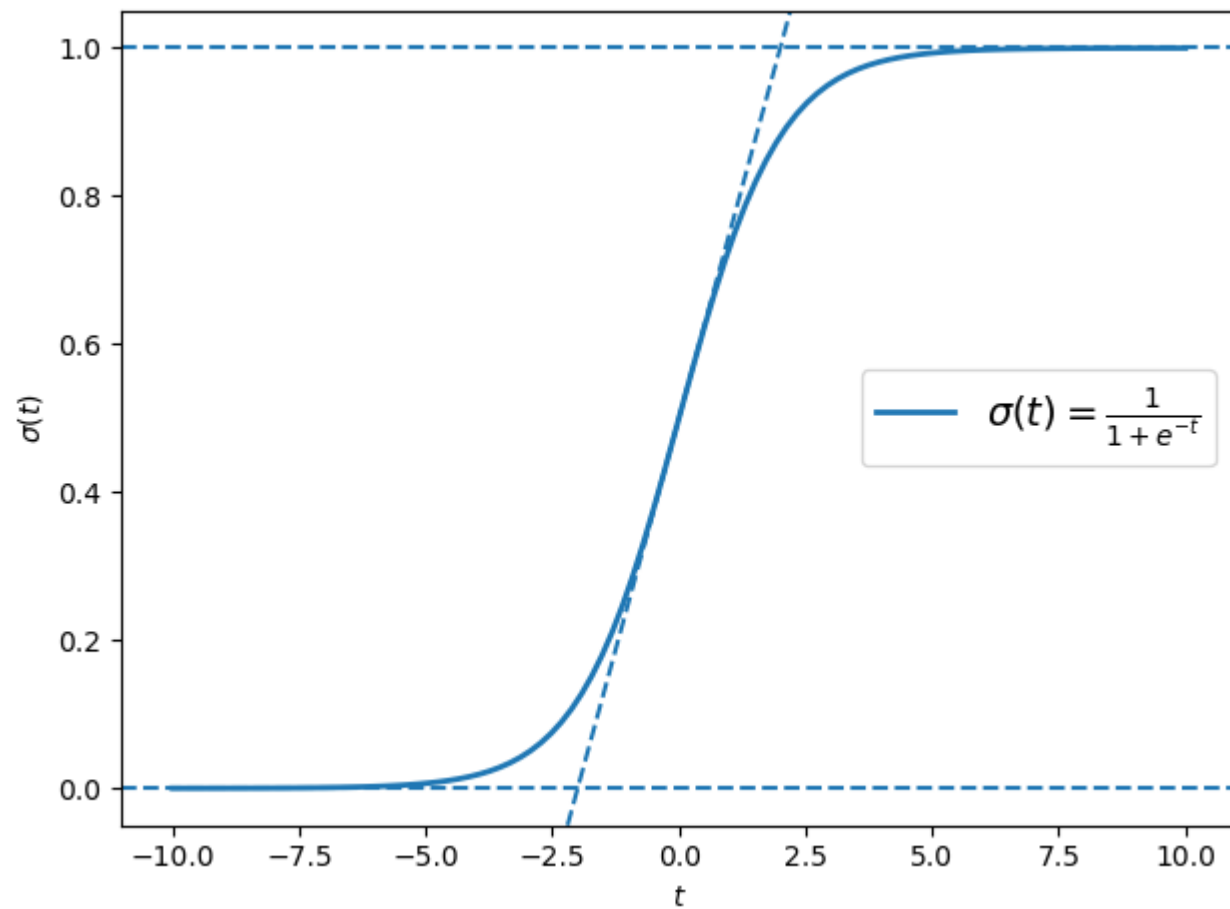
- `import matplotlib.pyplot as plt`
- `import numpy as np`
- `y =`
`[1320,1283,1274,1291,1339,1412,1502,1602,1699,1782,1839,1866,1853,1805,1729,1637`
`,1541,1446,1363,1305]`
- `x = range(0, len(y), 1)`
- `plt.axhline(y=1400, linestyle="--")`
- `plt.axhline(y=1700, linestyle="--")`
- `plt.axline((7, 1602), slope=98, linestyle="--")`
- `plt.scatter(x, y, label="Data")`
- `plt.legend(loc="best", fontsize=14)`
- `plt.xlabel("Frame number")`
- `plt.ylabel("Pixel position")`
- `plt.show()`

自訂函數

- `import numpy as np`
- `def f(x,y):`
- `return (1-x/2+x**5+y**3)*np.exp(-x**2-y**2)`
- `n = 16`
- `x = np.linspace(-3, 3, n)`
- `y = np.linspace(-3, 3, n)`
- `print(f(x,y))`

Problem

- 使用matplotlib畫出下邊的圖



Problem-Based Learning

Python 3

Lecture: 04-for

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



向量 vector

- $a = 0.95$
- 變數只能放一個數字

- 向量有 3 個變數

$$a = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$$

- 把 3 個變數放在一起，需要一個盒子

- `import numpy as np`
- `a = np.array([1,2,3])`

- `print(a)`

盒子(人類可以理解的形式)

盒子(將人類可以理解的形式轉成電腦的形式)

$$a = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

任意維度的向量

- Write a program to **repeatedly** prompt for a number between 0 and 10. until the user enters “done”. **Once “done” is entered.** print the vector.

$$\begin{array}{lcl} b = () & & \\ \downarrow & & \\ b = (b_0) & \leftarrow \text{np.append} & \\ \downarrow & & \\ b = \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} & \leftarrow \text{np.append} & \\ \downarrow & & \\ b = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \end{pmatrix} & \leftarrow \text{np.append} & \end{array}$$

附加元素np.append

- `import numpy as np`
- `b = np.array([])`
- `while True:`
- `C = input("Enter number (or 'done' to finish)= ")`
- `if C.lower() == "done":`
- `break`
- `try:`
- `a = float(C)`
- `if a >= 0 and a <= 10:`
- `b = np.append(b, a)`
- `except:`
- `print("Not a number!")`
- `print(b)`
- 向量**b**共有**N**個元素，編號從**0**開始到**N-1**結束。

$$b = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_{N-1} \end{pmatrix}$$

迴圈和編號

- Write a program to **repeatedly** prompt for a number between 0 and 10. until the user enters “done”. **Once “done” is entered, find the maximum number.** If the number is out of range, print an error message.
- 向量**b**共有**N**個元素，編號從**0**開始到**N-1**結束。

$$b = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_{N-1} \end{pmatrix}$$

For迴圈

- 附加底下的程式
 - `max = 0`
 - `for i in range(len(b)):`
 - `if max < b[i]:`
 - `max = b[i]`
 - `print(max)`
 -
- 向量的第*i*個元素

$$b = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_{N-1} \end{pmatrix} = \begin{pmatrix} b_{i=0} \\ b_{i=1} \\ b_{i=2} \\ \vdots \\ \vdots \\ b_{i=N-1} \end{pmatrix} \left. \vphantom{\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_{N-1} \end{pmatrix}} \right\} \text{共有 } N \text{ 個元素}$$

- 說明：
- `for i in range(len(b)):`
 ↑ └─┬─┘
 範圍 得到N
- **i**從**0**開始到**N-1**結束,對應向量個別元素的編號

向量大小 vector norm

- $\vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$
- $\vec{a}^2 = a_0 \times a_0 + a_1 \times a_1 + a_2 \times a_2$
- $|\vec{a}| = \sqrt{\vec{a}^2}$

- 附加底下的程式
- `sum = 0`
- `for i in range(3):`
- `sum = sum + a[i] * a[i]`
- `print(sum)`
- `print(np.sqrt(sum))`
- 請解決錯誤！

向量內積 dot product

$$\vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix}$$

- $\vec{a} \cdot \vec{b} = a_0 \times b_0 + a_1 \times b_1 + a_2 \times b_2$

- 附加底下的程式
- `a = np.array([0.1, 0.2, 0.3])`
- `sum = 0.0`
- `for i in range(3):`
- `sum = sum + a[i] * b[i]`
- `print(sum)`
- 請解決錯誤！

向量相加 vector addition

$$\vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix}$$

- $c_0 = a_0 + b_0$
- $c_1 = a_1 + b_1$
- $c_2 = a_2 + b_2$

- 附加底下的程式
- `a = np.array([0.1, 0.2, 0.3])`
- `c = np.zeros(3)`
- `for i in range(3):`
- `c[i] = a[i] + b[i]`
- `print(c)`

- 請解決錯誤！

術語

電腦：

- 陣列(array)

人：

- 向量(vector)

Problem

- Write a program to use **for** loop to calculate the cross product of two vectors:

$$\vec{a} = \begin{pmatrix} 0.5 \\ 0.3 \\ 0.4 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} 2 \\ 1 \\ 4 \end{pmatrix}, \quad \vec{c} = \vec{a} \times \vec{b} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix}$$

$$\vec{c} = \begin{vmatrix} \hat{x} & \hat{y} & \hat{z} \\ a_0 & a_1 & a_2 \\ b_0 & b_1 & b_2 \end{vmatrix} = (a_1 b_2 - a_2 b_1) \hat{x} + (a_2 b_0 - a_0 b_2) \hat{y} + (a_0 b_1 - a_1 b_0) \hat{z}$$

$$c[i] = a[j] * b[k] - a[k] * b[j]$$

Problem-Based Learning

Python 3

Lecture: 05-for (2)

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



矩陣 matrix

- 3 維矩陣有 9 個變數

$$\mathbf{a} = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}$$

- Python把 3 維矩陣看成 3 個向量組成

$$\mathbf{a} = \begin{pmatrix} \vec{a}_0 \\ \vec{a}_1 \\ \vec{a}_2 \end{pmatrix}, \quad \begin{aligned} \vec{a}_0 &= (a_{00} & a_{01} & a_{02}) \\ \vec{a}_1 &= (a_{10} & a_{11} & a_{12}) \\ \vec{a}_2 &= (a_{20} & a_{21} & a_{22}) \end{aligned}$$

術語

電腦：

- 陣列(array)

人：

- 向量(vector)
- 矩陣(matrix)
- 張量(tensor)

矩陣 matrix

- `import numpy as np`
- `a = np.array(
 [[1, 2, 3],
 [4, 5, 6],
 [7, 8, 9]]
)`
- `print(a)`
- `print(a[0][1])`
- `print(a[1][2])`

矩陣相加

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} + \begin{pmatrix} 0.3 & 0.5 & 0.7 \\ 0.1 & 0.2 & 0.3 \\ 0.2 & 0.3 & 0.4 \end{pmatrix}$$

矩陣相加 matrix addition

- `b = np.array(
 [[0.3, 0.5, 0.7],
 [0.1, 0.2, 0.3],
 [0.2, 0.3, 0.4]]
)`
- `c = np.zeros((3,3))`
- `for i in range(3):`
- `for j in range(3):`
- `c[i][j] = a[i][j] + b[i][j]`
- `print(c)`

✓ [[1.3 2.5 3.7]
✓ [4.1 5.2 6.3]
✓ [7.2 8.3 9.4]]

矩陣相加 matrix addition: 定義函數def

- `def f(a,b):`
- `c = np.zeros((3,3))`
- `for i in range(3):`
- `for j in range(3):`
- `c[i][j] = a[i][j] + b[i][j]`
- `return c`
- `print(f(a,b))`

Problem

- Write a program and define two functions which uses **for loop** to calculate

1. Matrix dot vector:

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{pmatrix} \begin{pmatrix} 0.3 \\ 0.5 \\ 0.7 \end{pmatrix}$$

2. The multiplication of two matrices:

$$\begin{pmatrix} 0.1 & 0.2 & 0.3 \\ 0.2 & 0.3 & 0.4 \\ 0.3 & 0.4 & 0.5 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{pmatrix}$$

Problem-Based Learning

Python 3

Lecture: 05-for-example

Online Python

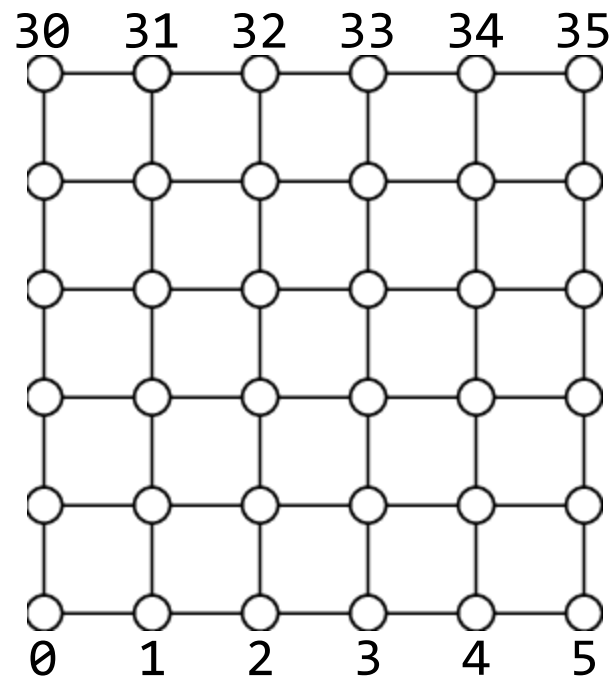
- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



Problem



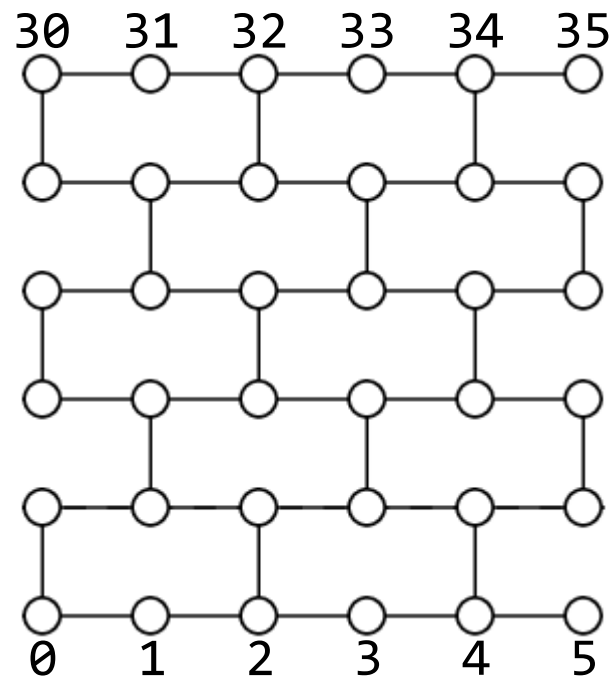
$0 \rightarrow (0, 0)$
 $1 \rightarrow (1, 0)$
 $2 \rightarrow (2, 0)$

$33 \rightarrow (3, 5)$
 $34 \rightarrow (4, 5)$
 $35 \rightarrow (5, 5)$

顯示結果

I	(x,y)	+x	-x	+y	-y
0	(0,0)	1	5	6	30
1	(1,0)	2	0	7	31
2	(2,0)	3	1	8	32
33	(3,5)	34	32	3	27
34	(4,5)	35	33	4	28
35	(5,5)	30	34	5	29

Problem



$0 \rightarrow (0, 0)$
 $1 \rightarrow (1, 0)$
 $2 \rightarrow (2, 0)$

$33 \rightarrow (3, 5)$
 $34 \rightarrow (4, 5)$
 $35 \rightarrow (5, 5)$

顯示結果

I	(x,y)	+x	-x	+y	-y
0	(0,0)	1	5	6	30
1	(1,0)	2	0	-1	-1
2	(2,0)	3	1	8	32
33	(3,5)	34	32	3	27
34	(4,5)	35	33	-1	-1
35	(5,5)	30	34	5	29

Problem-Based Learning

Python 3

Lecture: 07-list

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



Numpy陣列

- `import numpy as np`
- `a = np.array([1, 2 ,3])`
- `b = np.array([1, 'A', 3])`
- `print(a)`
- `print(a[1]+a[3])`
- `print(b)`
- `print(b[1]+b[3])`

- `[1, 2, 3]`
- `['1', 'A', '3']`

Numpy陣列不能同時存放字串(文字)和數字

需要一個能同時存放字串(文字)和數字的陣列

串列 **list**-能同時存放字串和數字的陣列

- `a = [1, 2, 'A', 4]`
- `print(a)`
- `print(a[0]+a[3])`
- `print(len(a))`

- `[1, 2, 'A', 4]`
- `5`
- `4`

練習：求最大值和最小值

- `a = [1, 3, 6, -13, 5, 67, -8]`
 - `max = 0`
 - `min = 0`
 - `for i in range(len(a)):`
 - `if max < a[i]:`
 - `max = a[i]`
 - `if min > a[i]:`
 - `min = a[i]`
 - `print(max,min)`
- 67 -13

練習：由小到大排序

- `a = [1, 3, 6, -13, 5, 67, -8]`
- `[-13, -8, 1, 3, 5, 6, 67]`
- 提示：
- `a[i], a[j] = a[j], a[i]`
- `for i in range(len(a)):`
- `for j in range(i):`
- `if a[i] < a[j]:`
- `a[i], a[j] = a[j], a[i]`
- `print(a)`

二維串列

- `a = [[96,65,73],
 [88,76,82],
 [92,84,89],
 [82,73,64],
 [70,83,68]]`

- `print(a)`

- `for i in range(5):`

- `print("學生", i+1, "的國文分為", a[i][0])`

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

二維串列

- `a = [[96,65,73],
[88,76,82],
[92,84,89],
[82,73,64],
[70,83,68]]`
- `for i in range(5):`
- `s = 0`
- `for j in range(3):`
- `s = s + a[i][j]`
- `print("學生",i+1,"總分為",a[i][3])`

Problem

- 五個學生國英數成績如下

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

```
[[96, 65, 73], [88, 76, 82], [92, 84, 89], [82, 73, 64], [70, 83, 68]]  
1 85.6  
2 76.2  
3 75.2  
  
1 78.0  
2 82.0  
3 88.33333333333333  
4 73.0  
5 73.66666666666667
```

- 計算各個學科平均分數
- 計算每一個學生的平均成績
- 找出每一個學生的最高分和最低分 and 其所對應的學科

Problem-Based Learning

Python 3

Lecture: 08-list (2)

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



串列(List)

- 一維串列List
- `a = [6, 2, 4, 3, 8, 9]`
- 二維串列List
- `a = [[96, 65, 73],`
- `[88, 76, 82],`
- `[92, 84, 89]]`

Problem

- 自製function
- 傳入list，傳出min

- `def min(a):`

-

-

- `return min`

- `a = [6, 2, 4, 3, 8, 9]`

- `print(min(a))`

- 自製function

- 傳入list，傳出max

- `def max(a):`

-

-

- `return max`

- `a = [6, 2, 4, 3, 8, 9]`

- `print(max(a))`

Solution

- `def min(a):`
- `min = a[0]`
- `for i in range(len(a)):`
- `if a[i] < min:`
- `min = a[i]`
- `return min`

- `a = [6, 2, 4, 3, 8, 9]`
- `print(min(a))`

- `def max(a):`
- `max = a[0]`
- `for i in range(len(a)):`
- `if a[i] > max:`
- `max = a[i]`
- `return max`

- `a = [6, 2, 4, 3, 8, 9]`
- `print(max(a))`

Problem

- 自製function
- 傳入list，傳出mean

```
• def mean(a):  
•  
•     return mean
```

```
• a = [6, 2, 4, 3, 8, 9]  
• print(mean(a))
```

$$s = \sqrt{\frac{\sum_i (\bar{x} - x_i)^2}{N - 1}}$$

- 自製function
- 傳入list和mean，傳出standard deviation

```
• def sd(a, mean):  
•  
•     return sd
```

```
• a = [6, 2, 4, 3, 8, 9]  
• print(sd(a, mean(a)))
```

Solution

- `def mean(a):`
- `mean = 0`
- `for i in range(len(a)):`
- `mean += a[i]`
- `return mean / len(a)`
- `a = [6, 2, 4, 3, 8, 9]`
- `print(mean(a))`

- `def sd(a, mean):`
- `sd = 0`
- `for i in range(len(a)):`
- `sd += (mean - a[i])**2`
- `sd = np.sqrt(sd / (len(a) - 1))`
- `return sd`
- `a = [6, 2, 4, 3, 8, 9]`
- `print(sd(a,mean(a)))`

Problem

- 五個學生國英數成績如下：

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

- 使用已經自製的function來計算每個學生的平均成績和標準差

$$s = \sqrt{\frac{\sum_i (\bar{x} - x_i)^2}{N - 1}}$$

- 將計算的平均值和標準差個別附加到原表格。

```
a[i].append(s)
```

- 最後結果顯示：

```
print(a)
```

Problem-Based Learning

Python 3

Lecture: 09-Pandas

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



串列

- 五個學生國英數成績如下，計算每個學生的平均成績
- 是否存在表格式的變數

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

Python list, Numpy array, Pandas DataFrame

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

✓ Numpy array:

```
[[96, 65, 73],  
 [88, 76, 82],  
 [92, 84, 89],  
 [82, 73, 64],  
 [70, 83, 68]]
```

✓ Python List:

```
[[96, 65, 73],  
 [88, 76, 82],  
 [92, 84, 89],  
 [82, 73, 64],  
 [70, 83, 68]]
```

✓ Pandas DataFrame:

	0	1	2 ← columns
index → 0	96	65	73
1	88	76	82
2	92	84	89
3	82	73	64
4	70	83	68

Python List → Numpy Array Python List → Pandas DataFrame

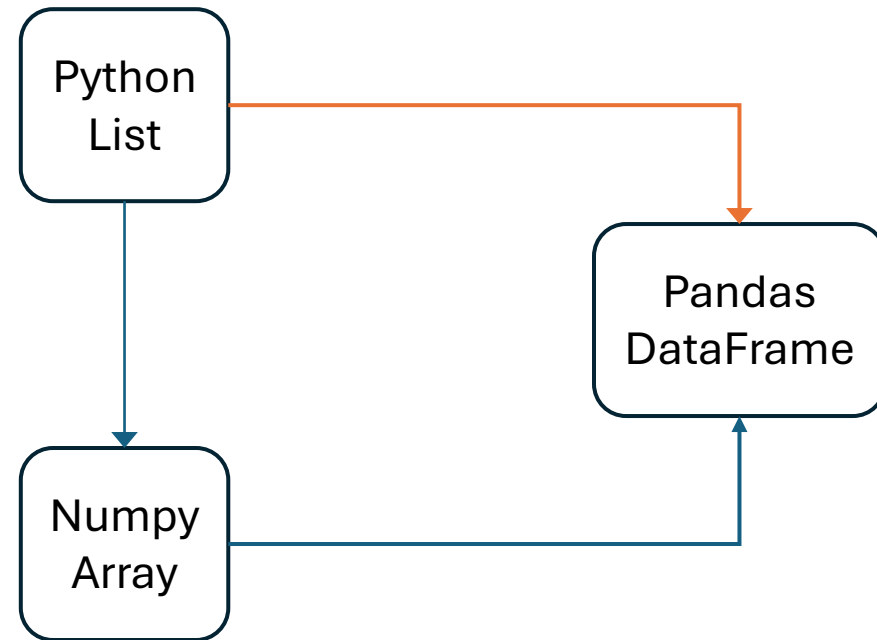
- `import numpy as np`
- `a = [[96, 65, 73],
 [88, 76, 82],
 [92, 84, 89],
 [82, 73, 64],
 [70, 83, 68]]`
- `b = np.array(a)`
- `print(b)`

- `import pandas as pd`
- `a = [[96, 65, 73],
 [88, 76, 82],
 [92, 84, 89],
 [82, 73, 64],
 [70, 83, 68]]`
- `d = pd.DataFrame(a)`
- `print(d)`

Numpy Array → Pandas DataFrame

- `import pandas as pd`
- `import numpy as np`
- `a = np.array(
 [[96, 65, 73],
 [88, 76, 82],
 [92, 84, 89],
 [82, 73, 64],
 [70, 83, 68]]
)`
- `d = pd.DataFrame(a)`

- `print(d)`



Pandas DataFrame : index & columns

- `import pandas as pd`
- `a = [[96,65,73],
[88,76,82],
[92,84,89],
[82,73,64],
[70,83,68]]`
- `ind = ["1", "2", "3", "4", "5"]`
- `col = ["國文", "英文", "數學"]`
- `d = pd.DataFrame(a, index=ind, columns=col)`
- `print(d)`

The diagram illustrates the structure of a Pandas DataFrame. A box labeled 'columns' has an arrow pointing to the column headers of a table. A box labeled 'index' has an arrow pointing to the row indices of the same table.

		國文	英文	數學
index	1	96	65	73
	2	88	76	82
	3	92	84	89
	4	82	73	64
	5	70	83	68

Pandas DataFrame : index & columns

- `d.index=["1", "2", "3", "4", "5"]`
- `d.columns=["國文", "英文", "數學"]`
- `print(d)`
- `print(d.shape)`
- `print(d.shape[0])`
- `print(d.shape[1])`

column長度 shape(1)	columns					
		↓	國文	英文	數學	
	index	→	1	96	65	73
		↓	2	88	76	82
index長度 shape(0)			3	92	84	89
			4	82	73	64
			5	70	83	68

Problem

- 五個學生國英數成績如下：
- 按照平均成績的大小**排序**

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

- 使用Pandas
- 僅使用for迴圈來計算平均成績
僅使用for迴圈來進行大小的排序
- 使用iloc[i][j]來存取特定位置的元素

Problem-Based Learning

Python 3

Lecture: 10-Pandas (2)

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>

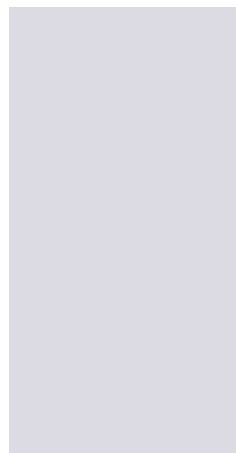


串列

- 五個學生國英數成績如下，計算每個學生的平均成績，並附加上原表格。

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

+



串列(List)、矩陣(Numpy)、表格(Pandas)

- 串列List

```
[[96, 65, 73], [88, 76, 82], [92, 84, 89]]
```

- 矩陣Numpy array

```
[[96 65 73]  
 [88 76 82]  
 [92 84 89]]
```

- 表格Pandas DataFrame

	A	B	C ← columns
a	96	65	73
b	88	76	82
c	92	84	89

↑
index

Python list → Pandas DataFrame

- `import pandas as pd`
- `a = [[1, 2, 3],
 [6, 7, 8],
 [2, 3, 4]]`
- `ind = ['a', 'b', 'c']`
- `col = ['A', 'B', 'C']`
- `d = pd.DataFrame(a, index=ind, columns=col)`
- `print(d)`

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4

Pandas DataFrame：增加直行單筆資料

- `b = d`
- `c = [7, 8, 9]`
- `b['D'] = c`
- `print(b)`

	A	B	C	D
a	1	6	2	7
b	2	7	3	8
c	3	8	4	9

Pandas DataFrame：增加橫列單筆資料

- `b = d`
- `c = [4, 9, 5]`
- `b.loc['d'] = c`
- `print(b)`

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4
d	4	9	5

Pandas DataFrame：增加直行多筆資料

- `b = d`
- `a2 = [[7,3],
[8,4],
[9,5]]`
- `col2 = ['D', 'E']`
- `d2 = pd.DataFrame(a2, index=ind, columns=col2)`
- `print(d2)`
- `b = pd.concat([b,d2], axis=1)`
- `print(b)`

	A	B	C	D	E
a	1	6	2	7	3
b	2	7	3	8	4
c	3	8	4	9	5

← axis=1

↑ axis=0

Pandas DataFrame：增加橫列多筆資料

- `b = d`
- `a2 = [[4,9,5],
 [5,1,6]]`
- `ind2 = ['d', 'e']`
- `d2 = pd.DataFrame(a2, index=ind2, columns=col)`
- `b = pd.concat([b,d2], axis=0)`
- `print(b)`

	A	B	C	← axis=1
a	1	6	2	
b	2	7	3	
c	3	8	4	
d	4	9	5	
e	5	1	6	

↑
axis=0

Pandas DataFrame : 計算min, max, mean

axis=0

- `c = d['A'].min()`
- `c = d['A'].max()`
- `c = d['A'].mean()`

axis=1

- `c = d.mean(axis=1)`
- `print(c)`
- `print(c.index)`
- `print(c.values)`

	A	B	C	← axis=1
a	1	6	2	
b	2	7	3	
c	3	8	4	

↑
axis=0

a	3.0
b	4.0
c	5.0

↑ ↑
index values

Pandas Series to DataFrame

- `b = d`
- `c = b.mean(axis=1)`
- `print(c)`
- `print(c.index)`
- `print(c.values)`

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4

axis=1

axis=0

a	3.0
b	4.0
c	5.0

index values

Pandas series to Pandas DataFrame

- `c = pd.DataFrame({"平均":c.values}, index=c.index)`
- `print(c)`

Problem

- 計算每個學生的平均成績，並附加到原表格。
- 按照學生的平均成績，由大到小排序

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68



平均

提示：
`b.min(axis=?)` `b.max(axis=?)` `b.mean(axis=?)`
`sort_values(by='欄位名稱', ascending=True 或 False)`

Problem-Based Learning

Python 3

Lecture: 12-dictionary

Online Python

- <https://www.programiz.com/python-programming/online-compiler/>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



Python list and Python dictionary

✓ Python list:

```
[[96, 65, 73], [88, 76, 82], [92, 84, 89]]
```

✓ Python dict:

```
{'a':[96, 65, 73], 'b':[88, 76, 82], 'c':[92, 84, 89]}
```

Python list → Pandas DataFrame

- `import pandas as pd`
- `a = [[1, 2, 3],
 [6, 7, 8],
 [2, 3, 4]]`
- `ind = ['a', 'b', 'c']`
- `col = ['A', 'B', 'C']`
- `d = pd.DataFrame(a, index=ind, columns=col)`
- `print(d)`

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4

Python dict (column-like) → Pandas DataFrame

- `import pandas as pd`
- `a = {
 'A': [1, 2, 3],
 'B': [6, 7, 8],
 'C': [2, 3, 4]
}`
- `print(a)`
- `ind = ['a', 'b', 'c']`
- `d = pd.DataFrame(a, index=ind)`
- `print(d)`

keys() values()
↓ ↓
{ 'A': [1, 2, 3],
 'B': [6, 7, 8],
 'C': [2, 3, 4] }

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4

Python dict (row-like) → Pandas DataFrame

- `import pandas as pd`

- `a = {
 'a': [1, 6, 2],
 'b': [2, 7, 3],
 'c': [3, 8, 4]
}`

- `col = ['A', 'B', 'C']`

- `d = pd.DataFrame(a.values(), index=a.keys(), columns=col)`

- `print(d)`

`keys()` `values()`

↓ ↓

`{ 'a': [1, 6, 2], 'b': [2, 7, 3], 'c': [3, 8, 4] }`

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4

Pandas DataFrame：增加直行單筆資料

- `b = d`
- `c = [7, 8, 9]`
- `b['D'] = c`
- `print(b)`

	A	B	C	D
a	1	6	2	7
b	2	7	3	8
c	3	8	4	9

Pandas DataFrame：增加橫列單筆資料

- `b = d`
- `c = [4, 9, 5]`
- `b.loc['d'] = c`
- `print(b)`

	A	B	C
a	1	6	2
b	2	7	3
c	3	8	4
d	4	9	5

Pandas DataFrame：增加直行多筆資料

- `b = d`
- `a2 = {
 'D': [7, 8, 9],
 'E': [3, 4, 5]
}`
- `ind = ['a', 'b', 'c']`
- `d2 = pd.DataFrame(a2, index=ind)`
- `b = pd.concat([b, d2], axis=1)`
- `print(b)`

	A	B	C	D	E	← axis=1
a	1	6	2	7	3	
b	2	7	3	8	4	
c	3	8	4	9	5	

↑
axis=0

Pandas DataFrame：增加直行多筆資料

- `b = d`
- `a2 = {`
 `'a': [7, 3],`
 `'b': [8, 4],`
 `'c': [9, 5],`
 `}`
- `col2 = ['D', 'E']`
- `d2 = pd.DataFrame(a2.values(), index=c.keys(), columns=col2)`
- `b = pd.concat([b, d2], axis=1)`
- `print(b)`

	A	B	C	D	E	← axis=1
a	1	6	2	7	3	
b	2	7	3	8	4	
c	3	8	4	9	5	

↑
axis=0

Pandas DataFrame：增加橫列多筆資料

- `b = d`
- `a2 = {
 'A':[4,5],
 'B':[9,1],
 'C':[5,6]
}`
- `ind2 = ['d', 'e']`
- `d2 = pd.DataFrame(a2, index=ind2)`
- `b = pd.concat([b,d2], axis=0)`
- `print(b)`

Pandas DataFrame：增加橫列多筆資料

- `b = d`
- `a2 = {
 'd':[4,9,5],
 'e':[5,1,6],
}`
- `d2 = pd.DataFrame(a2.values(), index=c.keys(), columns=col)`
- `b = pd.concat([b,d2], axis=0)`
- `print(b)`

Problem

- 計算每個學生的平均成績，並附加到原表格。
- 找出每一個學生的最高分和最低分 and 其所對應的學科，並附加到原表格。

	國文	英文	數學
學生1	96	65	73
學生2	88	76	82
學生3	92	84	89
學生4	82	73	64
學生5	70	83	68

+

平均

+

最高分

+

最低分

提示：
b.min(axis=?)
b.max(axis=?)
b.mean(axis=?)

b.idxmin(axis=?)
b.idxmax(axis=?)

Problem-Based Learning

Python 3

Lecture: 13-crawler

Online Python

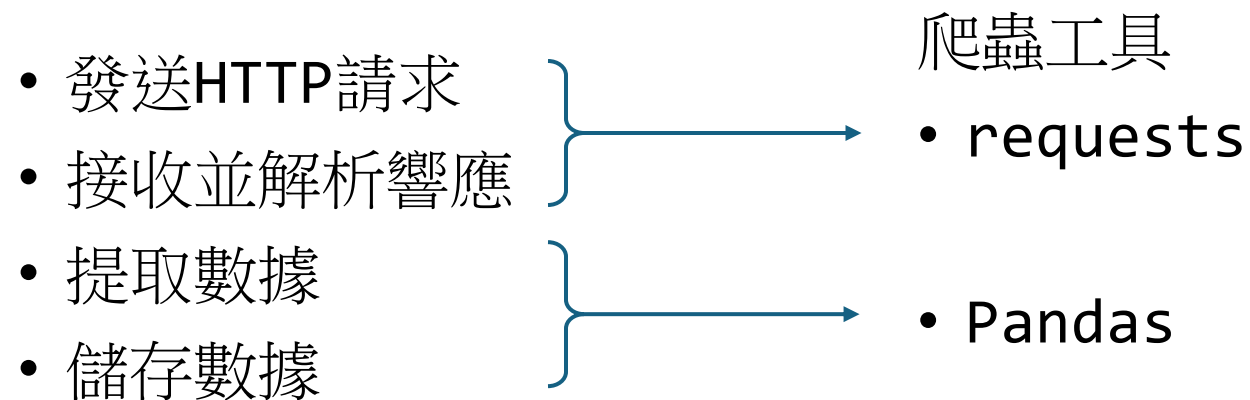
- <https://colab.research.google.com/>
- <https://www.onlinegdb.com/#>

Lecture Notes

- <https://web.phy.ntnu.edu.tw/~hongyi/?url=notes>



爬蟲



- 數據格式：

JSON

HTML

CSV

Accurate Weather Forecasts

- <https://open-meteo.com/>

`https://api.open-meteo.com/v1/forecast?latitude=25.0330&longitude=121.5654&daily=temperature_2m_max,temperature_2m_min`

URL解析

`https://api.open-meteo.com/v1/forecast?latitude=25.0330&longitude=121.5654&daily=temperature_2m_max,temperature_2m_min`

- `url = "https://api.open-meteo.com/v1/forecast"`
- `params = {
 "latitude": 25.0330,
 "longitude": 121.5654,
 "daily": "temperature_2m_max,temperature_2m_min"
}`

Request

- `response = requests.get(url, params=params)`
- `data = response.json()`
- `print(data)`  `dict in Python`

```
{"latitude":25,"longitude":121.625,"generationtime_ms":0.0259876251220703
,"utc_offset_seconds":28800,"timezone":"Asia/Taipei","timezone_abbreviati
on":"GMT+8","elevation":12,"daily_units":{"time":"iso8601","temperature_2
m_max":"°C","temperature_2m_min":"°C"},"daily":{"time":["2025-03-23","2025-
03-24","2025-03-25","2025-03-26","2025-03-27","2025-03-28","2025-03-
29"],"temperature_2m_max":[28.7,29.6,30.3,31.7,30.9,23.7,17.2],"temperatu
re_2m_min":[11.1,14.1,15.8,18.4,21.2,16.8,13.4]}}
```

JSON

- `response = requests.get(url, params=params)`
- `data = response.json()`

 dict in Python

```
{
    "daily": {
        "time": ["2025-03-23", "2025-03-24", "2025-03-25", "2025-03-26", "2025-03-27", "2025-03-28", "2025-03-29"],
        "temperature_2m_max": [28.7, 29.6, 30.3, 31.7, 30.9, 23.7, 17.2],
        "temperature_2m_min": [11.1, 14.1, 15.8, 18.4, 21.2, 16.8, 13.4]
    }
}
```



```
data["daily"]["time"][0]
```

Python程式

- `import requests`
- `import pandas as pd`
- `url = "https://api.open-meteo.com/v1/forecast"`
- `params = {`
 - `"latitude": 25.0330,`
 - `"longitude": 121.5654,`
 - `"daily": "temperature_2m_max,temperature_2m_min"``}`
- `response = requests.get(url, params=params)`
- `data = response.json()`

Python程式

- `d = pd.DataFrame({
 "日期": data["daily"]["time"],
 "最高溫度 (°C)": data["daily"]["temperature_2m_max"],
 "最低溫度 (°C)": data["daily"]["temperature_2m_min"]
})`
- `print("台北市一週天氣預報:")`
- `print(d)`

READ_HTML (1)

- `import pandas as pd`
- `url = "https://www.tiobe.com/tiobe-index/"`
- `d = pd.read_html(url, encoding="utf-8")`
- `print(d[0])`
- `# First table`

READ_HTML (2)

- `import pandas as pd`
- `url = "https://rate.bot.com.tw/xrt?Lang=zh-TW"`
- `data = pd.read_html(url, encoding="utf-8")`
- `d = data[0]`
- `print(d)`
- `d = d[[d.columns[0], d.columns[2]]]`
- `d.columns = ["幣別", "本行賣出"]`
- `d.set_index("幣別", inplace=True)`
- `print(d)`

幣別		現金匯率		Unnamed: 4_level_0	
幣別	Unnamed: 1_level_1	本行買入	本行賣出	本行買入	
0	美金 (USD) 美金 (USD)	27.27	27.94	27.595	27.745
1	港幣 (HKD) 港幣 (HKD)	3.394	3.598	3.515	3.585
2	英鎊 (GBP) 英鎊 (GBP)	36.69	38.81	37.585	38.215
3	澳幣 (AUD) 澳幣 (AUD)	19.74	20.52	19.955	20.3

READ_CSV

- `import pandas as pd`
- `url = "https://rate.bot.com.tw/xrt/flcsv/0/day"`
- `d = pd.read_csv(url)`
- `d.set_index("幣別", inplace=True)`
- `print(d)`

Problem

- 自己找一個網頁下載資料，利用Pandas整理資料，最後輸出結果
- 科學數據：
- 電影時刻表：
- 一週預報(XHR)：<https://www.cwa.gov.tw/V8/C/W/week.html>
- 空氣品質指標(AQI)：<https://data.gov.tw/dataset/40448>
- 統一發票：<https://invoice.etax.nat.gov.tw/index.html>
- 臺灣銀行牌告匯率網頁：<https://rate.bot.com.tw/xrt>