

運算邏輯與程式應用

鍾健雄 博士

7/10/2025

授課教師

- 鍾健雄
- 美國維吉尼亞大學系統工程博士 (**Ph.D of Systems and Information Engineering, University of Virginia, USA**)
- 研究興趣
 - 程式開發應用、機器學習、資料視覺化、資料庫分析實作、遊戲程式設計
 - 離散事件系統模擬、企業架構建模分析(UML)、工控系統資訊安全管理
- 手機 0919341293
- 電子郵件 jschungchit@gmail.com

課程目標

了解電腦
運作原理

明白電腦
語言運作

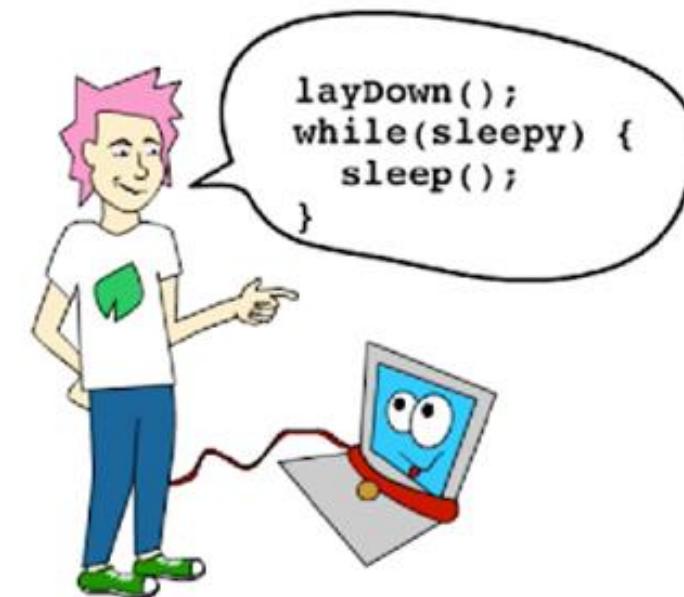
培養計算
思考概念

學習撰寫
程式設計

電腦「程式」(Program)是甚麼？

- 一組有次序的指令，電腦根據這些指令可以為我們執行特定工作。

- 電腦：執行指令的機器
- 次序：一步一步、有邏輯性
- 指令：電腦了解、溝通語言
- 工作：電腦執行指令的結果
- 我們：寫程式的人



學習如何與電腦溝通

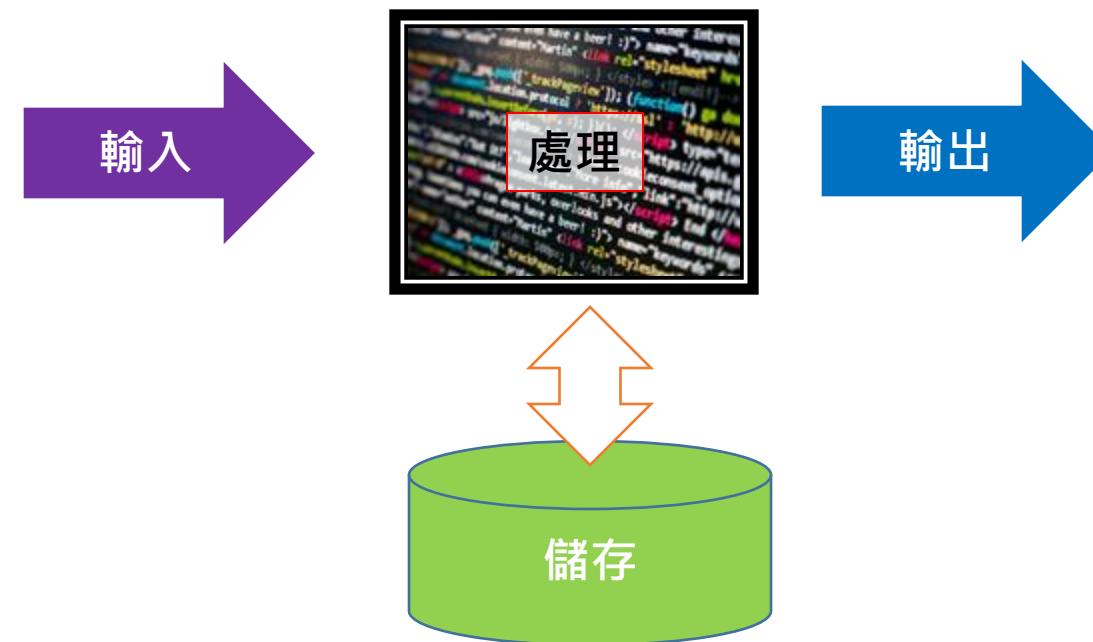
字彙
語法
規則



程式語言就是人類用來和電腦溝通的語言，是一行一行的指令，可以將人類的思考邏輯轉換成電腦能夠了解的語言，用來指揮電腦運算或工作的指令集合（程式碼）。

電腦是一個處理資料機器

- 輸入 (Input)
- 儲存 (Output)
- 處理 (Processing)
- 輸出 (Output)

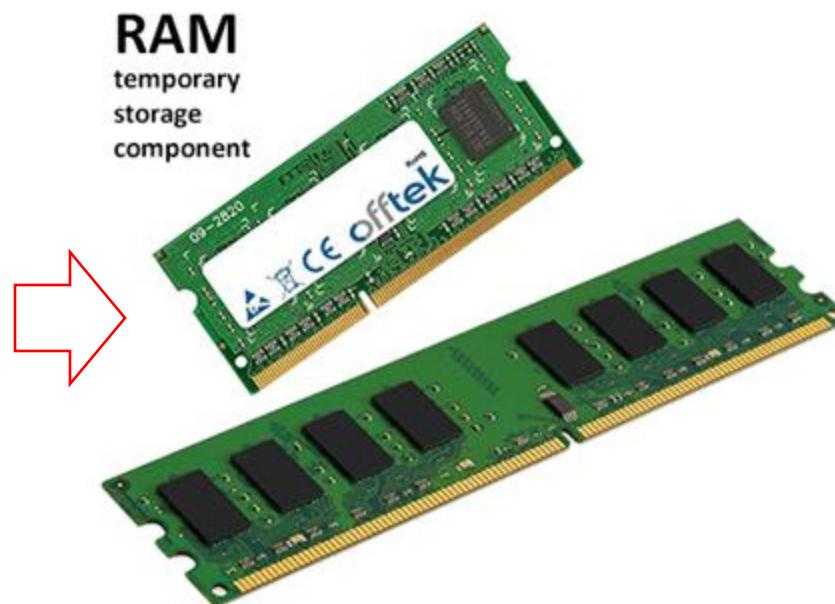


電腦基本組成

- **輸入**
 - 鍵盤、滑鼠、掃描器、麥克風、拇指碟、攝影機
- **處理**
 - 中央處理單元(CPU)、顯示卡、網路卡
- **儲存**
 - 隨機處理記憶體(RAM)、硬碟(hard disk)、光碟機(CD/DVD)
- **輸出**
 - 顯示器(螢幕)、印表機、繪圖機、喇叭



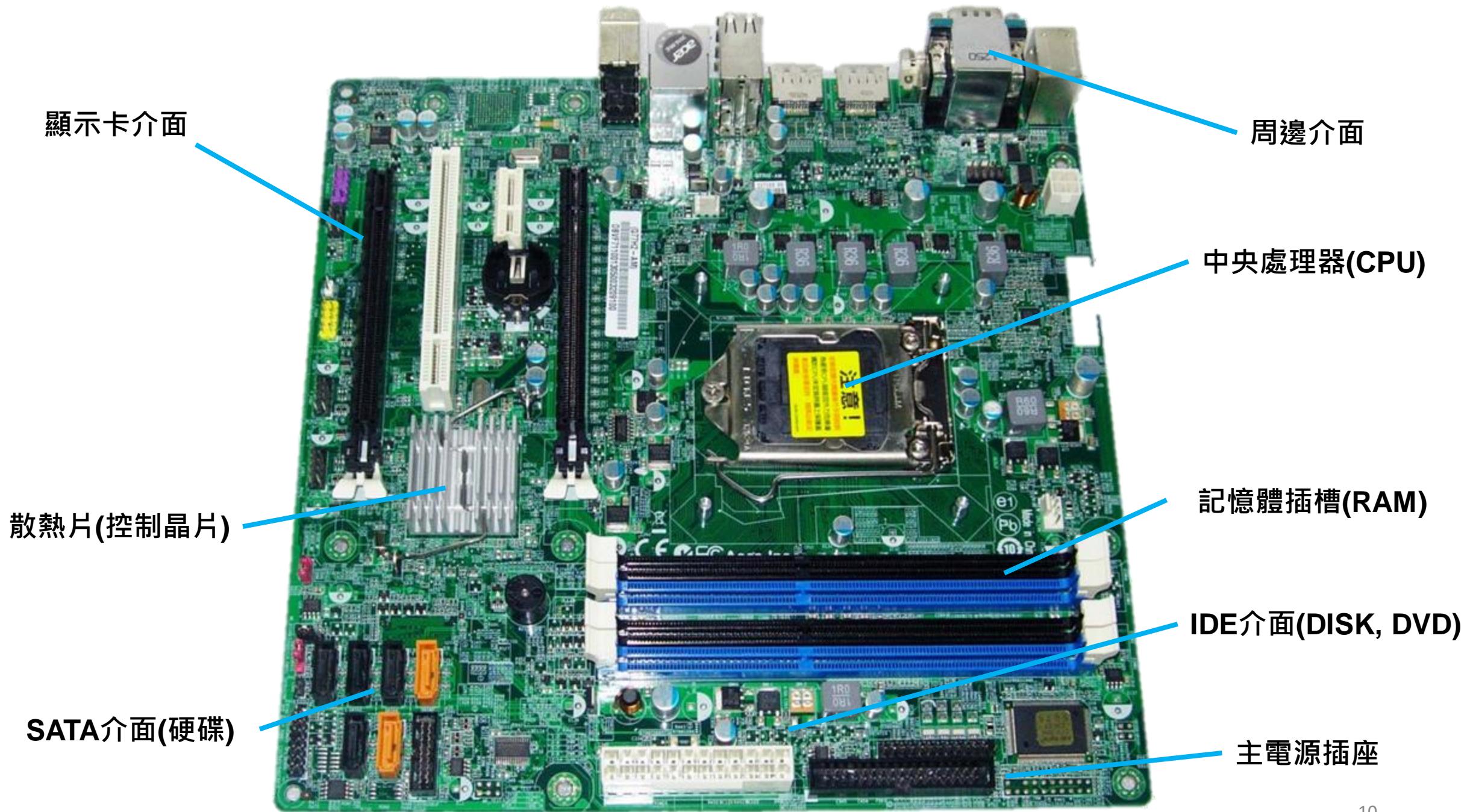
中央處理器
影像處理器



記憶體
硬碟



Hard Disk
Permanent storage component

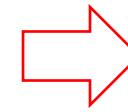


輸入裝置

BOTH

輸出裝置



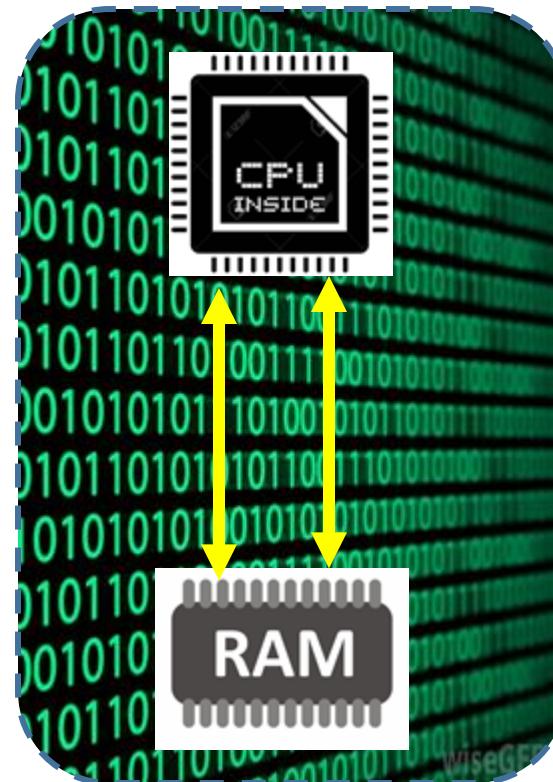


機器語言



程式開發

電腦處理



軟體應用



資料儲存

程式語言分類

機器語言

- 0與1的組合
- CPU直接讀懂
- 最低階、最無人性化、最難撰寫

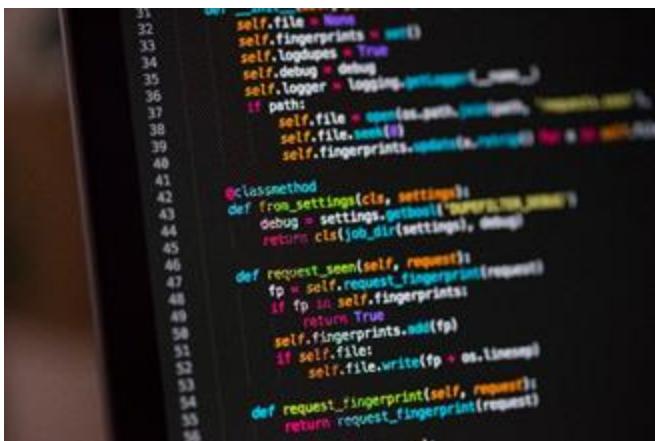
組合語言

- 接近口語的指令
- 低階語言

高階語言

- 接近人類語言語法
- 必須經過翻譯轉換為機器語言

程式語言翻譯



```
31     self.file = None
32     self.fingerprints = {}
33     self.logdups = True
34     self.debug = debug
35     self.logger = logger
36     if path:
37         self.file = open(path, 'w')
38         self.file.write('')
39         self.fingerprints = {}
40
41     @classmethod
42     def from_settings(cls, settings):
43         debug = settings.getboolean('DEBUG', False)
44         return cls(job_dir(settings), debug)
45
46     def request_seem(self, request):
47         fp = self.request_fingerprint(request)
48         if fp in self.fingerprints:
49             return True
50         self.fingerprints.add(fp)
51         if self.file:
52             self.file.write(fp + os.linesep)
53
54     def request_fingerprint(self, request):
55         return request_fingerprint(request)
```

高階語言
人類熟悉的溝通模式



翻譯過程



```
01101001111010011010011110100
01010001051010101000001010110
11011011100100110110111001001
01000101100010010001011000010
0101000101101101010001011011
0110100100011001011101000110
01101001111010011010011110100
1010001011011010100010110110
101101011100100110110111001001
01000101100010010001011000010
0101000101101101010001011011
01101110100011011011101000110
```

機器語言
電腦熟悉的模式

翻譯器

編譯器 (compiler)

- 編譯是先將程式全部翻譯成機器碼後，電腦再一次執行這些氣器碼，以後再執行程式，只要執行機器碼，不需要再重新編譯。

解譯器 (interpreter)

- 直譯則是每翻譯完一行程式，電腦就執行一行機器碼，接著再繼續翻譯下一行，執行一行機器碼，直到結束為止。每次電腦重新執行程式都要再經過直譯的過程。

Python 程式語言

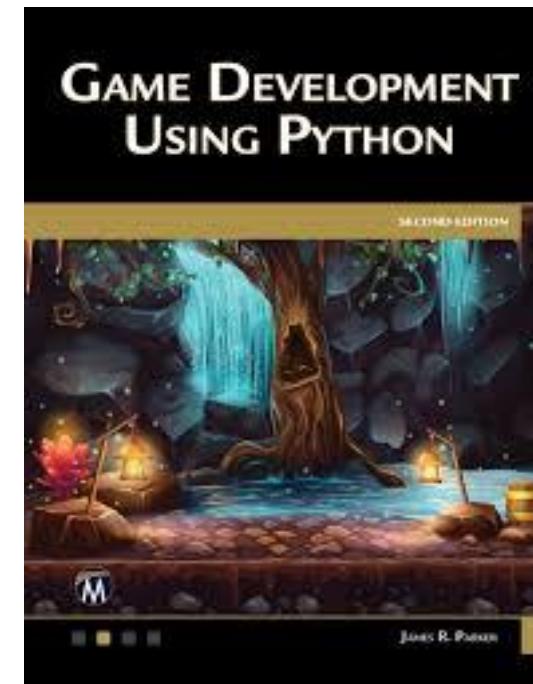
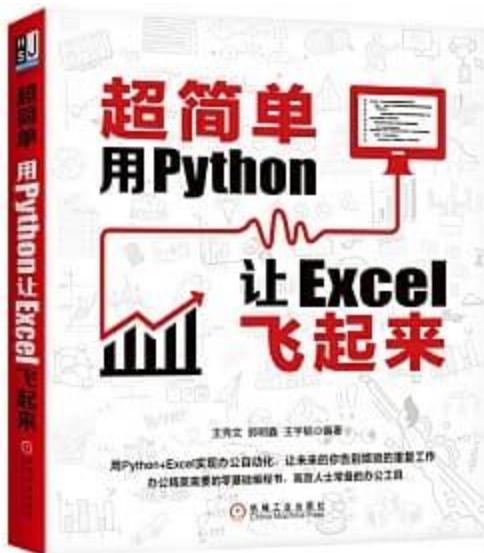
- 一種高階程式語言
- Python 語法易學易讀
- 人人都能使用的程式語言
- 入門程式語言
- 美國高中生必學的程式語言
- 自由/開放原始碼
- 直譯式語言



<https://cs50.harvard.edu/x/weeks/6/>

Python 應用領域

- 資料探勘 (Data Mining)
- 資料科學工作
- 網頁設計
- App 設計
- 遊戲設計
- 自動控制
- 生物科技
- 大數據



Python程式語言功能

變數與資料型
別

輸入與輸出

運算式

條件控制結構

迴圈重複執行

資料結構
(字串陣列)

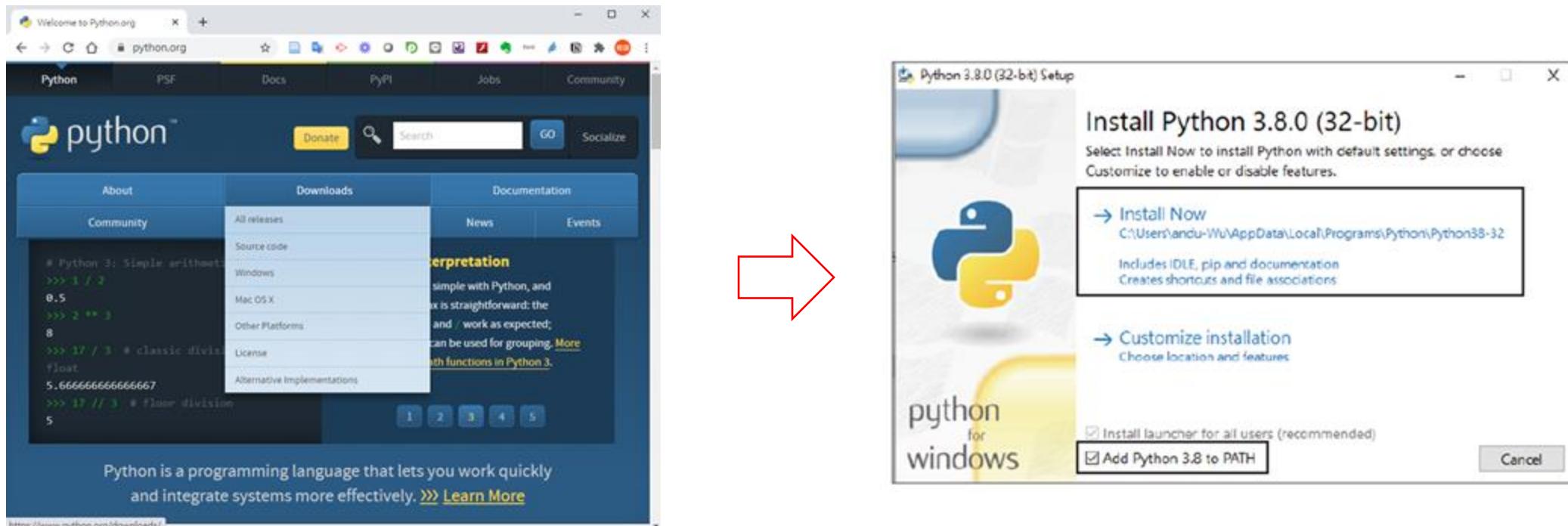
自訂函式

模組與套件

檔案處理

Python安裝 (python.org)

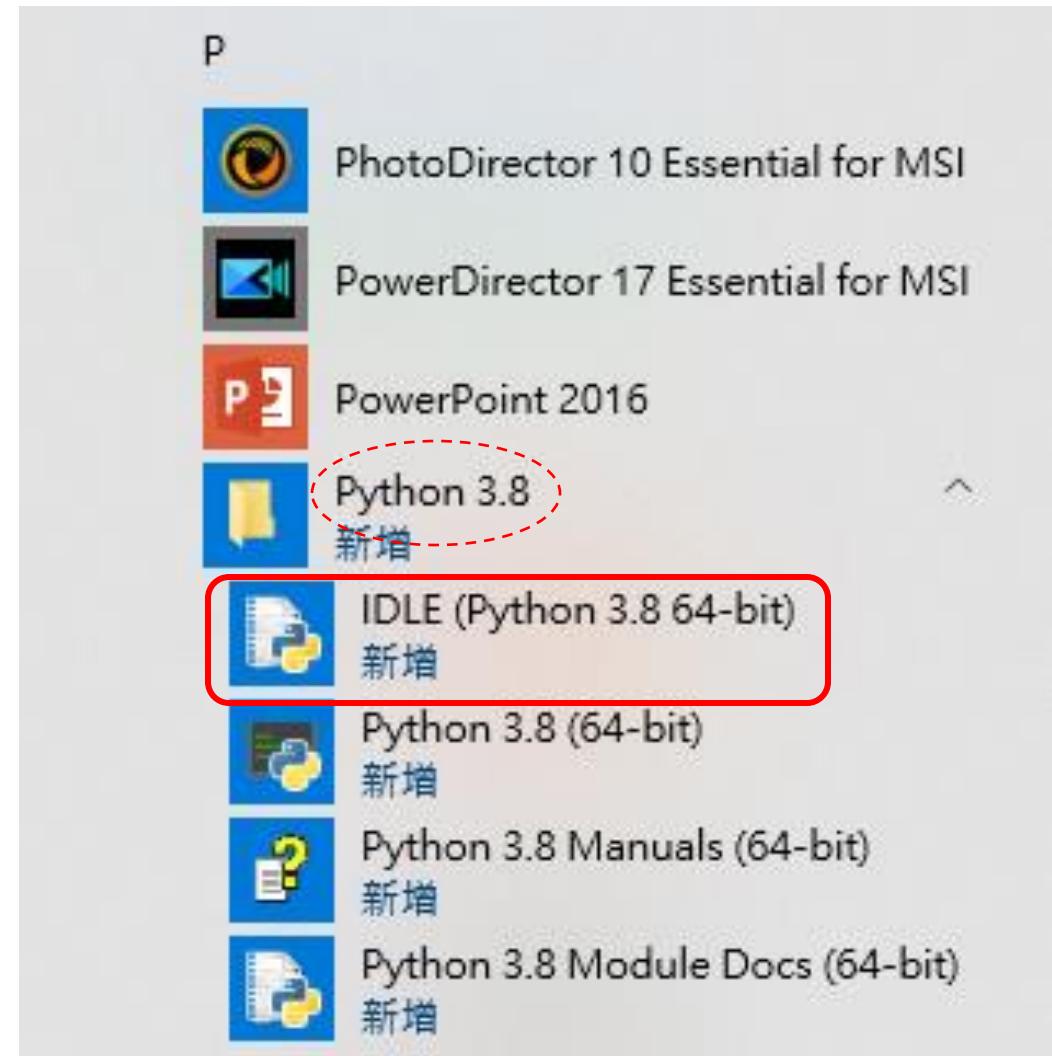
- Python 是一種跨平台的程式語言（Windows、Linux、Mac OS）都可以安裝與使用



執行 Python

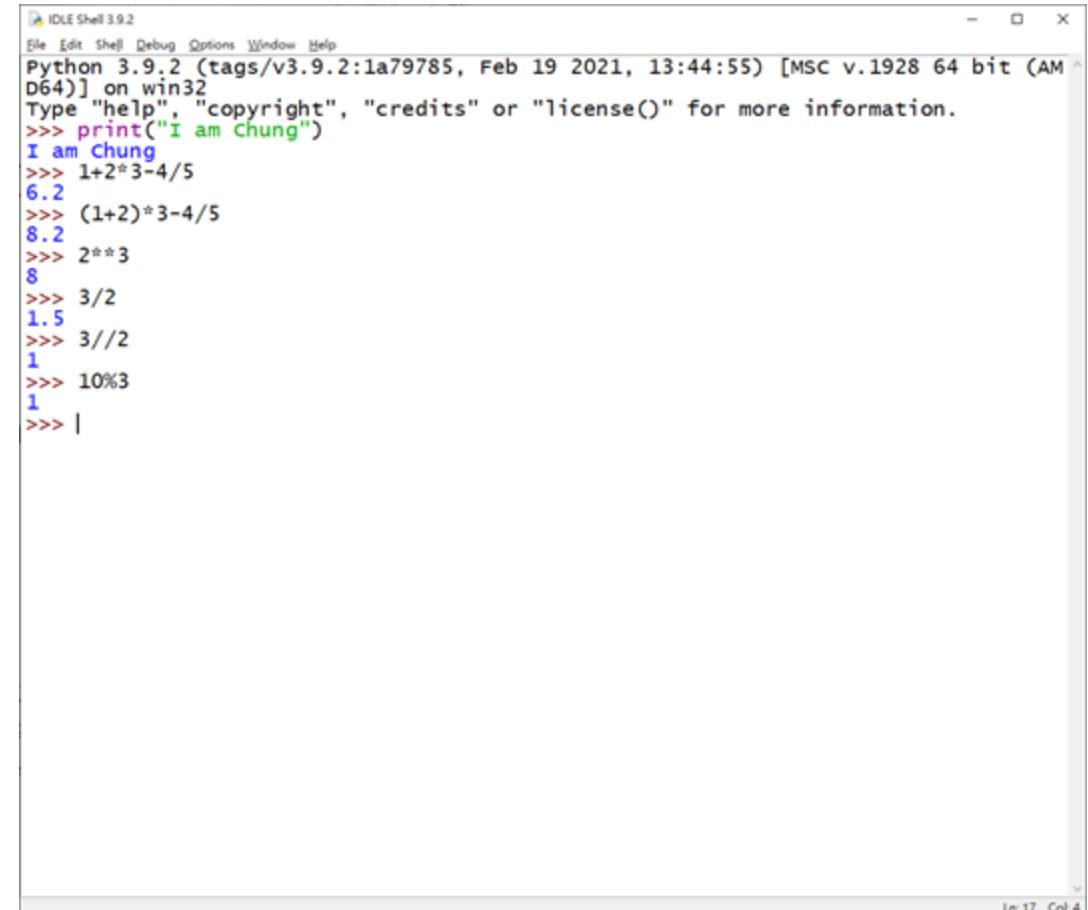
- **IDLE**

- 內建的 Python 整合式開發環境軟體(簡稱 IDE)
- 進入 Python 互動交談模式 (Interactive Mode)



Python 互動交談模式

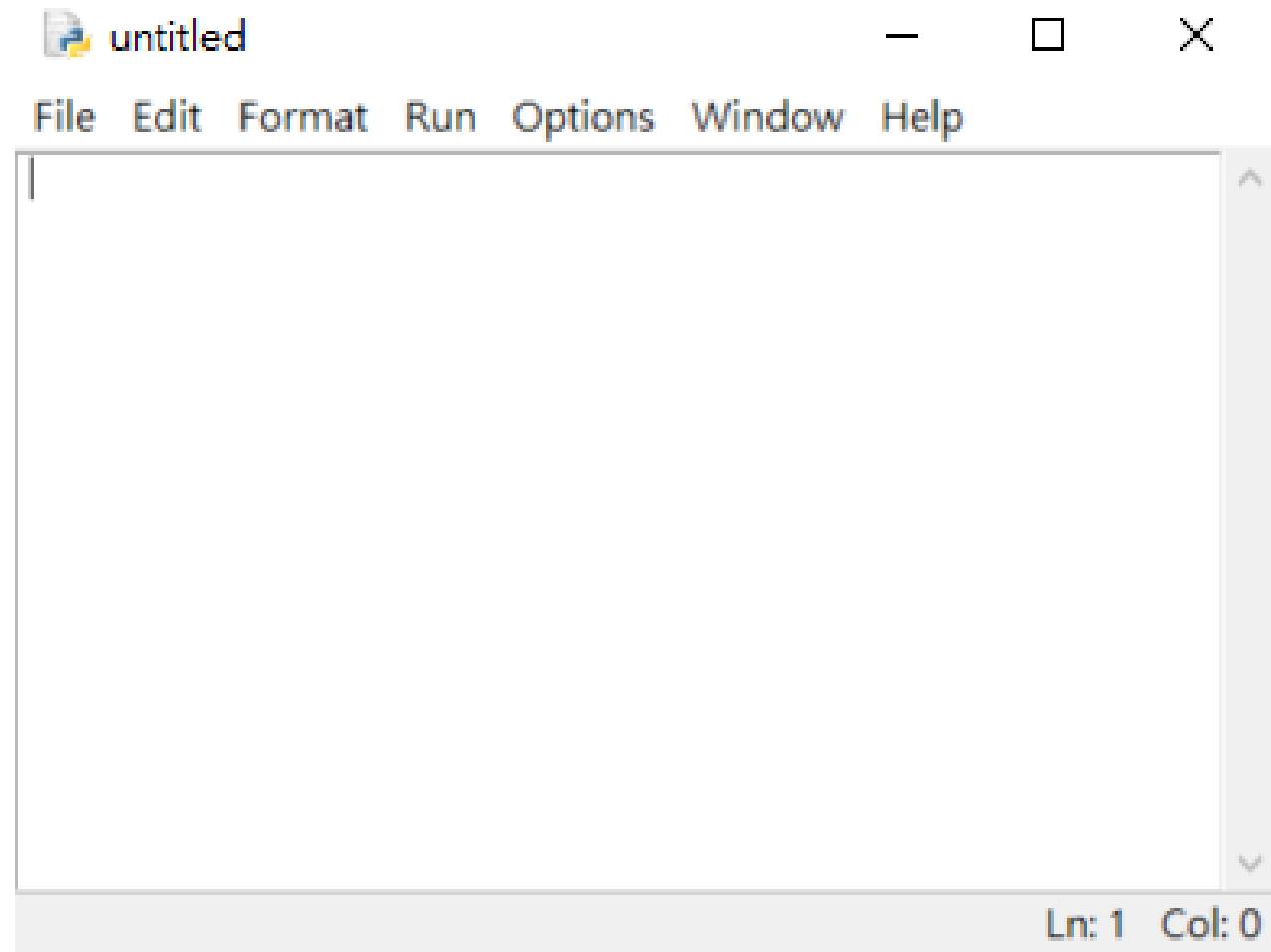
- 利用python程式與電腦溝通的介面
- 在 >>> 輸入python指令，按下 enter ，
python的「直譯器」讀取指令，執行指定
- 試試將python互動介面當作是計算機來使用
 - $1+2*3-2/5$
 - $(1+2)*3-2/5$
 - $2^{**}3$ (2的3次方)
 - $3//2$ (3÷2的結果取整數)
 - $10\%3$ (10÷3的餘數)



```
IDLE Shell 3.9.2
File Edit Shell Debug Options Window Help
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("I am Chung")
I am Chung
>>> 1+2*3-4/5
6.2
>>> (1+2)*3-4/5
8.2
>>> 2**3
8
>>> 3/2
1.5
>>> 3//2
1
>>> 10%3
1
>>> |
```

Python程式編輯模式

- **File > New File**
- 編寫程式
- 按 Run 執行程式



小烏龜繪圖套件

- **import turtle**
 - **from turtle import ***
- **win = turtle.Screen()** #建立視窗
- **tt = turtle.Turtle()** #建立一個烏龜物件
- **tt.shape("turte")** #烏龜形狀 circle, square, classic
- **tt.hideturtle()**
- **tt.showturtle()**
- **win.bgcolor("yellow")**

小烏龜繪圖(命令列模式)

```
>>> tt.forward(100) 向前走100步  
>>> tt.left(45) 左轉45度  
>>> tt.forward(100) 向前走100步  
>>> tt.left(90) 左轉90度  
>>> tt.forward(100) 向前走100步  
>>> tt.pencolor("red") 改變畫筆紅色  
>>> tt.left(70) 左轉70度  
>>> tt.forward(100) 向前走100步  
>>> tt.right(70) 右轉70度  
>>> tt.pencolor("blue") 改變畫筆藍色  
>>> tt.forward(100) 向前走100步  
>>> tt.right(60) 右轉60度  
>>> tt.pencolor("green") 改變畫筆綠色  
>>> tt.forward(100) 向前走100步  
>>> tt.left(120) 左轉120度  
>>> tt.forward(200) 向前走100步  
>>> tt.right(75) 右轉75度  
>>> tt.pencolor("purple") 改變畫筆紫色  
>>> tt.forward(100) 向前走100步  
>>> tt.pencolor("black") 改變畫筆黑色  
>>> tt.home() 回到原點
```

小烏龜繪圖(程式模式)

#導入 turtle 套件

```
import turtle
```

#建立畫布

```
win = turtle.Screen()
```

#建立小烏龜

```
tt = turtle.Turtle()
```

#清除畫布

```
win.clear()
```

#重置繪圖狀態

```
tt.reset
```

```
tt.forward(100)
```

```
tt.left(45)
```

```
tt.forward(100)
```

```
tt.left(90)
```

```
tt.forward(100)
```

```
tt.pencolor("red")
```

```
tt.left(70)
```

```
tt.forward(100)
```

```
tt.right(70)
```

```
tt.pencolor("blue")
```

```
tt.forward(100)
```

```
tt.right(60)
```

```
tt.pencolor("green")
```

```
tt.forward(100)
```

```
tt.left(120)
```

```
tt.forward(200)
```

```
tt.right(75)
```

```
tt.pencolor("purple")
```

```
tt.forward(100)
```

```
tt.pencolor("black")
```

```
tt.home()
```

Turtle Graphic

圓形 (circle)

`tt.circle(100)`

三角形(trangle)

`tt.forward(100)`

`tt.right(120)`

`tt.forward(100)`

`tt.right(120)`

`tt.forward(100)`

四方型(rectangle)

`tt.forward(100)`

`tt.right(90)`

`tt.forward(100)`

`tt.right(90)`

`tt.forward(100)`

`tt.right(90)`

`tt.forward(100)`

`tt.right(90)`

小烏龜運動指令

- **forward(n)**
 - 向前走 n 步
- **back(n)**
 - 向後走 n 步
- **goto(x, y)**
 - 移動至 (x,y) 座標
- **setx(x)**
 - 移動至x座標(x)
- **sety(y)**
 - 移動至y座標(y)
- **setposition**
 - 設定座標位置
- **right(d)**
 - 右轉 d 度
- **left(d)**
 - 左轉 d 度
- **setheading(a)**
 - 設定方向 a
- **home()**
 - 移動至原點(0,0)

小烏龜畫筆指令

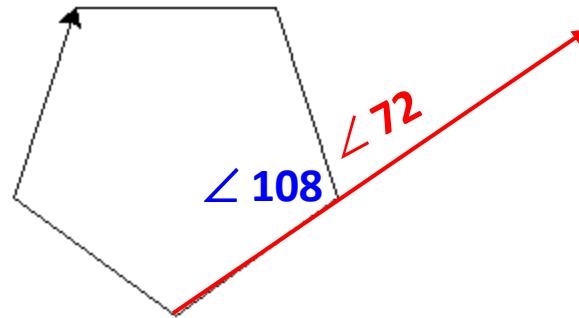
- **penup()**
 - 提筆(不留痕跡)
- **pendown()**
 - 下筆(留校痕跡)
- **pencolor("red")**
 - 畫筆顏色("red", "green", "blue"..)
- **pensize(3)**
 - 畫筆寬度
- **color(pencolor, fillcolor)**
- **fillcolor (塗色顏色)**
 - **begin_fill()**: 開始記錄圖形頂點
 - **end_fill()** : 結束紀錄圖形頂點
- **colormode(1.0 / 255)**
 - 1.0 : (1.0, 0.0, 0.0)
 - 255 : (155, 255, 125) RGB值



五邊形

五邊形

```
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)  
tt.forward(100)
```



$$N \text{ 邊形內角總和} = 180 * (N-2)$$

$$5 \text{ 邊形內角總和} = 180 * (5-2) = 540$$

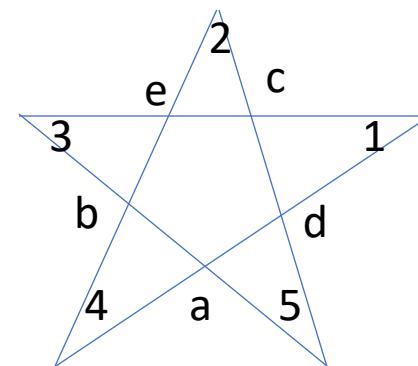
$$\text{每個內容} = 540/5 = 108$$

$$\text{每個頂點轉} 180-108 = 72$$

五角星

星形(star)

```
tt.forward(100)  
tt.right(144)  
tt.forward(100)  
tt.right(144)  
tt.forward(100)  
tt.right(144)  
tt.forward(100)  
tt.right(144)  
tt.forward(100)  
tt.right(144)
```



N角星内角和 = $180 * (N-2 * 2)$
N角星内角和/N
 $180 - (\text{N角星内角和}/N)$

$$1+3+a=180$$

$$2+5+b=180$$

$$3+5+c=180$$

$$2+4+d=180$$

$$1+4+e=180$$

$$a+b+c+d+e=540$$

$$2(1+2+3+4+5)+540=900$$

$$1+2+3+4+5 = (900-540)/2=180$$

$$180/5=36$$

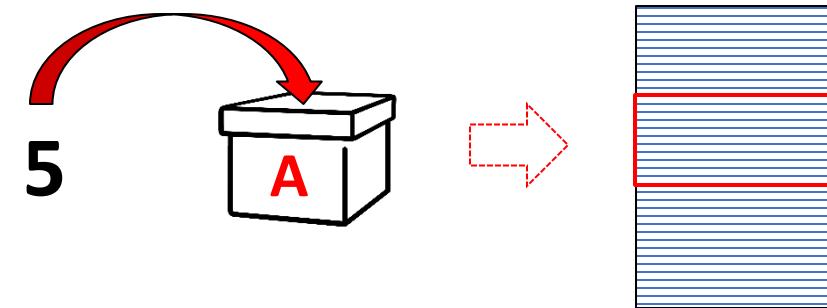
$$180-36=144$$

學習程式的秘訣

嘗試錯誤、勞者多能

變數

- 程式設計時，變數可代表某個資料
 - `messag = "Jeremy Lin is a good NBA player"` (字串)
 - `n = 20` (整數)
 - `pi = 3.1415926535897932` (浮點數)
- 變數如同黑盒子，內含資料，佔據一塊記憶體
 - `A = 5`
 - A表示變數，5表示資料
 - 「=」為「指定」運算子，表示「資料指定給變數」
- 變數可以組成運算式
- Python的變數是一種「動態型別」（Dynamically Typed），使用變數前不需要事先宣告型別
 - C/C++, Java 程式語言使用變數前必須先宣告型別



變數命名規則

1. 第一個字母必須是英文字母或底線(_abc)
2. 其他字元可以搭配英文大小寫字母、數字
3. 變數名稱區分大小寫
4. 變數長度不限
5. 變數名稱不可含有特殊符號
6. 關鍵字不能作為變數名稱

Python keywords

False	break	for	not
None	class	from	or
True	continue	global	pass
__peg_parser__	def	if	raise
and	del	import	return
as	elif	in	try
assert	else	is	while
async	except	lambda	with
await	finally	nonlocal	yield

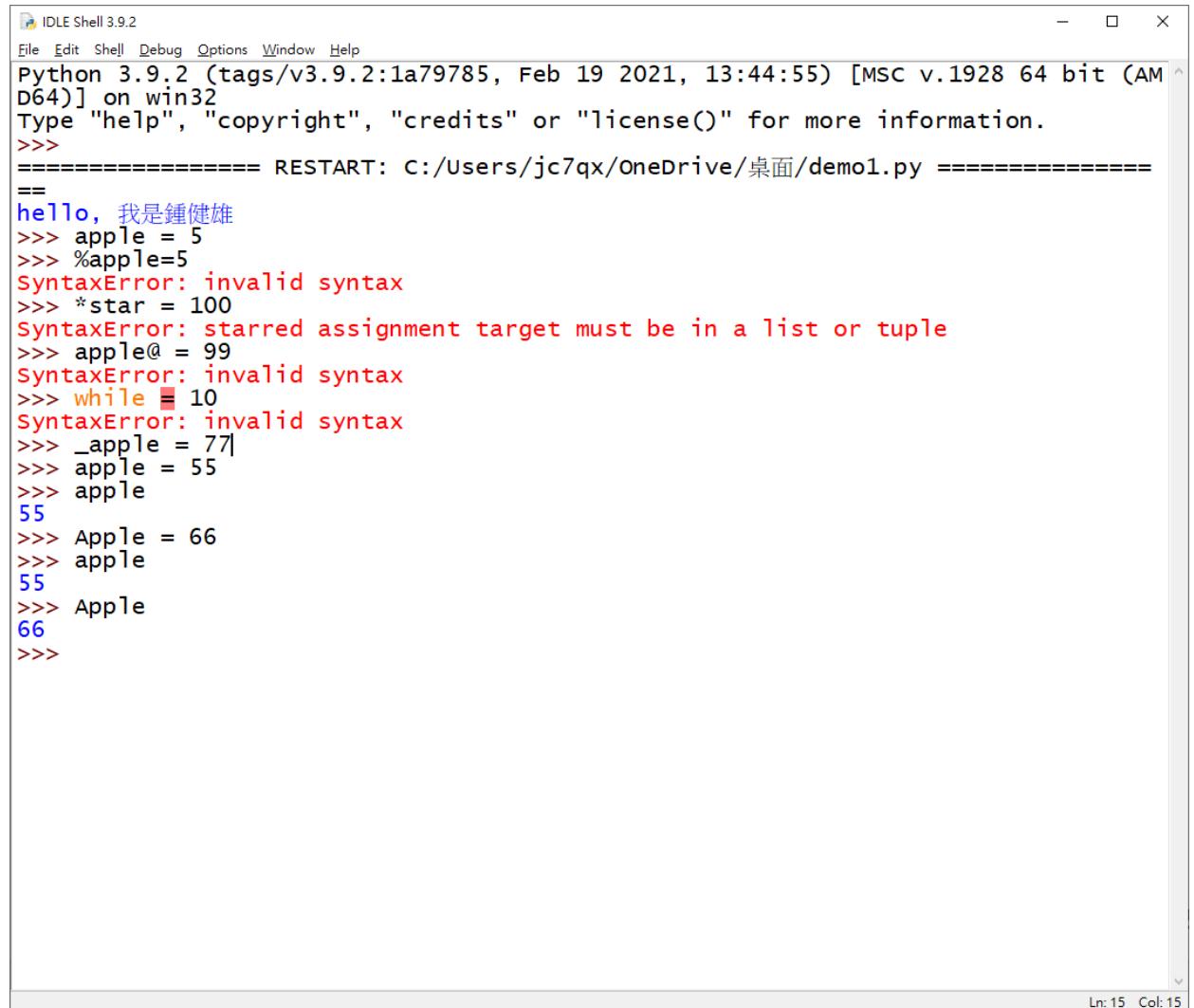
變數名稱範例

合法名稱	不合法名稱
Apple	%abc (rule 1)
_apple	5apple (rule 1)
app_le	apple@ (rule 5)
apple	*star (rule 5)
	while (rule 4)
	Try (rue 4)

上機實做

以下敘述何者正確？何者錯誤？
錯誤訊息

```
>>> Apple = 5
>>> apple = 5
>>> *apple = 5
>>> App_le = 5
>>> _apple = 5
>>> 123app = 5
>>> app@le = 5
```



The screenshot shows the Python 3.9.2 shell interface in IDLE. The user has run a script named 'demo1.py' located at C:/Users/jc7qx/OneDrive/桌面/demo1.py. The script contains several invalid assignments:

```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=====
RESTART: C:/Users/jc7qx/OneDrive/桌面/demo1.py =====
==

Hello, 我是鍾健雄
>>> apple = 5
>>> %apple=5
SyntaxError: invalid syntax
>>> *star = 100
SyntaxError: starred assignment target must be in a list or tuple
>>> apple@ = 99
SyntaxError: invalid syntax
>>> while ■ = 10
SyntaxError: invalid syntax
>>> _apple = 77|
>>> apple = 55
>>> apple
55
>>> Apple = 66
>>> apple
55
>>> Apple
66
>>>
```

運算式 電腦最擅長做的事 – 數學計算

運算元及運算子

- 運算式 = 運算元 + 運算子
- 運算符號，稱為「運算子(operator)」
 - 數值運算子包括： $+$, $-$, $*$, $/$, $\%$ (取餘數), $**$ (乘方), ...
- 運算的資料，稱為「運算元(operand)」
 - 運算元可以是常數、變數、函式
 - $3+5$ (+ 代表運算子, 3, 5 代表運算元)
 - $a=5; a**3$ ($**$ 代表乘方運算子, a, 3 代表運算元)
 - $math.sqrt(x**2+3.32**2)$

運算結果受到運算子及資料型別影響

- $5/3 = 1.66666666667$ (結果 float)
- $5//3 = 1$ (結果 int) · 但 $5//3.0 = 1.0$ (結果 float)
- $5**2 = 25$ (結果 int) · 但 $5**2.0 = 25.0$ (結果 float)
- $5%3 = 2$ (結果 int) · 但 $5%3.0 = 2.0$ (結果 float)

算術運算子

數學運算	運算子	範例
加	+	$1+1=2$
減	-	$3-2=1$
乘	*	$5*3=15$
除	/	$5/3=1.666666666666667$
除法去除小數點	//	$5//3 = 1$ $5//3.0 = 1.0$
餘數	%	$5\%3 = 2$ $5.0\%3 = 2.0$
乘方	**	$5**2 = 25$ $5**2.0 = 25.0$ $2**0.5 = 1.4142135623730951$ (開根號) $2**(-1) = 0.5$ (倒數)

實做練習

```
>>> a = 3  
>>> b = 5  
>>> c = 6.7  
>>> d = a+b  
>>> d = b+c  
>>> 3/5  
>>> 3.0/5  
>>> 3/float(5)  
>>> c/b  
>>> c//b  
>>> a+b*c  
>>> (a+b)*c  
>>> 5%3 (5除3的餘數)  
>>> a**b (a的b次方)  
>>> a**0.5 (開根號)  
>>> a**(-1) (倒數)  
>>> d = b**2-4*a*c  
>>> d**(0.5)
```

運算式的表示

- 半徑為 100 公分的圓面積與週長(r代表半徑)

- 圓面積 = πr^2 (π乘r的平方)
- 圓周長 = $2\pi r$ (2乘π乘r)
- r : 半徑

- Python計算式

- `r = 100; pi = 3.14159`
- `A = pi*r**2`
- `P = 2*pi*r`

運算式

- 身高體重指數 (BMI)
 - w = 體重 · 單位: 公斤
 - h = 身高 · 單位: 公尺
 - $BMI = \frac{w}{h^2}$
- Python 計算式
 - $BMI = w / (h^{**2})$

運算式

- 求解一元二次方程式

$$ax^2 + bx + c = 0$$

$$x = \frac{-b \mp \sqrt{b^2 - 4ac}}{2a}$$

- 判別式

- $b^2 - 4ac < 0$
 - 無解
- $b^2 - 4ac = 0$
 - 有唯一解, $x = -b/(2*a)$
- $b^2 - 4ac > 0$
 - 有二解 x_1, x_2 , $d = (b**-4*a*c)**(1/2)$
 - $x_1 = (-b+d)/(2*a)$
 - $x_2 = (-b-d)/(2*a)$

課後實做練習

- 請利用上頁公式，使用python程式解以下一元二次方程式

$$1) \quad x^2 + 5x + 6 = 0$$

$$2) \quad 4x^2 - 4x + 1 = 0$$

$$3) \quad x^2 + 2x + 6 = 0$$

專案#1：一元二次方程式求解

- $ax^2+bx+c=0$ 為一元二次方程式，求解過程如下
- $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$, 其中 $d = b^2 - 4ac$ 稱為判斷式，如果
 - $d > 0$, 則 x 有二解, $(\frac{-b + \sqrt{b^2 - 4ac}}{2a}, \frac{-b - \sqrt{b^2 - 4ac}}{2a})$
 - $d = 0$, 則 x 有一解, $-b/2a$
 - $d < 0$, 則無解
- 請轉寫一個 python 程式，由鍵盤讀取 a, b, c 三個變數值，計算一元二次方程式($ax^2+bx+c=0$)的解。

比較運算子

運算子	意義	範例	結果
<code>==</code>	相等	<code>5==5</code>	True
<code>!=</code>	不相等	<code>8!=5</code>	True
<code>></code>	大於	<code>3>8</code>	False
<code><</code>	小於	<code>5<8</code>	True
<code>>=</code>	大於或等於	<code>5>=10</code>	False
<code><=</code>	小於或等於	<code>5<=5</code>	True

邏輯運算子

- 邏輯運算子 **not**、**and**、**or**

A	B	A or B	A and B	Not A	Not (A and B)
True	True	True	True	False	False
True	False	True	False	False	True
False	True	True	False	True	True
False	False	False	False	True	True

課堂練習

1. `>>> 5 == 5` True
2. `>>> 8 != 5` True
3. `>>> 3 > 8` False
4. `>>> 5 < 8` True
5. `>>> 5 >= 10` False
6. `>>> (5 == 5) and (5<5)` True and False
7. `>>> (5 == 5) and (5>3)` True and True
8. `>>> (5 < 3 or 5 == 5)` False or True
9. `>>> not(5 < 3 or 5 == 5)` not (False or True)

Python運算子優先等級

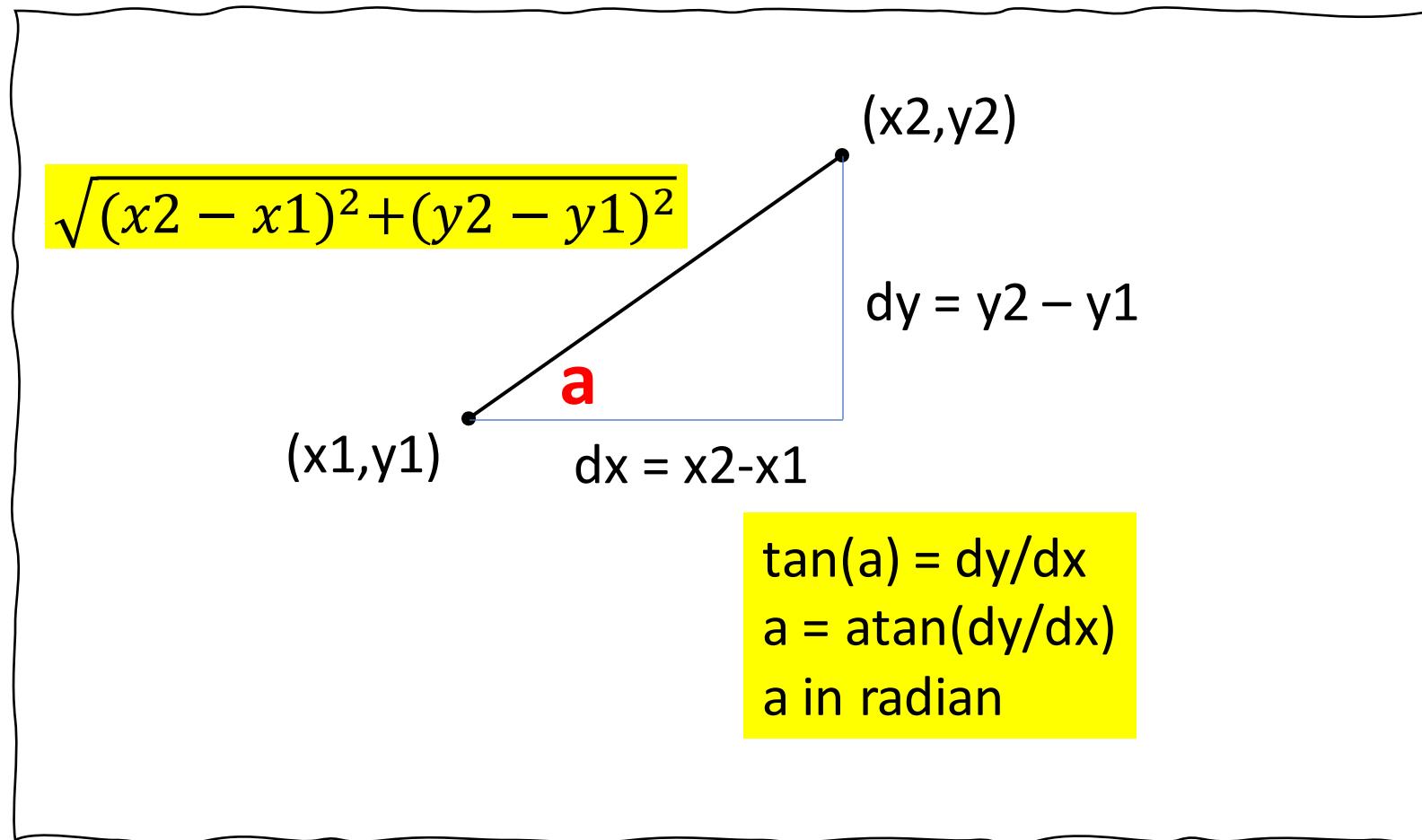
Operator	Description
<code>()</code>	括號
<code>**</code>	求乘方
<code>+x, -x, ~x</code>	正、負、非位元
<code>*, /, //, %</code>	乘、除、求餘數
<code>+, -</code>	加、減
<code>==, !=, >, >=, <, <=,</code> <code>is, is not, in, not in</code>	比較、確認、群組屬於運算子
<code>not</code>	邏輯 NOT
<code>and</code>	邏輯 AND
<code>or</code>	邏輯 OR

已知 $a=3, b=5, c=7, d=9$ 計算以下結果

1. $a+b-c*d/a$
2. $a + b - (c * d) / a$
3. $a + (b - c) * d) / a)$
4. $(a + b) - (c * d / a)$
5. $(a + (b - c) * (d / a))$

Operator	Description
<code>()</code>	括號
<code>**</code>	求乘方
<code>+x, -x, ~x</code>	正、負、非位元
<code>*, /, //, %</code>	乘、除、求餘數
<code>+, -</code>	加、減
<code>==, !=, >, >=, <, <=,</code> <code>is, is not, in, not in</code>	比較、確認、群組屬於運算子
<code>not</code>	邏輯 NOT
<code>and</code>	邏輯 AND
<code>or</code>	邏輯 OR

運算式應用: 計算兩點間距離及方向



小烏龜的運動

- 從目前位置 (x_0, y_0) 移動至下一點位置 (x_1, y_1)
- 將小烏龜方向轉向 (x_1, y_1)
 - $dx = x_1 - x_0, dy = y_1 - y_0$
 - $a = \text{atan2}(dy/dx)$ 経度量
 - 將 a 轉為角度 $a = \text{math.degrees}(a)$
- 計算向前走的步數
 - $\text{stelps} = \text{math.sqrt}((x_2 - x_1)^{**2} + (y_2 - y_1)^{**2})$
- 小烏龜轉向的基準 – 朝向東為0

print函數

- **print(x)**函數將 x 以「字串」格式列印至螢幕(stdio標準輸出)
 - `print(123)`
 - `a = 5.67; print(a)`
- **print(x, y)** 列印出 x y
 - `print(123, a) → 123 5.67`
- 列印的格式可以自訂(TBA)

input()輸入函數

- **input** 函數可以接收標準輸入裝置(如鍵盤)輸入的資料，數值、字串、或字元，同時可以將結果指定給變數
- **a = input("提示字串")**
 - **input()**輸入的結果為「**字串**」(由鍵盤輸入)
 - 將結果指定給變數**a**
 - 如果輸入的結果期望是數值，則需要轉換
- **程式範例(請打開程式視窗)**

```
name = input("輸入姓名")
height = input("輸入身高")
print("我是 ", name, "身高 ", height, "公分")
```

課後練習: 輸入身高、體重，計算BMI值

- 輸入身高: (ht)

- `ht = input("身高(公分): ")`
- 轉換為浮點數(float) `ht = float(ht)`
- 將公分轉算為公尺: `ht = ht/100`

- 輸入體重: (wt)

- `wt = input("體重(公斤): ")`
- 轉換為浮點數(float) `wt = float(wt)`

- 計算 BMI

$$bmi = \frac{\text{體重}}{\text{身高}^2}$$

- 列印身高、體重、BMI值

- `print(ht, wt, bmi)`

程式結構

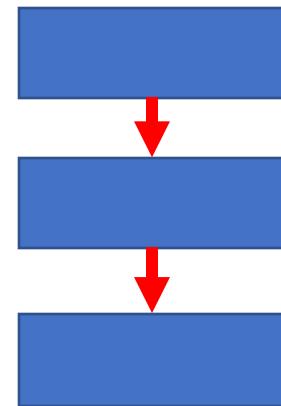
結構化程式

- 程式模組化

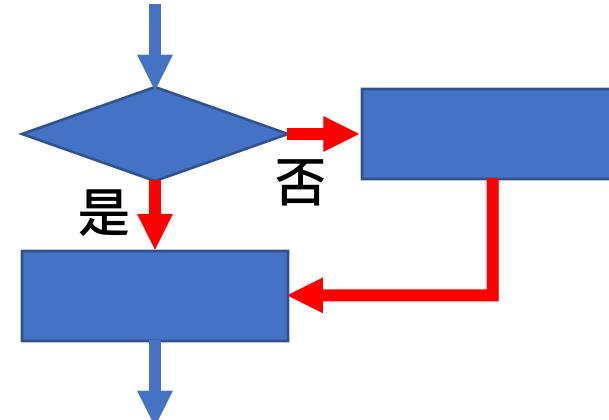
- 將問題分解成較小的範圍，使解題的步驟簡單清楚
- 每一個小問題的解題步驟設計成程式模組(Module)

- 流程結構化

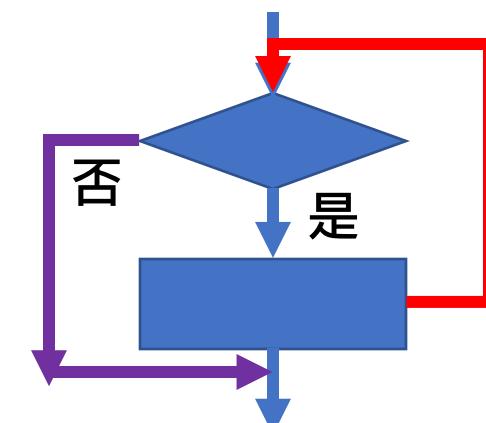
- 循序流程
- 選擇流程
- 重複流程



循序流程



選擇流程



重複流程

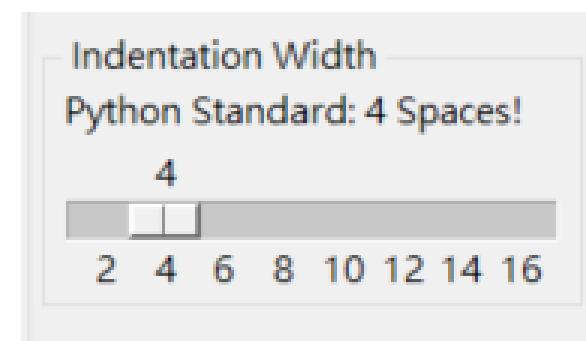
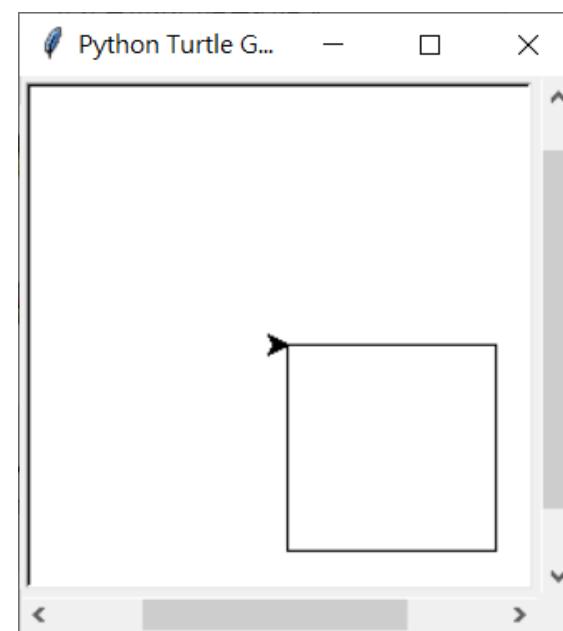
程式區塊與縮排

- Python的程式區塊是以「：」及「縮排」來定義
- 同一區塊的程式碼使用相同的空格數縮排

```
test2.py - G:/teaching/python... - X
File Edit Format Run Options Window Help
import turtle
turtle.Screen()
t = turtle
t.showturtle()

for i in range(4):
    t.forward(100)
    t.right(90)

Ln: 9 Col: 0
```



關係運算子 (Relational operator)

a = 5; b = 3						
描	述	運	算	式	布	林
大於			a > b			True
大於等於			a >= b			True
小於			a < b			False
小於等於			a <= b			False
等於			a == b			False
不等於			a != b			true

邏輯運算子 (Logical operator)

- 三種邏輯運算子：and (且) · or (或) · not (非)
 - $a \leq 90 \text{ and } a \geq 60$ (a 在 $[60, 90]$ 範圍內 true, 否則 false)
 - $a \% 2 == 0 \text{ or } a \% 3 == 0$ (如果 x 可以被 2 或被 3 整除就是 True , 否則是 False)
 - $\text{not } (a \% 2 == 0 \text{ or } a \% 3 == 0) \rightarrow a \% 2 != 0 \text{ and } a \% 3 != 0$

A	B	A and B	A or B	Not (A or B)
True	True	True	True	False
True	False	False	True	False
False	True	False	True	False
False	False	False	False	True

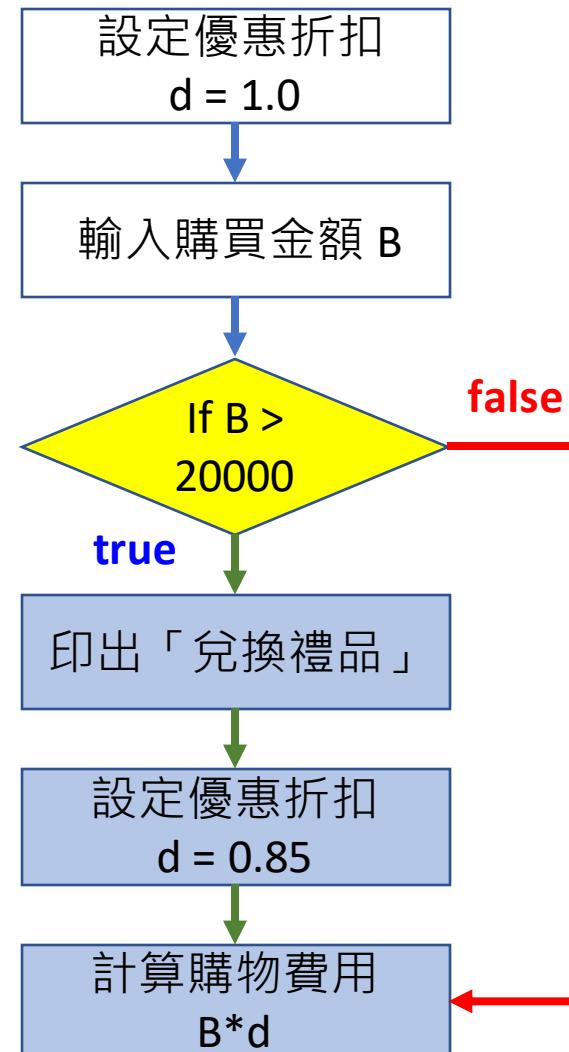
選擇性結構

- 簡單條件判斷式

if 條件判斷式：

□□程式區塊1

程式區塊2



購買商品，如果有優待卷，則商品打85折

```
test2.py - G:/teaching/python/test2.py (3....) ─ ┐ ×
File Edit Format Run Options Window Help
d = 1.0
B = float(input("購物金: "))
if B > 2000:
    print("請至7樓兌換禮品")
    d = 0.85
print("今天購物金額 = ", B*d)
Ln: 7 Col: 0
```

A screenshot of a Python code editor window titled "test2.py". The code defines a variable `d` as 1.0, reads a purchase amount `B` from input, and prints a message if `B` is greater than 2,000. If true, it sets `d` to 0.85 and prints the total shopping amount. The code editor shows the file path "G:/teaching/python/test2.py" and the current line and column numbers.

- 雙向條件判斷式

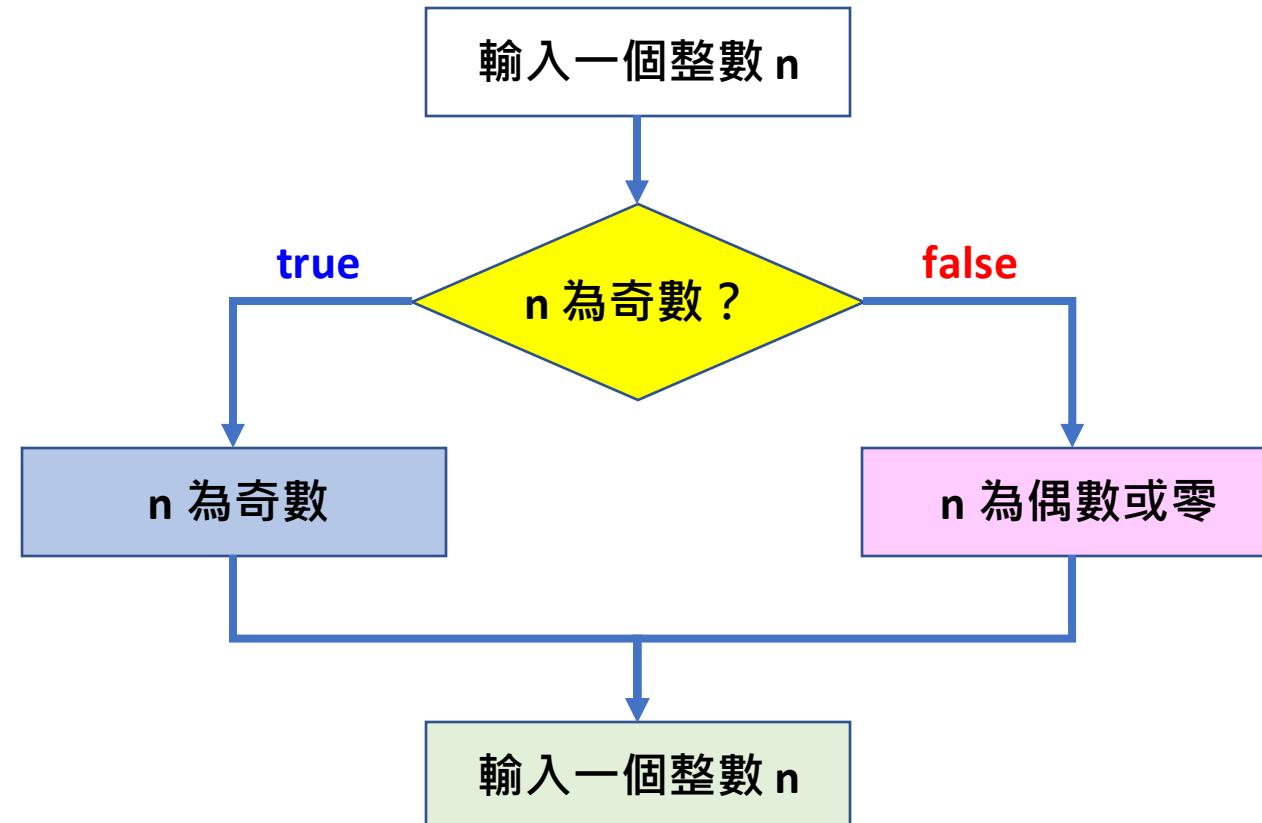
`if 條件判斷1 :`

`程式區塊1`

`else :`

`程式區塊2`

`程式區塊3`



程式碼

```
n = int(input("輸入一個整數: "))
```

```
if n%2==1:  
    print("%d 是奇數" % n)  
else:  
    print("%d 是偶數或零" % n)
```

- 巢狀條件判斷式

- if 條件判斷1 :

- if 條件判斷2 :

- 程式區塊1

- else:

- 程式區塊2

- else:

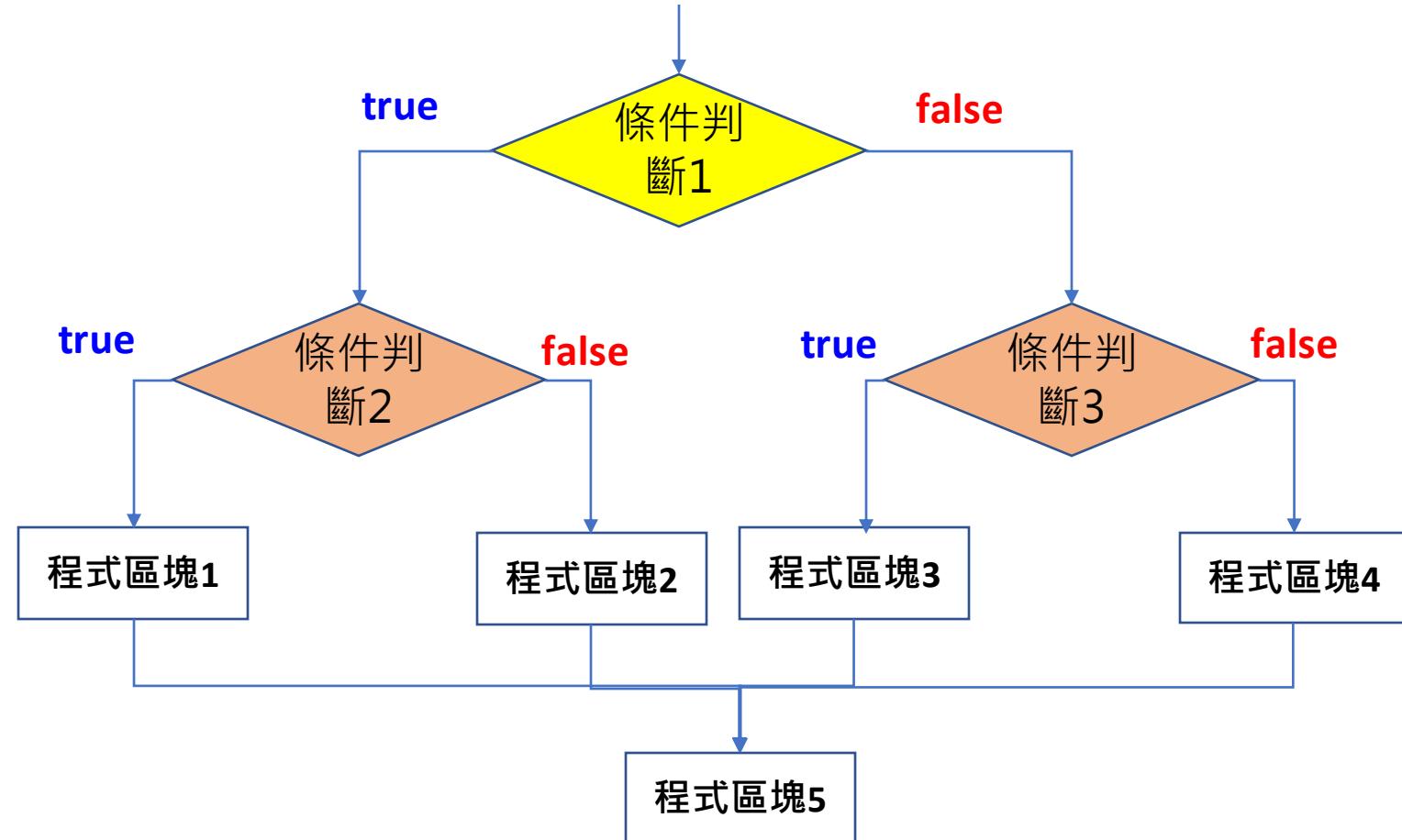
- if 條件判斷3 :

- 程式區塊3

- else:

- 程式區塊4

- 程式區塊5



判斷整數是奇數、偶數、零？

```
n = int(input("輸入一個整數: "))

if n%2==1:
    print("%d 是奇數" % n)
else:
    if n == 0:
        print("%d 是零" % n)
    else:
        print("%d 是偶數" % n)
```

要不要帶雨傘出門？

if 天正在下雨：
 帶雨傘出門

else:

 if (陰天) and (濕度 > 75)
 帶雨傘出門

else:

 不帶雨傘

- 已知 $a = \text{天是否下雨(true/false)}$ · $b = \text{陰天/晴天}$ · $c = \text{濕度(float)}$ ，根據左邊判斷式，以下狀況是否帶雨傘？
 - 晴天、濕度 = 50
 - 雨天、濕度 = 70
 - 陰天、濕度 = 70
 - 陰天、濕度 = 90

讓我們來寫個程式吧！

```
w = int(input("晴天：1 · 雨天：2 · 陰天：3 = "))

h = float(input("濕度："))

if (w==2):
    print("一定要帶雨傘出門喔\n")
else:
    if (w==1)
        print("輕輕鬆鬆出門，不要帶雨傘\n")
    else:
        if (w==3 and h >= 75)
            print("最好要帶雨傘出門喔\n")
        else:
            print("帶不帶雨傘隨便你\n")
```

- If ... elif ... else 結構

if 條件判斷 1 :

 程式區塊1

elif 條件判斷2 :

 程式區塊2

.....

elif 條件判斷 n :

 程式區塊n

else :

 程式區塊n

 程式區塊n+1

成績評分

- 90分以上 : A (`score >=90`)
- 75~90分 : B (`score >=75 and score < 90`)
- 60~74分 : C (`score >=60 and score <= 74`)
- 60分以下 : D (`score < 60`)

```
score = int(input("輸入一個整數: "))

if score >= 90:
    print("A:Outstanding")
elif score >=75 and score < 90:
    print("B:Good Job")
elif score >= 60 and score <= 74:
    print("C:Fair")
else:
    print("D:Fail")
```

計算BMI，並印出分類

- 輸入身高、體重 (使用)
- 計算BMI值: $BMI = \frac{\text{體重(公斤)}}{\text{身高}^2(\text{公尺}^2)}$
- 依下列規則印出BMI分類，例如 `bmi=20` 印出「正常範圍」

身體質量指數(BMI) (kg/m ²)	分類
$BMI < 18.5$	體重過輕
$18.5 \leq BMI < 24$	正常範圍
$24 \leq BMI < 27$	過重
$27 \leq BMI < 30$	輕度肥胖
$30 \leq BMI < 35$	中度肥胖
$BMI \geq 35$	重度肥胖

依據BMI值做健康分類建議

- 已知 BMI 值

```
if bmi < 18.5:  
    print("體重過輕")  
elif bmi < 24 and bmi >= 18.5:  
    print("體重正常")  
elif bmi < 27 and bmi >= 24:  
    print("體重過重")  
elif bmi < 30 and bmi >= 27:  
    print("輕度肥胖")  
elif bmi < 35 and bmi >= 30:  
    print("中度肥胖")  
else:  
    print("重度肥胖")
```

格式化列印

- Python 允許以特定格式字串來列印結果
 - `print(f"I am {"鍾健雄"}, and I am {55} years old")`
- `{...}` 稱為「hold place」可以嵌入要印出的資料(常數、變數、運算式)，如“鍾健雄”及 55...等。同時也可以制定列印特性，例如列印長度、小數點位數、正負號、左右中對齊...
 - `quantity = 6`
 - `item = "apple"`
 - `unit_price = 17`
 - `print(f"{quantity} {item} cost ${quantity*unit_price})`
 - `print(f"{quantity} {item} cost {quantity*unit_price:.2f}")`
 - `print(f"{quantity} {item} cost {quantity*unit_price:10.2f}")`
 - `print(f"{quantity} {item} cost {quantity*unit_price:>10.2f}")`
 - `print(f"{quantity} {item} cost {quantity*unit_price:<10.2f}")`
 - `print(f"{quantity} {item} cost {quantity*unit_price:^10.2f}")`
 - `print(f"{quantity} {item} cost {quantity*unit_price:010.2f}")`

```
quantity = 6
item = "apple"
unit_price = 17.2345
6 apple cost 103.40700000000001
6 apple cost 103.41
6 apple cost 0000103.41
```

字串資料格式定義

語法	功能
:s	以 str 的形式輸出文字
:f	以浮點數形式輸出數字
:d	以十進位形式輸出數字
:b	以二進位形式輸出數字
:o	以八進位形式輸出數字
:x, :X	以十六進位形式輸出數字
:e, :E	以科學記號形式輸出數字
:%	以百分比形式輸出數字
:c	以字元形式輸出 (ASCII)

專案#2：計算 bmi 值

- 世界衛生組織建議以身體質量指數（ Body Mass Index, BMI ）來衡量肥胖程度。BMI計算的算法是「體重(公斤)/ 身高²(公尺)²」
- BMI標準（男女通用）/BMI理想範圍
 - $Bmi < 18.5$ (體重過輕)
 - $18.5 \leq bmi < 24$ (體重正常)
 - $24 \leq bmi < 27$ (體重過重)
 - $Bmi \geq 27$ (體重肥胖)
- 請撰寫一個 python 程式，可由鍵盤輸入姓名、體重(公斤)、身高(公分)，計算 BMI 值，並依據 BMI 標準提供體重分類參考，輸出格式
- [姓名] [身高]公分 [體重]公斤 [BMI值] [體重判定]
 - 範例: 王大明 170公分 53公斤 21.8 體重正常

迴圈結構

迴圈程式結構

- 回圈內的程式碼會重複執行
- 重複執行方式
 - 計數型：重複執行固定次數
 - 條件型：依條件判斷執行

計數型迴圈程式結構 (for loop)

```
for i in [1, 2, 3, 4, 5]:  
    print("hello Python %d" % i)
```

- *for ... in* : for 迴圈語法
- *i* : 控制變數
- *[1, 2, 3, 4, 5]* : 計數容器(循序性儲存物件)

執行結果 :

```
hello Python 1  
hello Python 2  
hello Python 3  
hello Python 4  
hello Python 5
```

List 清單(陣列)

- List是一種有序的資料除存容器
 - A = [1, 2, 3, 4, 5]
 - B = ["red", "orange", "green", "blue", "purple"]
- List中的每一個資料都有一個地址(index)
 - 第一個元素的地址(註標)= 0
 - A[1] = 2
 - B[3] = "blue"
- [] 代表空的列表(list)

range() 函數

- **range()**
 - 建立一個序列清單的函數
- **range(5)**
 - $[0, 1, 2, 3, 4]$
- **range(m,n) → [m, m+1, m+2, ...n-1]**
 - Range(1,6) → [1, 2, 3, 4, 5]
- **range(m, n, step) → [m, m+s, m+2s..., m+ps], where m+ps ≤ n**
 - range(1, 6, 2) → [1, 3, 5]

Turtle Graphic

圓形 (circle)

`tt.circle(100)`

三角形(trangle)

`tt.forward(100)`

`tt.right(120)`

`tt.forward(100)`

`tt.right(120)`

`tt.forward(100)`

四方型(rectangle)

`tt.forward(100)`

`tt.right(90)`

`tt.forward(100)`

`tt.right(90)`

`tt.forward(100)`

`tt.right(90)`

`tt.forward(100)`

`tt.right(90)`

迴圈(Loop) – 做同樣的事

- 四方形

tt.forward(100)

tt.right(90)

tt.forward(100)

tt.right(90)

tt.forward(100)

tt.right(90)

tt.forward(100)

tt.right(90)

for i in [1, 2, 3, 4]:

tt.forward(100)

tt.right(90)

自己做做看

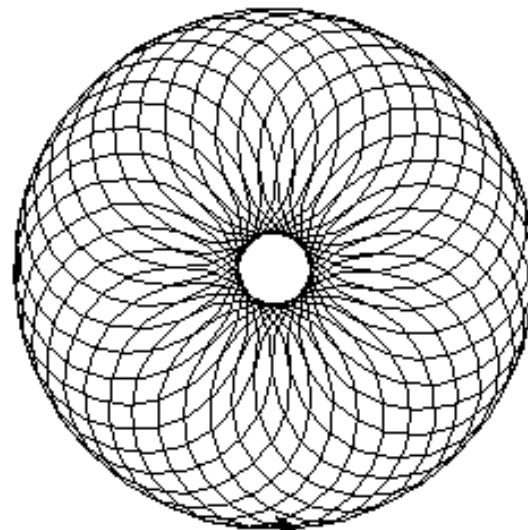
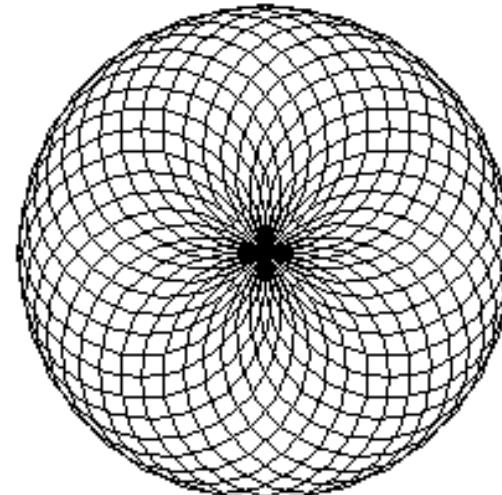
五邊形

```
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)  
tt.forward(100)  
tt.right(72)
```

```
sides = 5  
steps = 100  
innerAngle = 180*(sides-1)/sides  
headingAngle = 180 – inner_angle  
for i in range(sides):  
    tt.forward(steps)  
    tt.right(headingAngle)
```

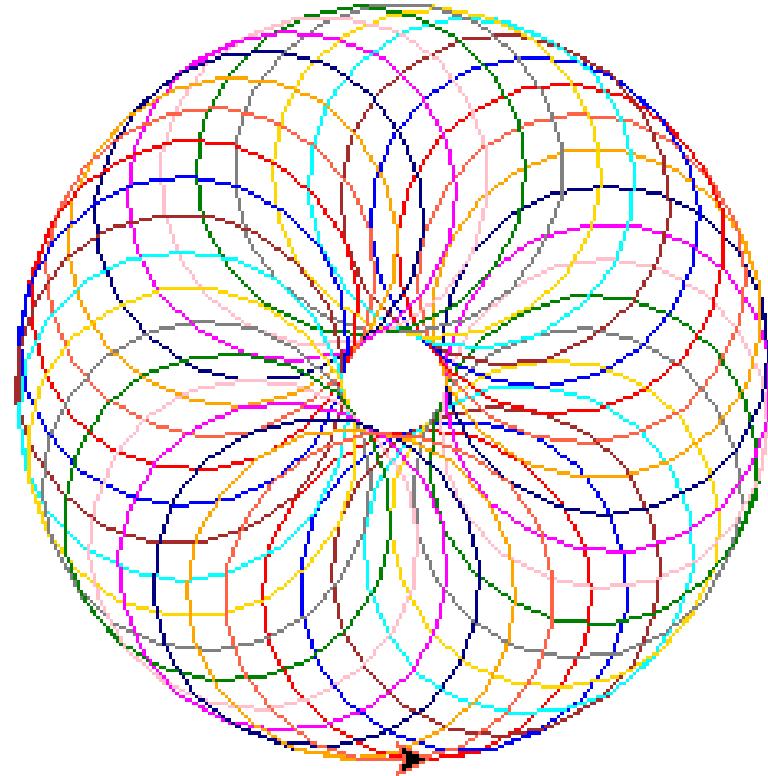
曼陀圖騰

```
from turtle import *
win=Screen()
tt=Turtle()
tt.speed(0)
ang=10
for i in range(360//ang):
    circle(50)
    left(ang)
    # forward(ang*2)
```



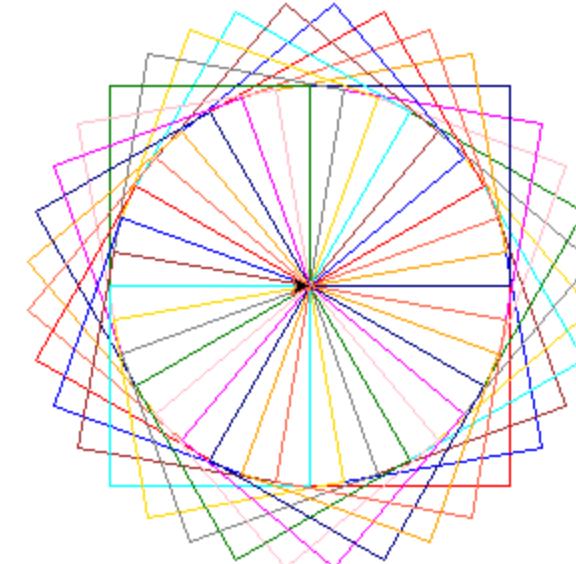
彩色曼陀圖騰1

```
from turtle import *
win = Screen()
tt = Turtle()
tt.speed(0)
ang = 10
colors = ["red", "blue", "brown", "cyan", "gold",
"gray", "green", "pink", "magenta", "navy",
"orange", "tomato"]
for i in range(360//ang):
    pencolor(colors[i%len(colors)])
    circle(50)
    left(ang)
    forward(ang*2)
```



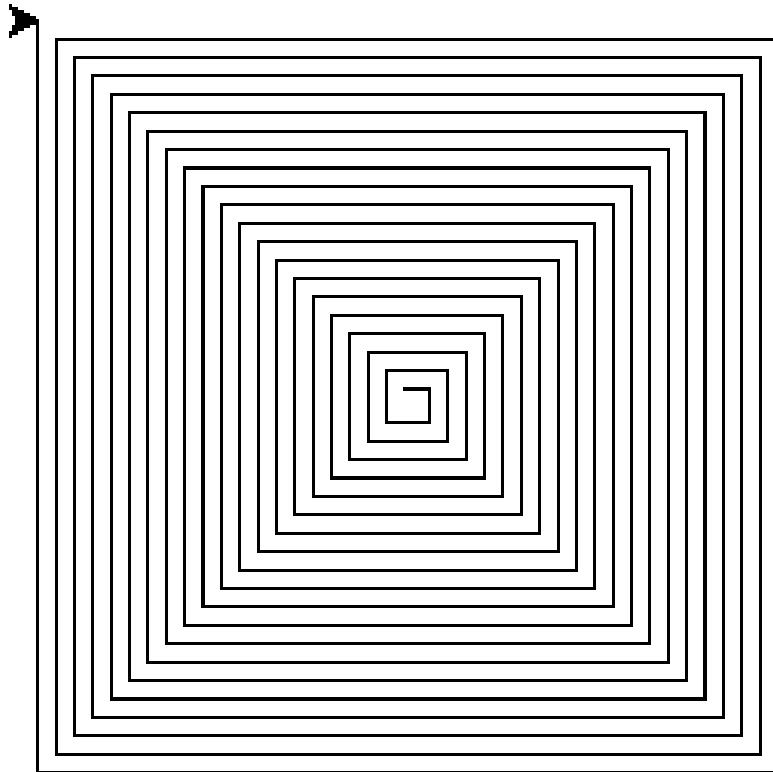
彩色曼陀圖騰2

```
for i in range(360//ang):
    pencolor(colors[i%len(colors)])
    for j in range(4):
        forward(100)
        right(90)
    left(ang)
```



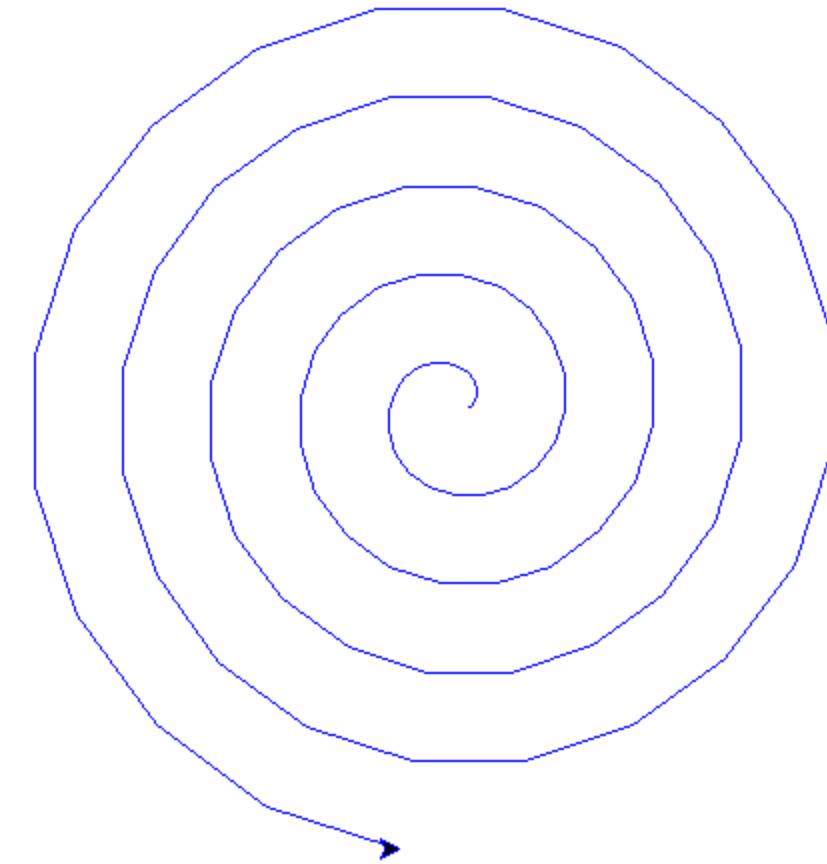
螺旋方形

```
win.clear()  
steps=5  
for n in range(20):  
    # pencolor(colors[n%len(colors)])  
    for j in range(4):  
        steps = steps+3  
        fd(steps)  
        right(90)
```



螺旋圓

```
import turtle  
  
# 設定畫筆速度和顏色  
turtle.speed(0)  
turtle.pencolor("blue")  
  
# 設定螺旋的起始位置  
turtle.penup()  
turtle.goto(0, 0)  
turtle.pendown()  
  
# 畫螺旋  
for i in range(100):  
    turtle.forward(i * 0.7) # 變更前進距離  
    turtle.left(18)         # 變更畫筆方向  
  
turtle.done()
```



For Loop 應用: $1+2+3+4+\dots+10=?$

pig_bank = 0 #空的小豬存錢筒

```
for m in range(1, 11):  
    pig_bank = pig_bank + m  
print(pig_bank)
```

Pig_bank=0 (空存錢筒)
m=1, pig_bank = 0+1=1
m=2, pig_bank = 1+2=3
m=3, pig_bank = 3+3=6
m=4, pig_bank = 6+4=10
m=5, pig_bank = 10+5=15
m=6, pig_bank = 15+6=21
m=7, pig_bank = 21+7=28
m=8, pig_bank = 28+8=36
m=9, pig_bank = 36+9=45
m=10, pig_bank = 45+10=55

range(1,11)

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]



存錢筒

$$1+3+5+7+\dots+19=?$$

```
sum = 0
```

```
for n in range(1, 20, 2):
```

```
    sum = sum + n
```

```
print(sum)
```

range(1, 20, 2)



[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]

1至100數字中，能被3整除，但不能被5整除的數的總和？

```
sum = 0  
for n in range(1, 101):  
    if (n%3==0) and (n%5 != 0):  
        sum = sum + n  
    print(n, end=" ")  
  
print()  
print("總和 = %d" % sum)
```

3 6 9 15 18 21 27 30 33 39 42 45 51

54 57 63 66 69 75 78 81 87 90 93 99

sum = 25

甚麼是「質數」？

- 一個整數 n ，除了「1」及「 n 」可以整除它，找不到其他整數可以整除它。
- n 整除 m ，表示 $m = n \cdot k$ ， $m \div n$ 的餘數為 0
- 以 python 運算式表示為 $m \% n == 0$ ($\%$ 為一個運算式求取 $m \div n$ 的餘數)

$2+2*(n-1)+2$

```
n = int(input())
is_prime = True
for i in range(2, n+1):
    if n%i == 0:
        is_prime = False
if is_prime:
    print("n是質數")
else:
    print("n不是質數")
```

$2+2*(n//2-1)+2$

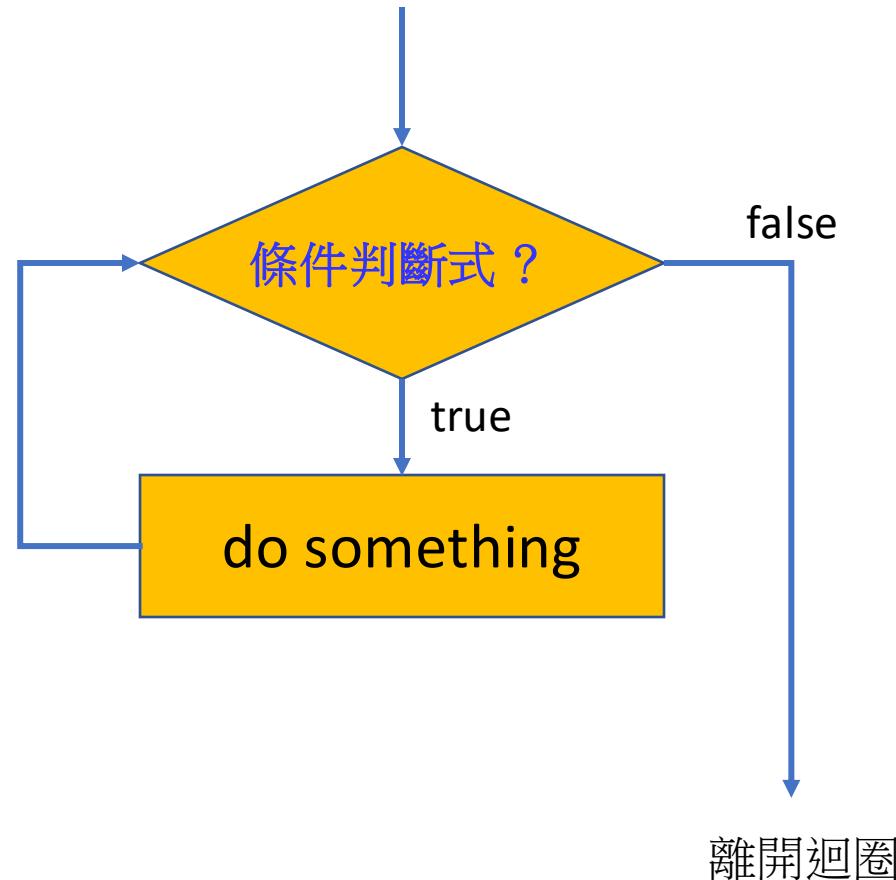
```
n = int(input())
is_prime = True
for i in range(2, n//2+1):
    if n%i == 0:
        is_prime = False
if is_prime:
    print("n是質數")
else:
    print("n不是質數")
```

```
n = int(input())
is_prime = True
for i in range(2, n//2+1):
    if n%i == 0:
        is_prime = False
        break;
if is_prime:
    print("n是質數")
else:
    print("n不是質數")
```

While 迴圈

while 條件判斷式 :

do something



$$1+2+3+\dots+10=?$$

sum = 0

i = 1

while (i<=10):

 sum = sum + i

 i = i + 1

print("sum = %d " % sum)

Break指令

- break指令可以跳離迴圈(for 或 while) , 通常需要使用if 條件式判斷
- 範例：1+2+3+... , 當總和大於100時 , 請問有幾個數？總和=？

```
max = 100
sum = 0
i = 1
count = 0
while True:
    if (sum > max):
        break
        sum += i
        count += 1
        i += 1
print("sum = %d count = %d" % (sum-i, count-1))
```

continue指令

- Continue指令可以跳到迴圈下一回合 (for 或 while) , 通常需要使用 if 條件式判斷
- 範例：餐桌上喊數字，喊到3的倍數就要跳過(禁聲)

```
for i in range(1, 11):
    if (i%3==0):
        continue
    print("大喊:%d" % i)
```

專案#3：判斷質數

- 請撰寫一個 **python** 程式，輸入一個整數n，判斷n是否為質數?
 - 註: 一個自然數，若僅能被1及本身整除者，稱為「質數」

資料結構

在電腦科學中，資料結構（ data structure ）是電腦中儲存、組織資料的方式。不同種類的資料結構適合不同種類的應用，正確的資料結構選擇可以提高演算法的效率

Python基礎資料結構

- 清單(List)
- 元組(Tuple)
- 字串(String)
- 字典(Dictionary)
- 集合(set)

清單(List)

- 清單(串列)是在 python 線性儲存資料的容器，容器中資料可以改變 (mutable)

[1, 2, 3, 4, 5]

利用前後中括
號定義清單

- 在清單中的存放的資料(元素)有連續性、順序性

- $A[0]=1, A[3]=4$

索引	0	1	2	3	4
元素	1	2	3	4	5

- python清單中的資料型態可以不必一樣

- $Y = [1, 2, "魔獸", 1e10]$

- [] 表示空清單

建立清單

- `p = ["陳傑憲", "徐若曦", "林昱珉", "林立", "江坤宇", "張育成"]` 字串清單
- `n = [1, 2, 3, 4, 5]` 數字清單
- `x = [1, 2, 3.14, "王建民"]` 混和資料型態清單
- 建立空清單
 - `e = []`
- 利用`list`函數，結合`range()`，創建數字清單
 - `a = list(range(5))` → `a = [0, 1, 2, 3, 4]`
- `len()`內建函數可獲得清單元素的數量
 - `len(p) = 7`

讀取清單資料

- $p = ["陳傑憲", "徐若曦", "林昱珉", "林立", "江坤宇", "張育成"]$

$p \rightarrow$ 0 陳傑憲 1 徐若曦 2 林昱珉 3 林立 4 江坤宇 5 張育成

- 清單內元素的「索引值」由 0 開始
- 如要讀取清單內第 3 個資料， $p[2]$ （2為索引值）
- $p[4]$ 為清單 p 中第 5 個資料，"江坤宇"
- $\text{len}(p) = 6$ (清單元素數量/長度)

資料定位

- `p = ["陳傑憲", "徐若曦", "林昱珉", "林立", "江坤宇", "張育成"]`
- 使用 `in` 、 `not in` 運算子檢查資料是否存在？結果為布林值
- "林昱珉" `in p` → True
- "王建民" `in p` → False ("王建民" `not in p` → True)
- 使用 `index()` 獲得資料在清單中位置
 - `p.index("江坤宇")` → 4

讀取清單某一範圍資料

- `colors = ["red","blue","yellow","green","purple","orange","cyan"]`
- 讀取單一資料 `colors[2] → "yellow"`
- 讀取範圍資料 [起點: 終點: 間隔]
- `colors[2:5] → ["yellow","green","purple"]`
 - 讀取清單中索引值 2 到 4 的資料，不包含 5
- `colors[:4] → ["red","blue","yellow","green"]`
 - 讀取清單中索引值 0 到 3 的資料，不包含 4
- `colors[3:] → ["green","purple","orange","cyan"]`
 - 讀取清單中索引值 3 到 6 的資料(資料結尾)
- `colors[1:6:2] → ["blue", "green", "orange "]`
 - 讀取清單中索引值 1 到 6 的資料，間隔 2，不包含 6

改變清單內容

- `colors = ["red","blue","yellow","green","purple","orange","cyan"]`
- `colors[1] = "black"`
 - 將`colors`清單中的第2位置資料「`blue`」換為「`black`」
 - `colors = ["red","black","yellow","green","purple","orange","cyan"]`
- `colors[2:4] = ["pink", "magenta"]`
 - `colors = ["red","blue", "pink", "magenta","purple","orange","cyan"]`

增加清單元素

- 利用 `insert()`方法將項目插入指定位置

```
colors = ["red", "blue", "yellow", "green", "purple", "orange", "cyan"]
```

```
colors.insert(1, "magenta")
```

```
colors = ["red", "magenta", "blue", "yellow", "green", "purple", "orange", "cyan"]
```

```
len(colors) ➔ 8
```

- 利用 `extend()`方法，將串列相加

`a = [1, 2, 3, 4, 5]`

`b = [6, 7]`

`a.extend(b)`

`[1,2,3,4,5,6,7]`

- 注意 `a.insert(b)`的結果

`a.insert(5, b)`

`[1, 2, 3, 4, 5, [6, 7]]` #巢狀清單

- 利用 `append()`方法，將項目加到串列尾端
- `colors = ["red","blue","yellow","green","purple","orange","cyan"]`
- `colors.append("magenta")`
- `colors = ["red","blue","yellow","green","purple","orange","cyan", "magenta"]`

- 注意 `colors.append(["black","gray"])`
- `colors = ["red","blue","yellow","green","purple","orange","cyan",
["black","gray"]]`

刪除清單元素

- 利用 `remove()` 方法刪除元素，知道資料內容
- `colors = ["red","blue","yellow","green","purple","orange","cyan"]`
- `colors.remove("yellow")`
- `colors = ["red","blue","green","purple","orange","cyan"]`

- 使用 pop() 方法取出一個項目，並刪除
 - colors = ["red", "blue", "yellow", "green", "purple", "orange", "cyan"]
 - colors.pop() == pop(-1)
- 知道要刪除的元素的位置
 - colors = ["red", "blue", "yellow", "green", "purple", "orange"]
 - colors.pop(3)
 - colors = ["red", "blue", "yellow", "purple", "orange", "cyan"]

- 利用改變元素來刪除某一範圍的資料
- `colors = ["red","blue","yellow","green","purple","orange","cyan"]`
- `colors[2:5] = [] #空清單`
- `colors = ["red","blue", "orange","cyan"]`

二維清單(矩陣)

- 清單的元素可以是另一個清單，構成「多維」清單
- 右表稱為 5x5 二維清單，又可成為 5x5 矩陣，由 5 個一維清單組成

```
m = [[1, 10, 12, 17, 6],  
      [11, 2, 16, 7, 18],  
      [13, 15, 3, 19, 23],  
      [14, 8, 20, 4, 24],  
      [9, 21, 22, 25, 5]]
```

- $m[1] \rightarrow [11, 2, 16, 7, 18]$
- $m[1][3] \rightarrow 7, m[2][2] \rightarrow 3$
- $m[0][5] \rightarrow \text{list range out of range}$

1	10	12	17	6
11	2	16	7	18
13	15	3	19	23
14	8	20	4	24
9	21	22	25	5

```
for i in range(5):  
    print(m[0][i], end=",")  
print()  
# 1, 10, 12, 17, 6  
  
for j in range(5):  
    print(m[j][3], end=",")  
print()  
# 17, 7, 19, 4, 25
```

二維清單操作程式碼

```
m = [[1, 10, 12, 17, 6],\ #用於多行連接
      [11, 2, 16, 7, 18],\
      [13, 15, 3, 19, 23],\
      [14, 8, 20, 4, 24],\
      [9, 21, 22, 25, 5]]

print(m[2])
print(m[1][2])
print(m[1][5])
print(len(m)) # 5 列
print(len(m[1])) # 第2列有5元素

for i in range(5):
    print(m[0][i], end=",")
print()

for j in range(5):
    print(m[j][3], end=",")
print()
```

清單操作方法

操作方法	功能描述	a = [1, 2, 3]
append()	在清單尾端加入一個元素	a.append(4)
copy()	建立一個清單拷貝	b = a.copy()
clear()	清空清單的元素	a.clear()
count()	計算清單中元素的數量	a.count(1)
extend	在清單尾端加入一個清單	a.extend([5,6,7])
index()	獲得清單元素的位置（二個相同元素回傳索引值較小）	a.index(3)
insert()	在清單特定位置(index)插入一個元素	a.insert(3, 99)
pop()	由清單尾端或特定位置讀取並移除一個元素	a.pop() or a.pop(2)
remove()	移除清單中特定元素	a.remove(4)
reverse()	清單中將排列位置反向	a.reverse()
sort()	清單元素排序	a.sort()

專案#4：計算平均成績

- 五位學生的成績紀錄，包括姓名、國文、英文、數學成績。請利用二維矩陣儲存五位學生的成績，其中成績利用 `randrange()` 函數產生隨機各科目成績值。
- 請撰寫一個 `python` 程式，計算每個學生的平均成績，及每個科目的平均成績。

學生姓名	國文	英文	數學	平均
王大明	65	79	95	
張圓圓	75	88	52	
劉長春	82	65	87	
鄭天財	95	85	62	
郝美麗	65	55	45	
各科平均				

元祖 (tuple)

- 元祖是一種循序線性排列的清單，其內容值不可改變(immutable)
 - 清單 (list)、字典 (dictionary)、集合 (set) 都是「可變」 (mutable) 的資料型態
- 元祖利用「小括號」定義，逗號區隔項目
 - `x = (1, 2, 3)`
 - `G = ("李珠垠", "李雅英", "南珉貞")`
- 元祖用於定義一些不會變動的資料，如銀行一天匯率 (0.034, 5.03, 0.25)
- 優點
 - 不可變 (immutable) 的特性，操作比清單快。
 - tuple 的項目不會不小心被更動
 - 占用的空間比較少

tuple的操作運算

- 只要不牽扯到變更值的事情，都與串列很相似，但只要會更動到值，tuple就完全不能做

```
1 red = ('flower', 'blood', 'hair')
2 print(len(red))
3 print(red.index("blood"))
4 print(min(red))
5 print(red.count('flower'))
6 red.insert(1, "skin")
```

```
3
1
blood
1
Traceback (most recent call last):
  File "main.py", line 6, in <module>
    red.insert(1, "skin")
AttributeError: 'tuple' object has no attribute 'insert'
```

tuple物件沒有insert屬性

探索 tuple 資料不可變

- >>> x = (1, 2, 3)
- >>> x[1] = 2
- >>> x[1] = 3.14
- **TypeError: 'tuple' object does not support item assignment**
- Tuple物件不支持內容指定

字符串(string)

字串(string)

- 字串(string)是常用的資料型態之一，可以成對的引號來呈現，單引號、雙引號、三個單引號、三個雙引號都可以拿來表示字串
 - “Hello, how are you doing today”
 - ‘123.45‘
 - “窗前明月光，疑似地上霜”
 - """窗前明月光，
 疑似地上霜，
 舉頭望明月，
 低頭思故鄉。 """
- 單/雙引號同時存在時要注意，
 - “It's a typhoon day.”
 - “Typhoon “Alice” is coming”
 - **SyntaxError: invalide syntax**
 - **Correct: “Typhoon \"Alice\" is coming” (跳脫字元 \")**

- 字串可視為一種「字元清單」資料結構，根據清單操作原理可以擷取子串部分內容

- `str = "Friday night are happy hours"`
- `str[5] → 'y'`
- `str[7:12] → 'night'`
- `str[5:20:3] → 'yitrh' ("Friday night are happy hours")`

19



- 字串內容不可改變 (immutable)

- `str = "Friday night are happy hours"`
- `str[7] = 'N'`

TypeError: 'str' object does not support item assignment

字串常用函式

- **str()** 函式可將各種資料型態轉換為字串
 - `x = str(123)` 將數值123轉換為字串‘123’
- **Len()** 函式取得字串長度
 - `N = len("窗前明月光 · 疑似地上霜")`
 - `Print(n) # 11`
- **type()** 函式顯示資料型別
 - `t = type ("窗前明月光 · 疑似地上霜")`
 - `print(t) #<class 'str'>`
- **sys.getsizeof(資料)** 顯示資料佔記憶體大小
 - `sys.getsizeof("一")#76`
 - `sys.getsizeof("一二")#78` 顯示一個中文字符佔2位元
 - `sys.getsizeof("1")#50`
 - `sys.getsizeof("12")#51` 顯示一個英文字符佔1位元

常用字串物件操作方法

- 方法 (Method) 在使用上需要接在要操作的目標之後，並且以一個句點. 連接
- 轉換英文字母大小寫
 - `upper()` 將所有字母改為大寫
 - `lower()` 將所有字母改為小寫
 - `capitalize()` 將字串中的第一個字母大寫，其餘小寫
 - `title()` 將每個詞第一個字母大寫，其餘小寫
- 辨別字串內容
 - `isalpha()` 字串內只含有字母時，且不是空字串時，返回 `True`
 - `isdecimal()` 字串內只含有數字時，且不是空字串時，返回 `True`
 - `isalnum()` 字串內只含有字母或數字時，且不是空字串時，返回 `True`
- 辨別開頭與結尾
 - `startswith()`、`endswith()` 這兩個字串方法可依照你在 () 內所輸入的引數，辨別字串的起始與結尾是否與引述一致，一致返回值為 `True`，反之為 `False`

- str = “happy Friday”
- print(str.upper()) #’HAPPY FRIEND’
- print(str.title()) #’Happy Friend’
- print(str.isalpha()) #False
- print(str.isdecimal()) #False
- print(str.isalnum()) #False
- print(str.startswith(‘bad’)) #False
- print(endswith(“Friend”)) #False

- **strip()** 將字串兩端的空白字元刪除
 - " Friday night are happy hours ".strip() # 'Friday night are happy hours'
- **Split()** 以特別字符來切割(預設值為「空格」) , 並回傳包含分拆後各個字串的清單(list)
 - " Friday night are happy hours ".split() # ['Friday', 'night', 'are', 'happy', 'hours']
- **join()** 字串結合方法 , 將清單中的各個字串結合起來 , 回傳的值是清單(list)
 - phone = ["886", "02", "2776545"]
 - "-".join(phone) # '886-02-2776545'
- **replace()** 替換方法 , 原來的字串值並不會改變 , replace()方法將指定的部分被替換
 - "Friday night are happy hours".replace("Friday night", "Saturday morning")
 - # 'Saturday morning are happy hours'

- **index()** 或 **find()** 找尋位置方法，字串後加上 `.index(要找的子字串)` 或 `.find(要找的子字串)`，當子字串找的到時，兩個方法的回傳值的是字串首個字母的位置
 - `"Friday night are happy hours".index("happy") #17`
 - `"Friday night are happy hours".find("night") # 7`
 - `"Friday night are happy hours".index("morning")`
`ValueError: substring not found`
- **count()** 計算出現的次數方法，**count()** 計算括號內指定的字串出現的次數
 - “輝達董事長黃仁勳指出輝達將在台北士林科學園區建議研發基地”.`count("輝達")`
 - `# 2`

字典(dictionary)

字典(dictionary)資料結構

- 字典中的元素為無順序排列
- 每一個元素以「映射」方式定義：**key : value**
 - key必須為不可變的資料型態(int、float、str....)
 - 字典中的key必須是唯一(找尋資料的參考值)
- 建立字典方法(左右以大括號定義)
 - `d = {"a": 11, "b":22, "c":33}`
 - `s = {1: "陳傑憲", 2: "江坤宇", 3: "張育成"}`
- 讀取字典的資料，利用 **key** 作為索引
 - `d['a'] → 11`
 - `s[2] → '江坤宇'`
- 新增一筆資料
 - `s[4] = '徐若曦' # {1: '陳傑憲', 2: '江坤宇', 3: '張育成', 4: '徐若曦'}`
- 修改內容
 - `s[2] = "張振瑀" # {1: '陳傑憲', 2: '張振瑀', 3: '張育成', 4: '徐若曦'}`

Python字典操作方法

設定一個字典資料結構 `a = {"a":111, "b": 222, "c":333, "d":444}`

字典方法	說明	執行結果
<code>a.clear()</code>	刪除字典所有資料	<code>print(a) #{}{}</code>
<code>a.copy()</code>	拷貝字典	<code>b = a.copy(); print(b)</code>
<code>a.get(key)</code>	回傳字典中 key 的內容(value)	<code>a.get("b")</code>
<code>a.items()</code>	回傳字典中每一個 key, value	<code>a.items([{"a": 111}, {"b": 222}, {"c": 333}, {"d": 444}])</code>
<code>a.keys()</code>	回傳字典中每一個 key	<code>a.keys(["a", "b", "c", "d"])</code>
<code>a.values()</code>	回傳字典中每一個 value	<code>a.values([111, 222, 333, 444])</code>
<code>a.pop(key)</code>	回傳字典中某一 key 的 vaule，並刪除 key:value	<code>a.pop("c")</code>
<code>a.popitem()</code>	回傳字典中最後輸入的 key:vaule，並刪 除	<code>a.popitem()</code>
<code>a.fromkeys(keylist, default)</code>	由 keylist 建立字典，並設定 default 值	<code>a.fromkeys([1,2,3], "aaa")</code>

利用 for 迴圈讀取字典中的資料

```
dict1 = {"a":111, "b":222, "c":333}
```

```
for a in dict1.items(): #每筆資料以清單 a 形式讀出  
    print(a[0], a[1])
```

```
for k, v in dict1.items(): #每筆資料以 k, v 變數讀出  
    print(k, v)
```

```
for v in dict1.values(): #讀出每筆資料的值  
    print(v)
```

```
for k in dict1.keys(): #讀出每筆資料的鍵值  
    print(k)
```

- `s = {1: '陳傑憲', 2: '張振瑀', 3: '張育成', 4: '徐若曦'}`
- `s.items()`

`dict_items([(1, '陳傑憲'), (2, '張振瑀'), (3, '張育成'), (4, '徐若曦')])`

`<class 'dict_items'>`

```
for p in s.items():
    print(p)

#(1, '陳傑憲')
#(2, '張振瑀')
#(3, '張育成')
#(4, '徐若曦')
```

```
for p in s.items():
    print(p[0], p[1])

#1 陳傑憲
#2 張振瑀
#3 張育成
#4 徐若曦
```

集合(set)資料結構

- 集合中的元素無排列順序且元素不會重複
- 使用{ }建立集合，元素間以「，」隔開
- eg., a = {1, 2, 3, 4, 5}
 - print(type(a)) # <class 'set'>
 - print(a[1]) #**TypeError: 'set' object is not subscriptable** (元素無排列順序)
- eg., b = {1, 2, 3, 3, 4, 5, 4}
 - print(b) #{1, 2, 3, 4, 5} (元素不會重複)
- 集合內只能放不可變的資料型態(**float**、**int**、**str**、**tuple**...)
 - 集合內不可放清單(List)

集合運算

- $A = \{1, 2, 3, 4, 5\}$, $b = \{3, 4, 5, 7, 9\}$
- 聯集「|」： $a | b \rightarrow \{1, 2, 3, 4, 5, 7, 9\}$
- 交集「&」： $a \& b \rightarrow \{3, 4, 5\}$
- 差集「-」： $a - b \rightarrow \{1, 2\}$

集合方法 ($a = \{1, 2, 3\}$)

方法	說明	結果
<code>a.add(5)</code>	將 5 加入 a 中	{1, 2, 3, 5}
<code>a.copy()</code>	傳回 a 的副本	<code>b = a.copy()</code>
<code>a.remove(5)</code>	移除 a 中的 5，若找不到引發 <code>KeyError</code> 例外	{1, 2, 3}
<code>a.pop()</code>	隨機傳回並移除一個資料項，a 為空集合時引發 <code>KeyError</code> 例外	1 <code>a={2,3}</code>

資料排序(sorting)

- 清單排序 `l1 = [5, 2, 9, 1]`

- `l1.sort(reverse=False)` 對 `l1` 進行由小到大排序(升序) · 結果將改變 `l1`
- `print(l1) #[1, 2, 5, 9]` 數字排序
- `aList = ['123', 'Google', 'Runoob', 'Taobao', 'Facebook']` 字串排序
- `aList.sort()`
- `print(aList) # ['123', 'Facebook', 'Google', 'Runoob', 'Taobao']`
- `vowels = ['e', 'a', 'u', 'o', 'I']`
- `vowels.sort(reverse=True) #由大到小(將序)`
- `Print(vowels) #[‘u’, ‘o’, ‘I’, ‘e’, ‘a’]`

排序函數

- 利用 `sorted()`函數對清單、元組、字典...進行排序
- `sorted(iterable, reverse=False, key)`
 - `iterable`: 可重複執行的物件 (如字串、元祖、清單或集合、字典、...)
 - `reverse`: `True/False` 預設 (下降/上升)
 - `key`: 比較函數
- `sorted()`函數回傳一個排序完的物件
- Eg., `a = [2, 4, 6, 3, 1]`
 - `b = sorted(a) # b =[1, 2, 3, 4, 6]`

字典排序

- `emp = {"mary":200, "john":500, "tom":300, "alice":400}`
- `sorted(emp)`
 - 針對 emp 的 key 排序
 - `['alice', 'john', 'mary', 'tom']`
- `sorted(emp.items())`
 - `[('alice', 400), ('john', 500), ('mary', 200), ('tom', 300)]`
- `sorted(emp.items(), key=lambda x: (x[1], x[0]))`
 - `[('mary', 200), ('tom', 300), ('alice', 400), ('john', 500)]`
- `sorted(emp.items(), key=lambda x:(x[1], x[0]), reverse=True)`
 - `[('john', 500), ('alice', 400), ('tom', 300), ('mary', 200)]`

字典練習

- 有關國家的人均所得如下
 - USA 63390
 - Singapore 94500
 - France 46900
 - Japen 45000
 - Korea 40500
 - China 16760
 - Taiwan 25360
- 請將上述資料建立成一個字典，名為 `countrys = {"USA":63390, ...}`
- 請利用 `sorted` 函數找出人均所得最高的國家

專案#5：找出中華職棒全壘打王

- 請利用 **python dictionary** 資料結構儲存 10 位選手上一季全壘打數，傳寫一個程式找出全壘打數最多的選手姓名，並將 10 位選手依全壘打數排序。

函式與套件

函式簡介

- 當程式越來越大，其中有很多重複的程式碼時
 - 程式越來越難理解
 - 重複的程式碼影響寫程式人時間
- 希望程式能模組化
 - 由一大團程式變成一個一個程式模組(函式)
 - 重複的程式碼可以變成程式模組(函式)
- 有設麼好處呢？
 - 程式結構化
 - 程式易了解
 - 容易寫程式

何謂函式？

- 國中數學 $y = f(x)$
 - y 是結果或稱輸出
 - x 是參數
 - $f()$ 稱為函數
- 當給予函數 f 一組參數 x ，就會產生一個結果 y
 - $y = 3x+2$
 - $f(x) = 3x + 2$
 - $x = 5 \rightarrow y = f(5) = 3*5+2 = 17$
 - $x = 11 \rightarrow y = f(11) = 3*11 + 2 = 35$

Python自訂函式(程式模組)

def 函數名稱(參數1, [參數2, 參數3, ...]):

程式碼

[return] 結果

- **def** 自定函式的語法關鍵詞
- 函數名稱，要遵守**變數命名原則**
- (...) 內為參數，一個以上參數用「,」隔開
- **def** 最後要有分號「:」
- 函式內的程式碼要內縮4格

**執行函式時，只要呼叫函式名稱，給予所要求的參數，就可以執行函式內的
程式碼獲得結果**

$y = f(x) = 3x+2$ 函式範例

```
def y(x):  
    a = 3*x+2  
    return a  
  
x = 5  
print(y(x))
```

函式分類

- **函數**

- 有輸入參數、有回傳值

```
def y(x):  
    a = 3*x+2  
    return a
```

- **副程式**

- 有/無輸入參數、沒有回傳值

```
def sayhello(5):  
    for i in range(5):  
        print("hello!!!")
```

函式參數

- 函式參數個數可以1個以上，若超過一個需用「,」隔開
- 參數的型別沒有限制，可以是「整數」、「浮點數」、「字串」...
- 參數分為「型式參數」及「實際參數」

```
def age(x): # x 是形式參數  
    print("我今年", x, "歲")
```

```
y = 20 # y 是實際參數  
age(y)
```

- 參數預設值

```
def age(x=3):  
    print("我今年%d歲" %x)
```

回傳結果

- 在函式的程式碼中碰到**return**關鍵字
 - **return** 後面的常數/變數/運算式為函式運算的結果
 - 程式碼碰到 **return** 就結束函式運算

```
def add(a, b):  
    return (a+b)  
    print("函式結束")
```

```
x = 77  
y = 22  
z = add(x, y)  
print("函式的下一步")
```

比大小

```
def larger(a, b):  
    if (a > b):  
        return a  
    else:  
        return b
```

```
x=5  
y = 9  
print("較大者為 %d" % larger(x,y))
```

變數作用範圍

- **區域變數**

- 在函式內部定義的變數
- 只在函式內有用，離開函式就沒有作用

- **全域變數**

- 在函式外部定義的變數
- 變數影響範圍為整個程式，函式都可引用全域函數

變數範圍例子

```
def add(a, b):  
    print(a, b)  
    print(x, y)  
    return a+b # a, b為區域變數
```

```
x = 2 # x, y為全域變數  
y = 9
```

```
print("%d + %d = %d"% (x, y, add(x, y)))  
print(a, b)
```

```
CD: C:\Users\jc7qx\OneDrive\桌面\py0602  
Current directory: C:\Users\jc7qx\OneDrive\桌面\py0602  
python.exe localvar.py  
Process started (PID=19544) >>>  
2 5  
2 9  
2 + 9 = 7  
Traceback (most recent call last):  
  File "localvar.py", line 10, in <module>  
    print(a, b)  
NameError: name 'a' is not defined  
<<< Process finished (PID=19544). (Exit code 1)  
===== READY =====
```

專案#6: 1-100000的數字中質數

- 請撰寫一個程式，建立判斷質數的函式，並利用質數判斷函式找尋1到100000的數字有幾個質數？質數的總和？

Python math套件

- import math

數學常數

- math.e
- math.pi
- math.inf
- math.nan

數學函數 (math.函數名)

- ceil(x), floor(x), trunc(x)
- fabs(x), factorial(x), gcd(x), fmod(x, y), frexp(x), fsum(數列) , modf(x)
- exp(x), log(x[, b]), log2(x), log10(x), pow(x, y), sqrt(x)
- sin(x), cos(x), tan(x),
- asin(x), acos(x), atan2(y,x)
- degrees(x), radians(x)

隨機亂數模組

- 導入隨機變數模組
 - `import random`
- 產生 0-1 隨機論述
 - `random.random()`
- 在清單中隨機選取一個值
 - `random.choice(清單)`
- 在 n 到 m 範圍中隨機選取一個值
 - `random.randrange(n, m, s)`
- 在一個範圍中均值選取一個值
 - `random.uniform(m, n)`
- 設定亂數子
 - `random.seed(n)`
- `random.shuffle(清單)`
 - 隨機洗牌

專案#7：撲克牌洗牌

- 請寫一個 **python** 程式模擬撲克牌洗牌後，發出 5 張牌。
 - 撲克牌有四種花色(hearts, spades, clubs, 及diamonds)，每一花色有13張牌 A, 2, 3, 4, ..., 10, J, Q, K

專案#8：烏龜隨機散步

- 在一個範圍中($W \times L$)小烏龜隨機散步，每次隨機走50至100步，到達步數後隨機360度轉向，碰到邊界時轉向180度。
- 請撰寫一個python程式使小烏龜進行隨機散步200次。

OS模組

方法	參數	說明
<code>getcwd()</code>		取得目前程式的工作資料夾路徑
<code>chdir()</code>	<code>path</code>	改變程式的工作資料夾路徑
<code>mkdir()</code>	<code>folder</code>	建立資料夾
<code>rmdir()</code>	<code>folder</code>	刪除空資料夾
<code>listdir()</code>	<code>folder</code>	列出資料夾裡的內容
<code>open()</code>	<code>file, mode</code>	開啟檔案
<code>write()</code>	<code>string</code>	寫入內容到檔案
<code>rename()</code>	<code>old, new</code>	重新命名檔案
<code>remove()</code>	<code>file</code>	刪除檔案
<code>stat()</code>	<code>file</code>	取得檔案的屬性
<code>close()</code>	<code>file</code>	關閉檔案
<code>path</code>		取得檔案的各種屬性
<code>system</code>		執行系統命令 (等同使用 cmd 或終端機輸入指令)

os 模組練習

- `import os`
- `print(os.getcwd())`
- `print(os.listdir())`
- `print(len(os.listdir()))`
- `print(os.stat("107test1.py"))`

檔案處理

- 讀取檔案

```
file = open("test.txt", "r")
content = file.read()
print(content)
file.close()
```

- 寫入檔案

- file = open("test.txt", "w")
- file.write("hello, python")
- file.close()

- 附加檔案

- file = open("test.txt", "a")
- file.write("附加新的一筆資料於結尾")
- file.close()

- 檔案結束

- 檔案開啟處理後，正常程序需要關閉檔案，否則會佔據電腦資源

file.close()

mode	作業
r, rb, r+, rb+	讀取檔案
w, wb, w+, wb+	寫入檔案
a, ab, a+, ab+	附加檔案
x, xb, x+, xb+	建立檔案
b: 二位元檔案 +: r+(read/write), w+(write/read), a+(append/read), x+(read/write)	

檔案處理異常

程式執行出錯意外處理

```
try:  
    file = open("test.txt", "r")  
    content = file.read()  
    print(content)  
  
finally:  
    file.close()
```

確保檔案運作完一定關閉檔案

```
with open("geeks.txt", "r") as file:  
    content = file.read()  
    print(content)
```

檔案讀取函式

- **file.read([size])**
 - 從文件讀取指定的位元組數，如果未給定或為負則讀取所有
- **file.readline([size])**
 - 讀取整行，包括 “\n” 字元
- **file.readlines([size])**
 - 讀取所有行並返回清單，若給定sizeint>0，則是設置一次讀多少位元組，這是為了減輕
讀取壓力

CSV檔案處理

- CSV文件是一種純文字檔案格式，廣泛用於儲存和共享表格資料。CSV 代表「逗號分隔值」。
- CSV 檔案的內部結構，其中每一行代表一行數據，逗號用於分隔列

張圓圓, 75, 88, 52

劉長春, 82, 65, 87

鄭天財, 95, 85, 62

郝美麗, 65, 55, 45

- Excel的檔案可以儲存為 csv 格式檔案

- Python 處理 csv 方式

- 開啟 csv 檔案
- 讀取一列資料
- 利用 `str.split(",")` 方解欄位
- 將每一個欄位值寫入清單
- 處理所有列資料建立二維清單
- 根據清單規則處理數據

學生姓名	國文	英文	數學
王大明	65	79	95
張圓圓	75	88	52
劉長春	82	65	87
鄭天財	95	85	62
郝美麗	65	55	45

讀取 csv 檔案程式碼解析

```
import os
os.chdir("c:/Users/User") #設定檔案位置
dataTb = [] #建立一個資料表
with open("students.txt", "r", encoding="utf8") as fp: #開啟students.txt，留意中文編碼
    ll = fp.readlines() #讀取csv檔中每一列資料
    for line in ll:
        dline = line.split(",") #將每一列資料分解為清單
        dline[1]=int(dline[1]) #將字串資料轉換為整數
        dline[2]=int(dline[2])
        dline[3]=int(dline[3])
        dataTb.append(dline) #將清單併入資料表(形成二維清單)

    for d in dataTb: #列印二維清單
        print(d)
```

python物件導向程式

特別套件

Numpy簡介

- **numpy**是以**Pyhton**為基礎的科學計算套件
- **Numpy** 提供多維陣列運算，包含數學、邏輯、**shape manipulation**, 排序, 選取數據, **I/O**, 離散傅立葉轉換, 基本線性代數, 基本統計函數, 隨機模擬等，運算效能高
- **NumPy** 套件核心為 **ndarray** 物件，資料型別同質的 **n** 維陣列
- 使用 **numpy** 前須導入套件
 - `import numpy as np`

建立陣列

- 1-D

- `a = np.array([1,2,3,4])` #array([1,2,3,4])
- `a.ndim` #1
- `a.shape` #(4,)
- `len(a)` #4

1	2	3	4
---	---	---	---

- 2-D

- `b = np.array([[0,1,2],[3,4,5]])` # 2x3 array
- `b.ndim` #2
- `b.shape` #(2,3)
- `len(b)` #2

0	1	2
3	4	5

Numpy array 元素資料型別

- `numpy.bool, boolean`
- `numpy.int8, numpy.uint8 (numpy.byte)`
- `numpy.int16, numpy.uint16 (numpy.short)`
- `numpy.int32, numpy.uint32`
- `numpy.intp, numpy.uintp`
- `numpy.int64 (numpy.long)`
- `numpy.float16 (numpy.half)`
- `numpy.float32 (numpy.single)`
- `numpy.float64 (numpy.double)`
- `numpy.float96 (numpy.longdouble)`

特殊矩陣

```
np.ones((3,3))
```

```
array([  
    [1., 1., 1.],  
    [1., 1., 1.],  
    [1., 1., 1.]])
```

```
np.zeros((3,3))
```

```
array([  
    [0., 0., 0.],  
    [0., 0., 0.],  
    [0., 0., 0.]])
```

```
np.identity(3)
```

```
array([  
    [1., 0., 0.],  
    [0., 1., 0.],  
    [0., 0., 1.]])
```

```
np.full((3,3), 5)
```

```
array([  
    [5, 5, 5],  
    [5, 5, 5],  
    [5, 5, 5]])
```

```
np.empty((3,3))
```

• 切割陣列(array slicing)

- Slicing 1-D array

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

- Slicing 2-D array

```
arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])
arr[1, 1:4] or arr[0:2, 2] #array([3,8])
```

• 選取數據

- Access 1-D array elements

- arr = np.array([1, 2, 3, 4])
- arr[2] #3

- Access 2-D array elements

- arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
- arr[1, 4] #10

數學運算 (針對每一個陣列元素)

- 加減乘除
- **square, sqrt, power, ...**
- 三角函數
 - **sin, cos, tan, ...**
- 指數/對數
 - **exp, log, log10**
- 統計函數
 - **sum, mean, median, std, var, min, max**

```
import numpy as np
x=np.array([1, 2, 3])
print(np.square(x))
print(np.sqrt(x))
print(np.power(x, 5))
print(np.cos(x))
print(np.exp(x))
print(np.log(x))
print(np.mean(x))
print(np.var(x))
```

```
[1 4 9]
[1.           1.41421356 1.73205081]
[ 1  32 243]
[ 0.54030231 -0.41614684 -0.9899925 ]
[ 2.71828183  7.3890561  20.08553692]
[0.           0.69314718 1.09861229]
2.0
0.6666666666666666
```

亂數產生

- 設定亂數子

`np.random.seed(200)`

- 生成5筆[0,1)亂數的陣列

`np.random.random(5)`

- 生成5筆[10,100)整數亂數

`np.random.randint(10,100,size=5)`

- 生成5筆[0,1)符合normal distribution的亂數(uniform...)

`np.random.normal(0,1,5)`

- 由一個陣列中隨機抽樣5筆數據 with replacement

`np.random.choice(array, 5, replace=True)`

- 將一個陣列打亂重排

`np.random.shuffle(array)`

基本線性代數

- 線性代數在機器學習演算法開發中扮演重要的腳色
- 在線性代數，使用向量(vector)、矩陣(matrix)、及線性方程式(linear equations)解決問題
 - 向量代表一個數值與方向的物理量，它是用於解決工程或機器學習問題的工具
 - 矩陣在許多領域用來作為向量轉換工具
 - 線性方程式常用描述實際問題其結果與影響因子間呈現線性關係，求解線性方程式幫助了解結果與影響因子間關聯，線性方程式可以使用向量與矩陣來表示

$$\begin{cases} 3x_1 + 2x_2 = 12 \\ 2x_1 - 1x_2 = 1 \end{cases}$$

$$\begin{bmatrix} 3 & 2 \\ 2 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 12 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & -1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 12 \\ 1 \end{bmatrix}$$

- Python SciPy 套件提供線性代數工具模組，這個模組的底層是以 numpy 為基礎

線性代數函數

- 矩陣加減

```
import numpy as np
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])

# Matrix Addition
C = A + B
print("Matrix Addition:\n", C)

# Matrix Subtraction
D = A - B
print("Matrix Subtraction:\n", D)
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 11 & 12 \end{bmatrix}$$
$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} - \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} -4 & -4 \\ -4 & -4 \end{bmatrix}$$

- 矩陣相乘

```
import numpy as np
A = np.array([[1, 2], [3, 4]])
B = np.array([[5, 6], [7, 8]])

# Matrix Multiplication
C = A @ B
print("Matrix Multiplication with @:\n", C)

D = np.dot(A, B)
print("Matrix Multiplication with np.dot():\n", D)
```

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$A \cdot B = \begin{bmatrix} 1 * 5 + 2 * 7 & 1 * 6 + 2 * 8 \\ 3 * 5 + 4 * 7 & 3 * 6 + 4 * 8 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

- 矩陣轉置

```
import numpy as np
A = np.array([[1, 2], [3, 4]])
|
# Transposing the matrix
A_T = A.T
print("Transpose of A:\n", A_T)
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \xrightarrow{\text{轉置}} \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

- 行列式值

```
import numpy as np
A = np.array([[1, 2], [3, 4]])
|
# Determinant of the matrix
det = np.linalg.det(A)
print("Determinant of A:", det)
```

$$\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} = 1*4 - 2*3 = -2$$

- 反矩陣

```
import numpy as np
A = np.array([[1, 2], [3, 4]])

# Inverse of the matrix
A_inv = np.linalg.inv(A)
print("Inverse of A:\n", A_inv)
```

- 解線性方程式

```
import numpy as np
A = np.array([[3, 1], [1, 2]])
b = np.array([9, 8])

# Solving the linear system Ax = b
x = np.linalg.solve(A, b)
print("Solution of the linear system:", x)
```

$$A \times A^{-1} = I$$

$$A^{-1} = \frac{\text{adj}(A)}{\det(A)}$$

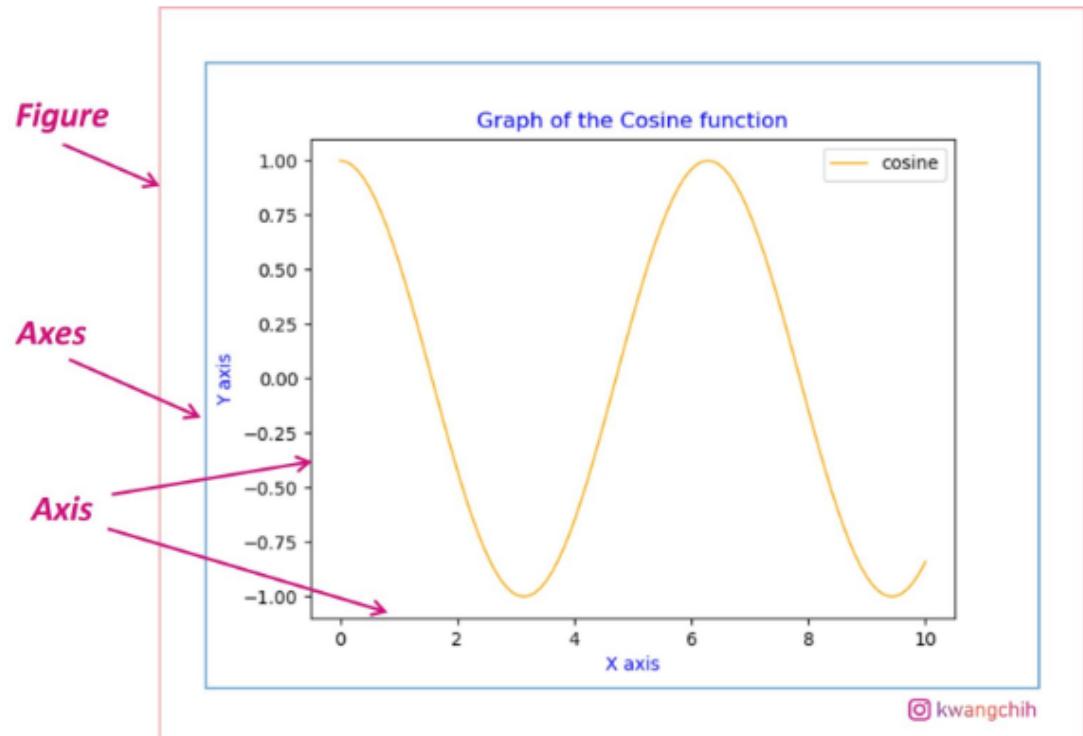
Matplotlib 科學繪圖

- Matplotlib 是一個 Python 2D 繪圖函式庫,用來繪圖、圖表及資料視覺化的一個綜合函式庫(Library)
- Matplotlib 是開源(Open Source) 工具包
- matplotlib 安裝
 - Pip install matplotlib
- 使用 matplotlib
 - import matplotlib
 - matplotlib.__version__ # 檢查版本號

認識繪圖結構

```
import matplotlib.pyplot as plt  
  
import numpy as np  
  
fig = plt.figure()  
  
ax = fig.add_subplot(111)  
  
x = np.linspace(0, 10, 100)  
  
y = np.cos(x)  
  
plt.plot(x,y)  
  
plt.show()
```

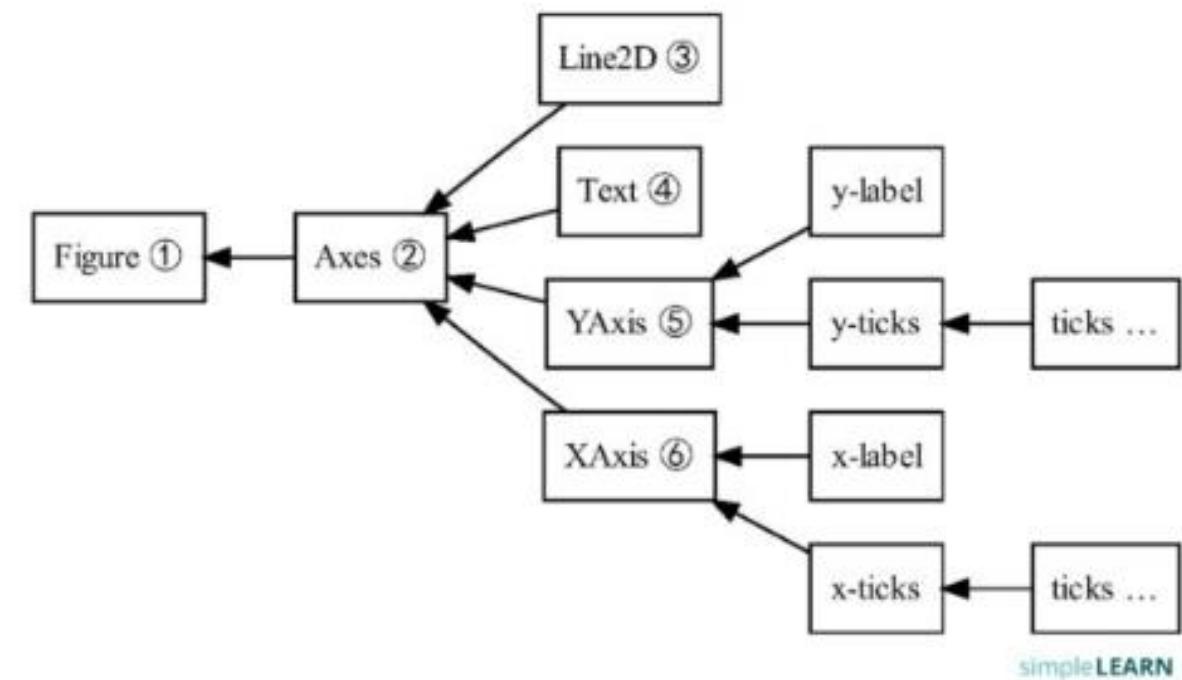
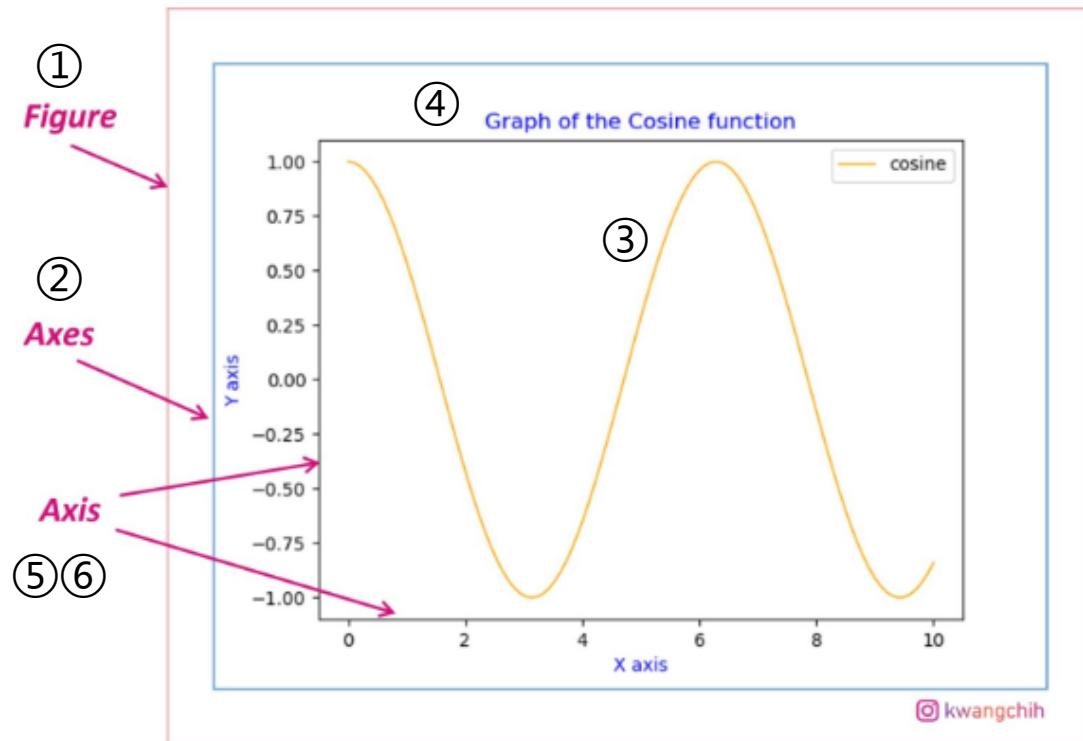
繪圖結構



@ kwangchih

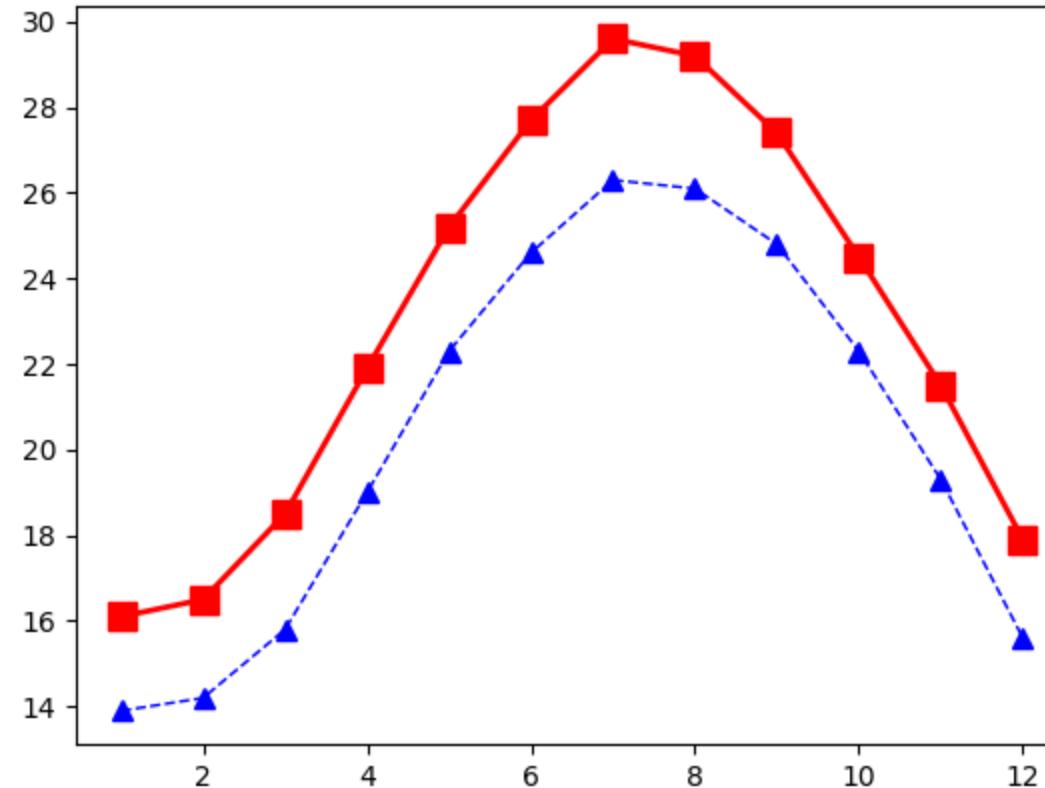
- 步驟一：準備資料
- 步驟二：建立繪圖
- 步驟三：客製化繪圖
- 步驟四：儲存圖形
- 步驟五：展示圖形

matplotlib繪圖的物件階層



基本繪圖參數

- **x, y**
- **Color = "red" | "blue" | "black"** #設定線條顏色
- **Marker = 'o' | 's' | 'x'** #資料點標形狀
- **Markersize = 10** #資料點標大小
- **Linestyle = '-' | '--' | ':'** #線型
- **Linewidth = 3** #線寬

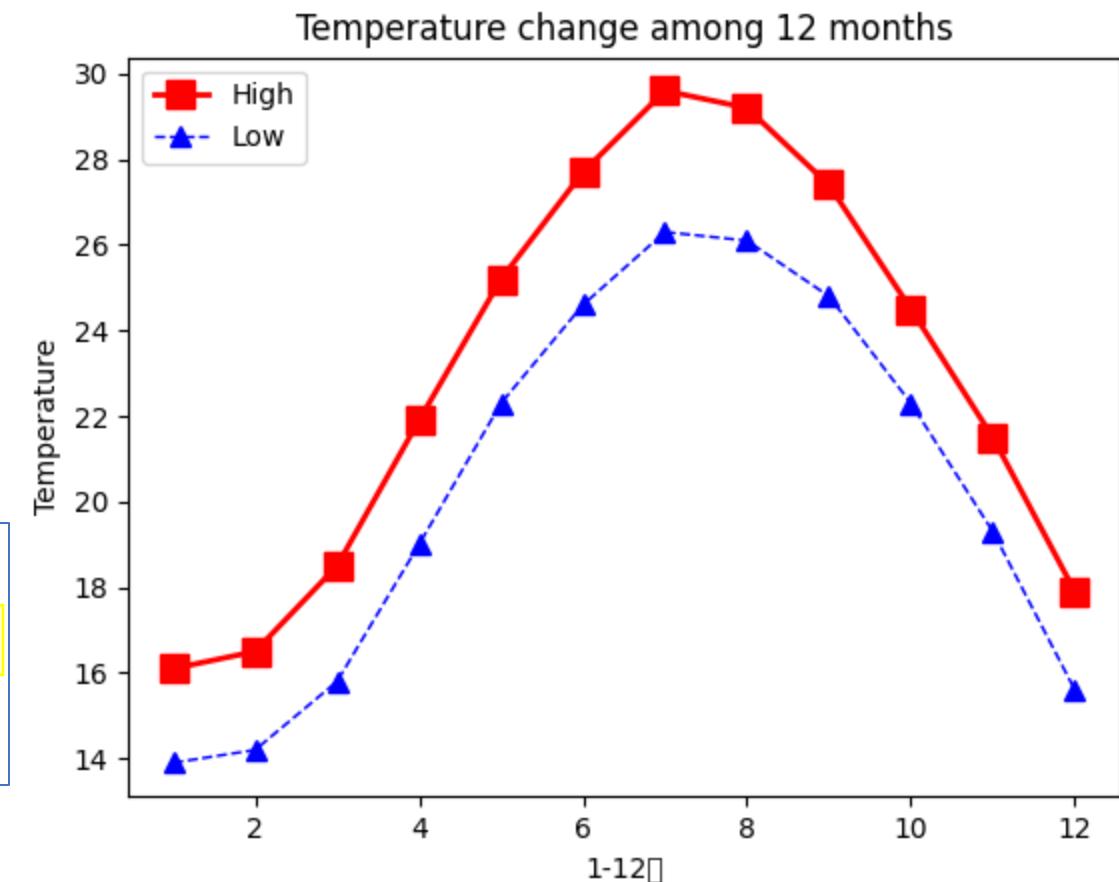


圖標題、軸標題、圖例

- `plt.xlabel("year")`
- `plt.ylabel("Temperature")`
- `plt.title("10年溫度變化")`
- `plt.legend(loc="upper left")`

```
import matplotlib.pyplot as plt
htemp = [16.1,16.5,18.5,21.9,25.2,27.7,29.6,29.2,27.4,24.5,21.5,17.9]
ltemp=[13.9,14.2,15.8,19.0,22.3,24.6,26.3,26.1,24.8,22.3,19.3,15.6]
month = range(1, 13)
plt.plot(month, htemp, color="red", marker='s', markersize=10, linestyle='-', linewidth=2, label="High")
plt.plot(month, ltemp, color="blue", marker='^', markersize=7, linestyle='--', linewidth=1, label="Low")
plt.legend(loc="upper left")
plt.xlabel("1-12月")
plt.ylabel("Temperature")
plt.title("Temperature change among 12 months")
plt.show()
```

legend

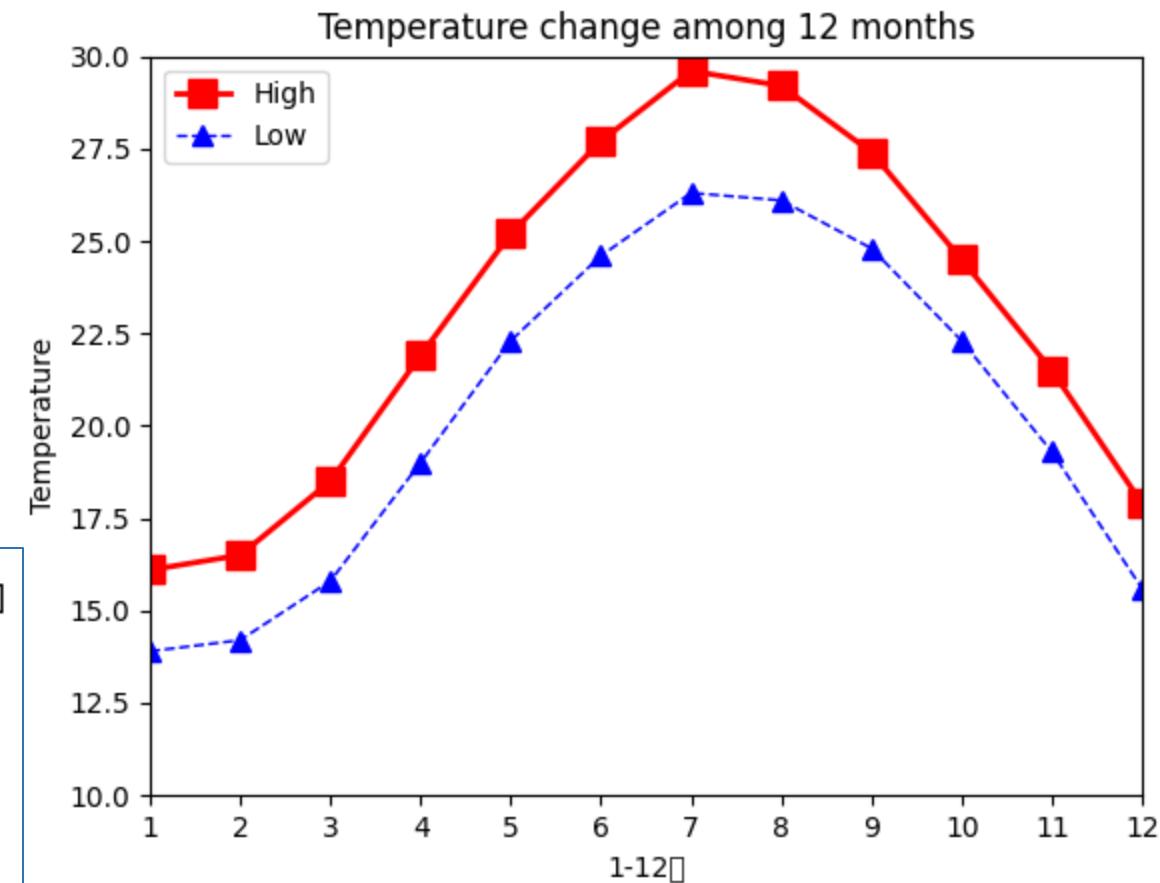


中文顯示有問題

軸刻度控制

- `plt.xlim(1,12)`
- `plt.ylim(20,30)`
- `plt.xticks([1,2,3,4,5,6,7,8,9,10,11,12])`

```
import matplotlib.pyplot as plt
htemp = [16.1, 16.5, 18.5, 21.9, 25.2, 27.7, 29.6, 29.2, 27.4, 24.5, 21.5, 17.9]
ltemp=[13.9, 14.2, 15.8, 19.0, 22.3, 24.6, 26.3, 26.1, 24.8, 22.3, 19.3, 15.6]
month = range(1, 13)
plt.plot(month, htemp, color="red", marker='s', markersize=10,
         linestyle='-', linewidth=2, label="High")
plt.plot(month, ltemp, color="blue", marker='^', markersize=7,
         linestyle='--', linewidth=1, label="Low")
plt.xlim(1,12)
plt.ylim(10,30)
plt.legend(loc="upper left")
plt.xlabel("1-12月")
plt.ylabel("Temperature")
plt.title("Temperature change among 12 months")
plt.xticks([1,2,3,4,5,6,7,8,9,10,11,12])
plt.show()
```

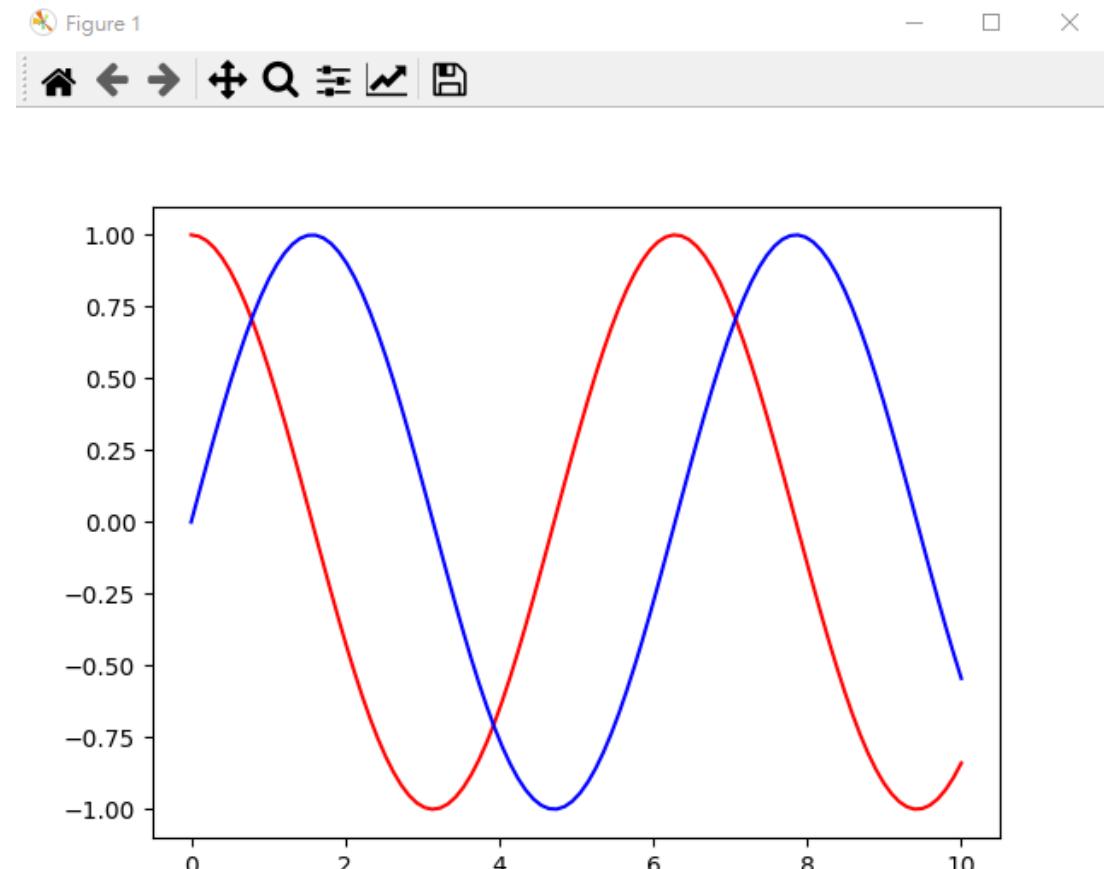


Matplotlib支援之圖表類型

- 折線圖 `.plot(x,y)`
- 散點圖 `.scatter(x,y)`
- 長條圖 `.bar(x,y)`
- 圖像 `.imshow(image, cmap)`
- 直方圖 `.hist(x)`
- 圓餅圖 `.pie(x)`

多圖顯示

```
import matplotlib.pyplot as plt  
import numpy as np  
  
fig = plt.figure()  
  
ax = fig.add_subplot(111)  
  
x = np.linspace(0, 10, 100)  
  
y = np.cos(x)  
  
z = np.sin(x)  
  
plt.plot(x,y, color="red")  
  
plt.plot(x,z, color="blue")  
  
plt.show()
```



sympy 符號運算

- SymPy 幫我們進行「符號化計算」，符號化計算帶入運算的不是某個具體的數值，而是抽象的數學符號，並課將最終得到的結果進行簡化
- SymPy 是一個用於執行符號計算的 Python 套件庫 ([SymPy Live Shell](#))
- SymPy 適用於基本符號算術、微積分、代數、離散數學、量子物理等領域
- SymPy 能夠將結果格式化為多種格式，包括 LaTeX、MathML 等
- SymPy 的應用領域包括多項式、微積分、離散數學、矩陣、幾何學、繪圖、物理、統計數據、組合數學
- 安裝 sympy
 - Pip install sympy
- 使用 sympy
 - import sympy
 - Sympy.__version__