第 2 章

從 Scratch 到 Python

學習過七、八年級的 Scratch 課程之後，相信同學們對於程式設計已經有初步的認識。本章選用 Python 文字式程式語言進行小專題，透過學習過的 Scratch 範例，期望同學能夠順利地將問題從積木式語言銜接到文字式語言。除了 Python 的基本語法外，只要配合不同的套件，Python 就能變身成為不同的工具，進而解決各式各樣的問題。此外，因為 Python 在各種應用上都有龐大的社群，若是需要解決某個問題的方法，只要使用搜尋引擎尋找，會發現這個問題可能已經很多人詢問，也可以找到各種不同的解決方法，而這就是 Python 強大的主因之一。

學習完數字與字串運算的概念之後，接下來我們用「計算學期成績」的程式為例，學習關係運算符號與選擇結構的概念吧！

範　例－計算學期成績程式

請設計一個程式，讓使用者輸入各項成績後，再將各項成績轉換為學期成績，並判斷學期成績是否及格？（其中，作業成績占 40%，測驗成績占 40%，平時成績占 20%，學期成績 60 分為及格分數。）

中、英文對照

保留字		保留字		變數／串列名稱	
if	如果	False	假	grade	成績
else	否則	True	真		
		elif（else if）	否則		

Scratch 積木

Python 程式碼

```
1 score1 = int(input('請輸入作業成績:'))
2 score2 = int(input('請輸入測驗成績:'))
3 score3 = int(input('請輸入平時成績:'))
4 grade = score1*0.4 + score2*0.4 + score3*0.2
5 print('學期成績是' + str(grade))
6 if grade < 60:
7     print('不及格')
8 else:
9     print('及格')
```

範例程式說明

1. 第 1 行：將使用者輸入的第一個數字存到變數 score1。
2. 第 2 行：將使用者輸入的第二個數字存到變數 score2。
3. 第 3 行：將使用者輸入的第三個數字存到變數 score3。
4. 第 4 行：計算學期成績。
5. 第 5 行：因為 grade 是浮點數，所以須強制轉換成字串，才能進行字串相加，並將結果輸出。
6. 第 6 ～ 9 行：使用雙向選擇結構判斷，如果 grade 小於 60，呈現出不及格，否則呈現出及格。

保留字是文字式程式語言中，已經具有特定意義的英文單字，不可用來作為使用者自訂的變數名稱。

2-1 認識 Python 程式語言

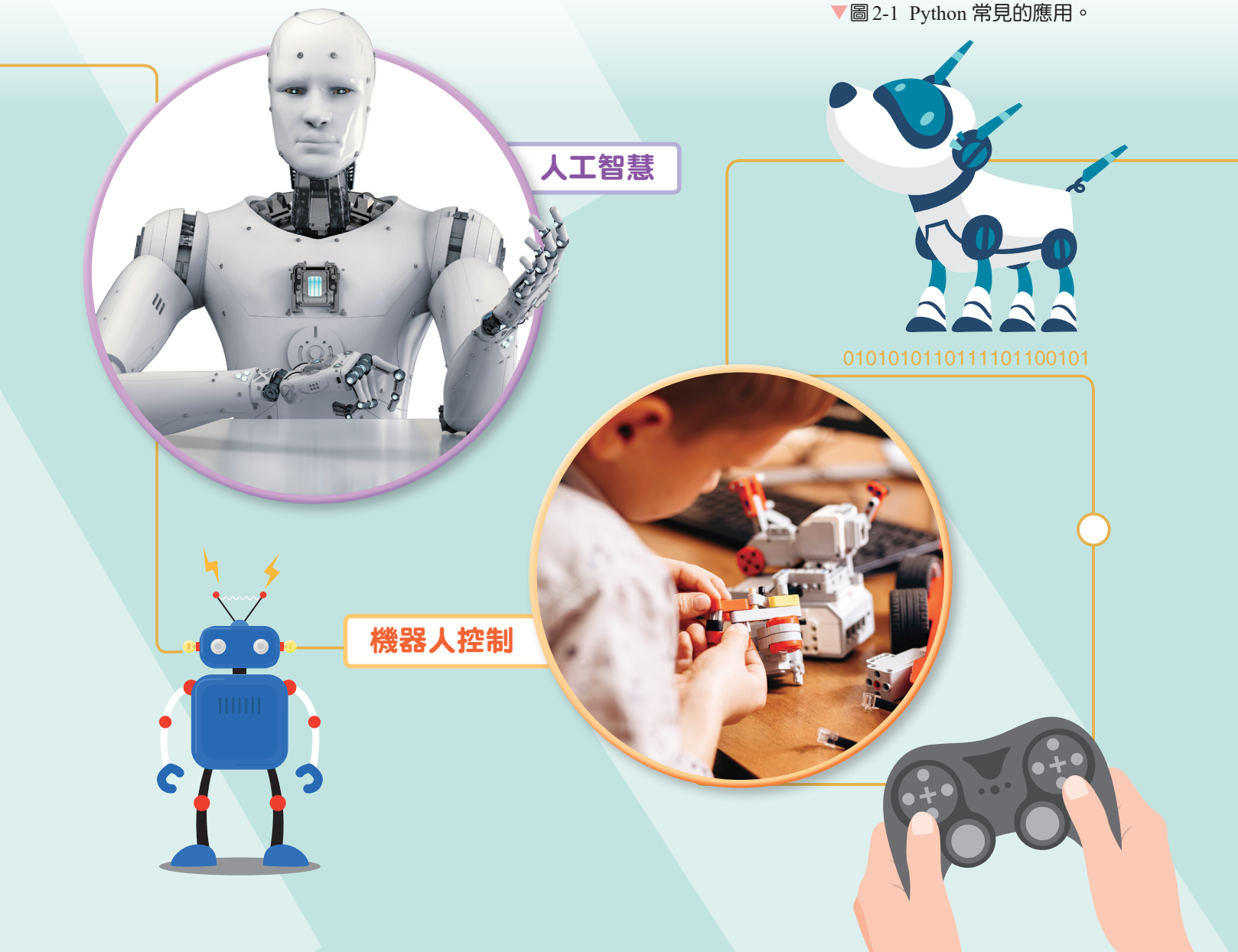
本課程七、八年級都使用 Scratch 來設計程式，但積木式程式在應用時受到比較多的限制，因此安排學習者從積木式轉到文字式程式語言（例如：Python、C++ 等），以備未來撰寫規模較大的程式。

本章選擇目前頗受重視的 Python，它是一種廣泛使用且功能強大的通用型程式語言，語句易讀、易懂，很適合做為第一個學習的文字式程式語言。在介紹 Python 的過程中，同時以過去學過的 Scratch 積木程式為例，對照 Python 的程式碼，讓學習者可以很快進入文字式程式語言的世界。

自 1991 年由吉多・范羅蘇姆（Guido van Rossum）研發並釋出後，陸續有許多程式設計者為 Python 提供了很多自行開發的模組，用來解決各式各樣的問題，因此學習 Python 不僅可以支援大部分的應用，也能從廣大的 Python 社群中獲得豐富的資源，其中常見的應用有資料分析、科學運算、網站開發、人工智慧、機器人控制等（圖 2-1）。

至於 Python 這個名稱的由來，在查閱過英文字典後，你會發現其中文翻譯為「蟒蛇」，然而事實上，它是取名於吉多・范羅蘇姆很喜歡的一部英國喜劇蒙提・派森的飛行馬戲團（Monty Python's Flying Circus）。

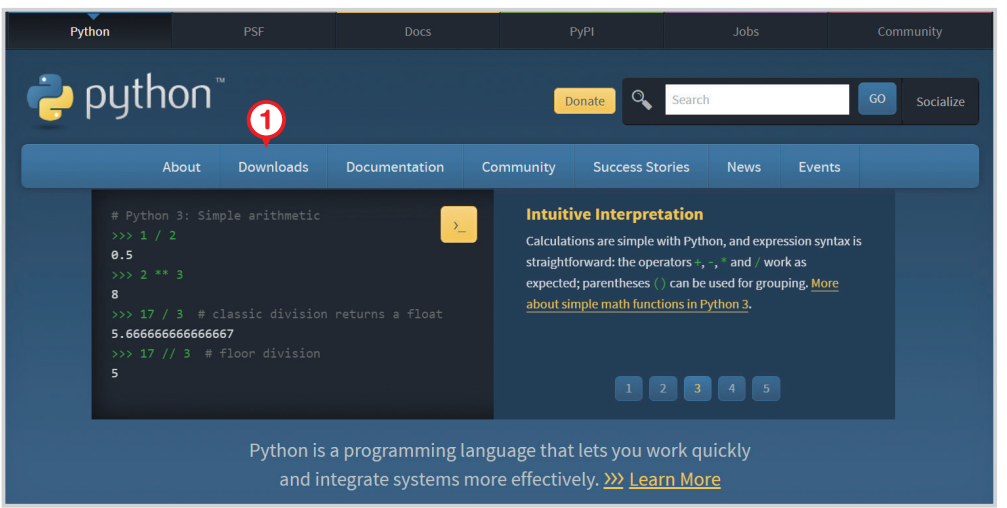
▼圖 2-1 Python 常見的應用。



2-1-1 Python 的下載與安裝

步驟 1 進入 Python 的官方網站（https://www.python.org/）。

① 進入官網，點擊 Downloads。

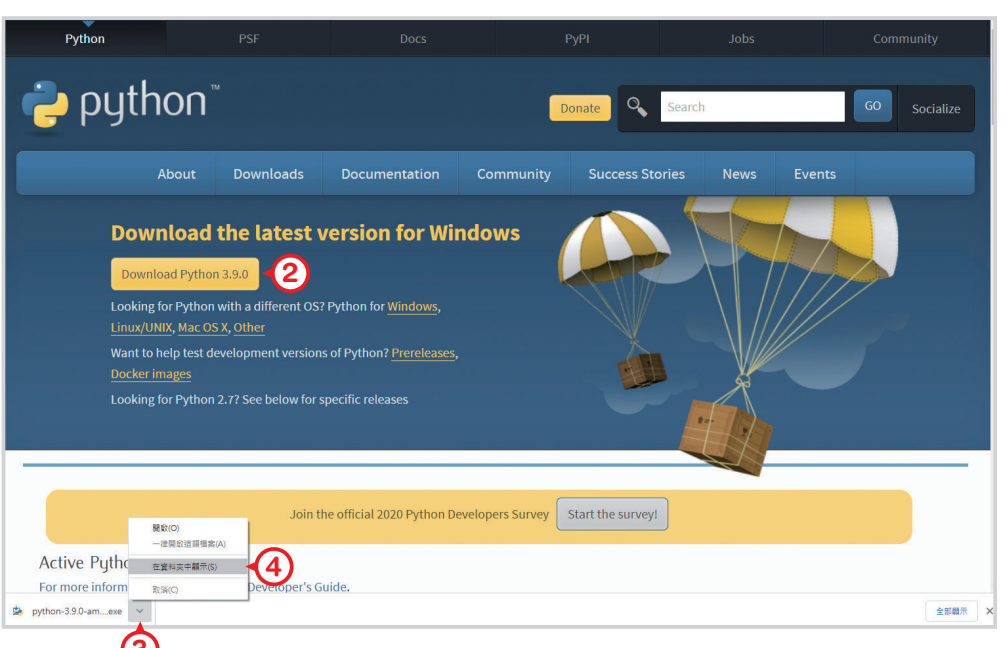


步驟 2 下載 Python。

② 點擊 Download Python 3.9.0。

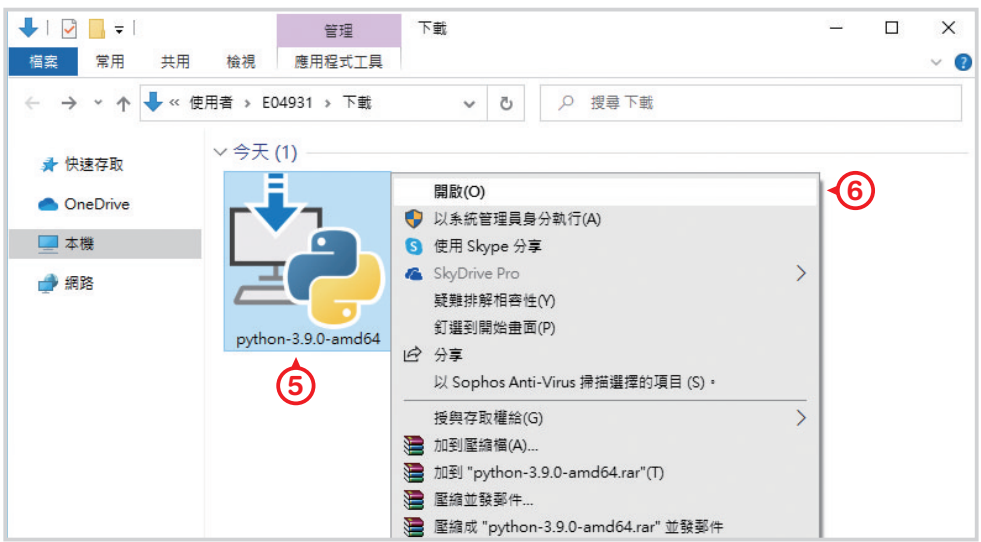
③ 下載完成後，點選畫面下方的 ⌄。

④ 在彈出視窗中，點選在資料夾中顯示。



⑤ 在下載的檔案上點選滑鼠右鍵。

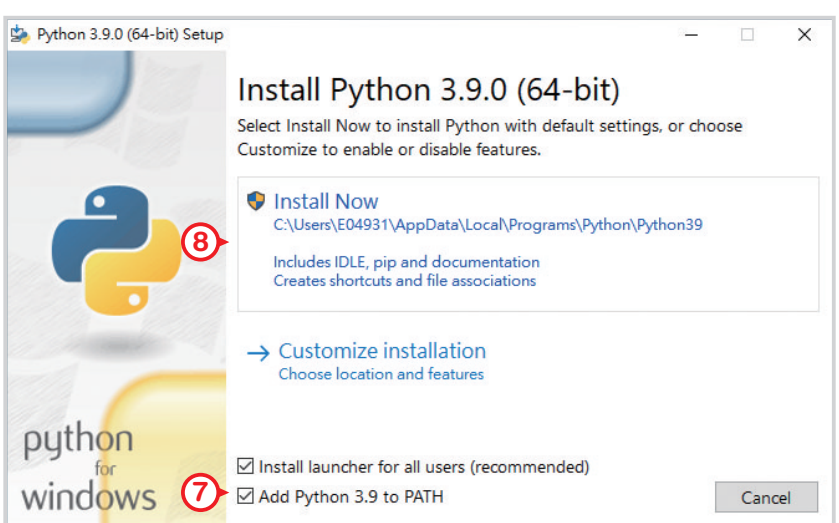
⑥ 在彈出選單中點選**開啟**，啟動安裝介面。



步驟 3 安裝 Python。

⑦ 執行安裝程式時，請勾選 Add Python 3.9 to PATH，讓安裝程式自動設定 PATH 環境變數，之後使用命令提示字元時，不管在哪一個資料夾底下都可以用 Python 指令執行程式。

⑧ 點選 Install Now，開始安裝 Python。



小知識

版本說明

官網上會持續推出新的 Python 版本，但因 Python 的套件很多，易與最新版本不相容，因此使用哪一種 Python 版本，應根據選用的套件是否支援決定。

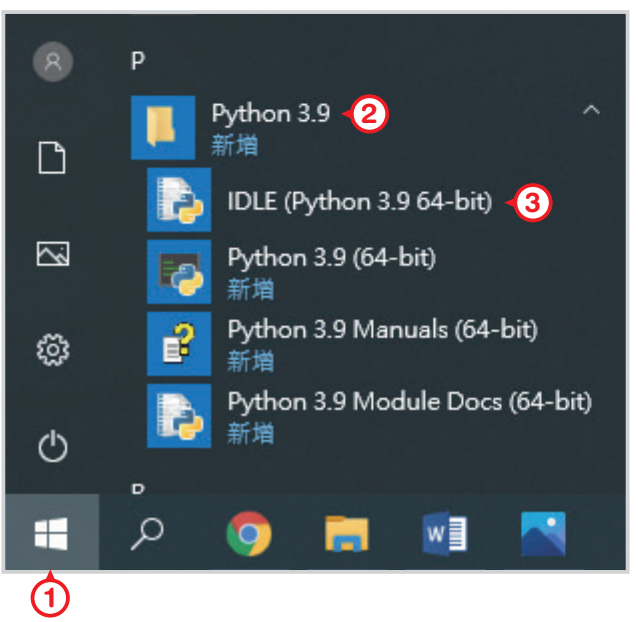
2-1-2 第一個 Python 程式

步驟 1 安裝 Python 後，使用內建的 IDLE 編輯器，練習 Python 程式的使用方式。

① 點選**開始功能表**。

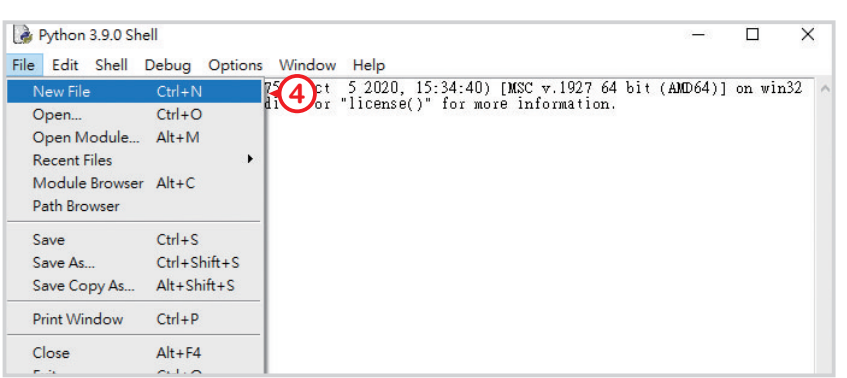
② 點擊 **Python 3.9** 的資料夾。

③ 開啟 **IDLE（Python 3.9 64-bit）**。

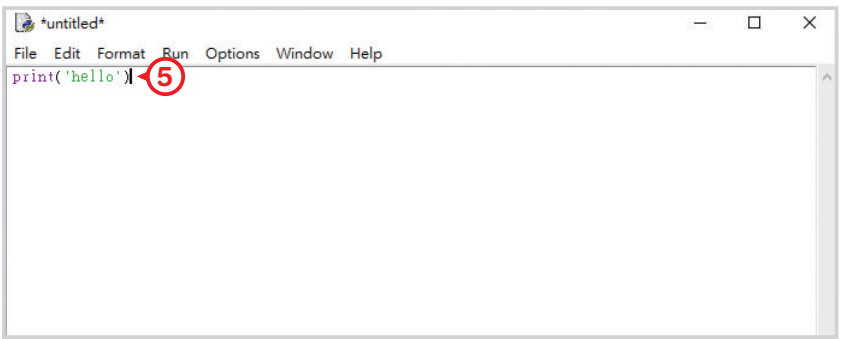


步驟 2 建立第一個程式。

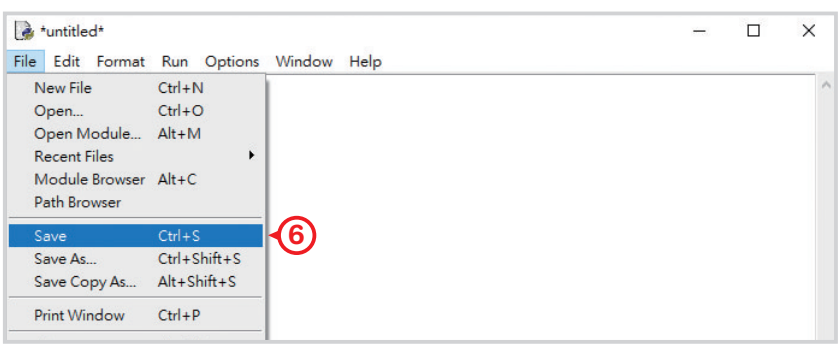
④ 點選 **File** 列表中的 **New File**，開啟新的檔案。



⑤ 在**編輯器**中，輸入程式碼。

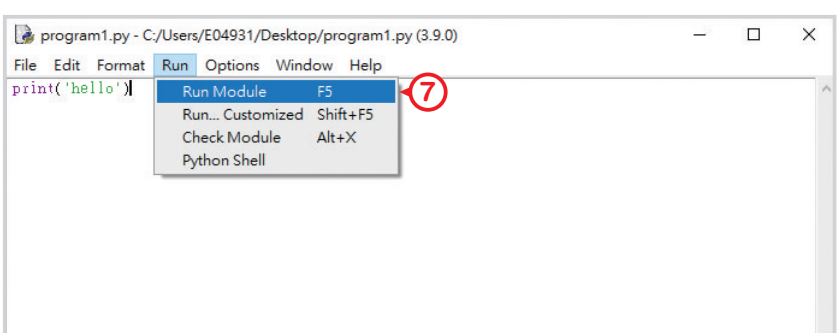


⑥ 在執行前，先點選 **Save** 儲存檔案。

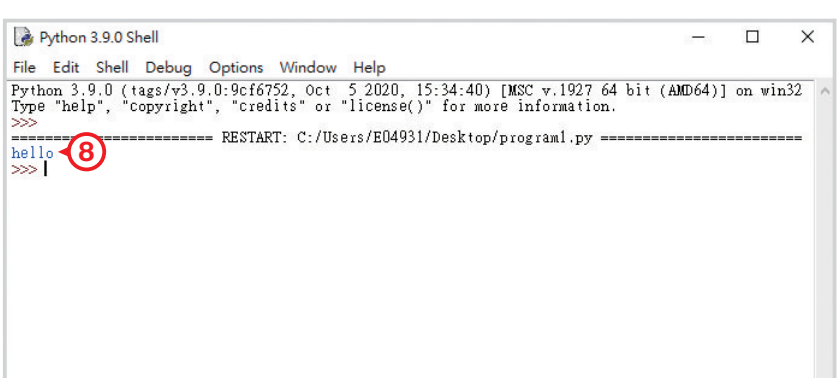


Python 程式的副檔名通常是 .py。

⑦ 點選 **Run** 列表中的 **Run Module** 執行程式。



⑧ 結果會呈現在**交談式介面**裡。



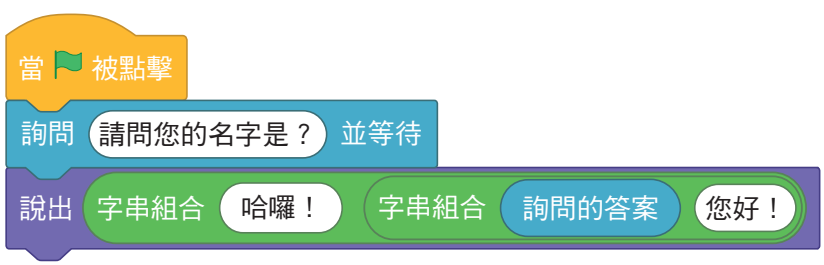
2-2 Python 程式設計－計算篇

編寫過第一個程式以後，相信同學們對 Python 及文字式的程式語言有了基本的認識，接下來讓我們用「哈囉」的程式為例，認識輸入指令（input）與輸出指令（print）的概念，真正地進入文字式程式語言的世界吧！

範　例－哈囉程式

請設計一個程式，讓使用者輸入名字後，電腦會將名字呈現在畫面上與使用者打招呼。

Scratch 積木



概念 input 指令

input 指令是讓使用者由鍵盤輸入資料，通常會用變數儲存該資料，語法如下：

```
變數 = input(['提示字串'])
```

程式中會出現各種符號，以下將分別說明各種符號的作用：

1. 等號 = 的作用：將等號右邊的數值儲存到等號左邊的變數。
2. 小括號 (　) 的作用：放入想要儲存的內容。
3. 中括號 [　] 的作用：[　] 中的內容可以擴充傳入的資料，也可以省略。
4. 單引號 '　' 的作用：'　' 中的內容會被程式視為字串，通常會在字串前後成對出現，字串的表示也可用雙引號 "　"。

中、英文對照

函數名稱 / 指令		變數 / 串列名稱		參數名稱	
input	輸入	name	名字	sep（separate）	分隔
print	輸出			end	結束

Python 程式碼

```
1 name = input('請問您的名字是？')
2 print('哈囉!', name, '您好!')
```

範例程式說明

1. 第 1 行：藉由 input 指令要求使用者輸入名字後，把名字存入到變數 name。
2. 第 2 行：藉由 print 指令，呈現出使用者輸入的名字，並打招呼。

概念 print 指令

print 指令為輸出資料，可以一次輸出多個項目，項目之間以「,」隔開，語法如下：

```
print(項目1 [, 項目2, ..., sep=分隔字元, end=結束字元])
```

print 指令中出現了 sep、end，分別說明如下：

1. sep 分隔字元：如果有多個項目，項目間使用分隔字元區隔，預設為一個空白字元。
2. end 結束字元：輸出完成時自動加入的字元，預設為一個換行字元。

學習完 input 與 print 的指令之後，接下來我們用「求三數之和」的程式為例，學習變數與資料型態、資料型態轉換與算術運算符號的概念吧！

範　例－求三數之和程式

請設計一個程式，讓使用者輸入三個數字後，再呈現三個數字相加的和。

Scratch 積木

中、英文對照

函數名稱 / 指令		函數名稱 / 指令		變數 / 串列名稱	
int（integer）	整數	bool（boolean）	布林值	score	成績
float	浮點數	str（string）	字串	average	平均值
				flag	旗子
				number	數字

Python 程式碼

```
1  x = int(input('請輸入數字 x:'))
2  y = int(input('請輸入數字 y:'))
3  z = int(input('請輸入數字 z:'))
4  print(x + y + z)
```

範例程式說明

1. 第 1 行：將使用者輸入的第一個數字存到變數 x。
2. 第 2 行：將使用者輸入的第二個數字存到變數 y。
3. 第 3 行：將使用者輸入的第三個數字存到變數 z。
4. 第 4 行：呈現出使用者輸入的三個數字相加的和。

使用 input( ) 指令從鍵盤輸入數字，其數字是以字串的資料型態來儲存，因此必須先用 int( ) 指令強制轉換為整數，才能使用加號進行運算，否則就會變成是字串相加，而不是數值相加。

概念 變數與資料型態

在程式中使用變數來儲存資料，語法如下：

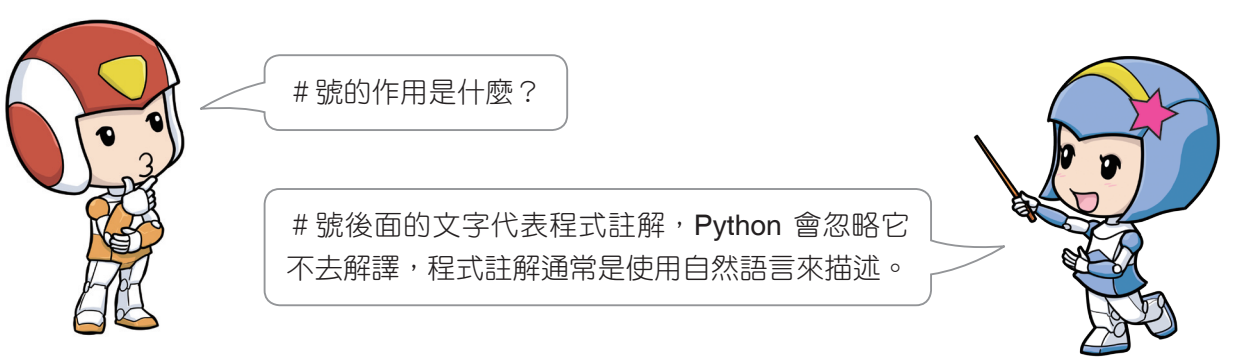
```
變數名稱 = 變數值
```

Python 會根據變數值設定適合的資料型態，對於不同性質的資料，須決定使用哪種資料型態來儲存，這也會影響到電腦要配置多大的記憶體空間給這個變數使用。

Pyhon 對應的資料型態	
常用的變數資料型態	變數值（範例）
整數（int）	80
浮點數（float）	80.8
布林值（bool）	True
字串（str）	John

例如

```
score = 80        # 變數值是 80，自動設定變數為整數。
average = 80.8    # 變數值是 80.8，自動設定變數為浮點數。
flag = True       # 變數值是 True，自動設定變數為布林值。
name = 'John'     # 變數值是 John，自動設定變數為字串。
```



概念 資料型態轉換

資料型態相同的變數才能進行運算。對於某些資料型態，Python 能夠進行簡單的自動轉換。如果是整數與浮點數運算，系統會先將整數轉換成浮點數再運算，而最後的運算結果仍為浮點數。

例如

```
number = 8 + 6.3  # 結果為 14.3，浮點數。
```

如果系統無法自動進行資料型態轉換，就需使用下列的強制資料轉換指令。

指令	使用時機
int( )	強制轉換為整數。
float( )	強制轉換為浮點數。
bool( )	強制轉換為布林值。
str( )	強制轉換為字串。

舉例來說，如果對整數與字串做加法運算，會產生錯誤。

例如

```
number1 = 56 + '23'       # 錯誤，無法進行運算。
number2 = 56 + int('23')  # 正確，結果為 79。
```

概念 算術運算符號

Python 中，可使用的算術運算符號有 6 種，如下表所示。

算術運算符號	意義	範例	運算結果
+	加法	8+5	13
-	減法	8-5	3
*	乘法	8*5	40
/	除法	8/5	1.6
%	除法取餘數	8%5	3
//	除法取商數	8//5	1

例如

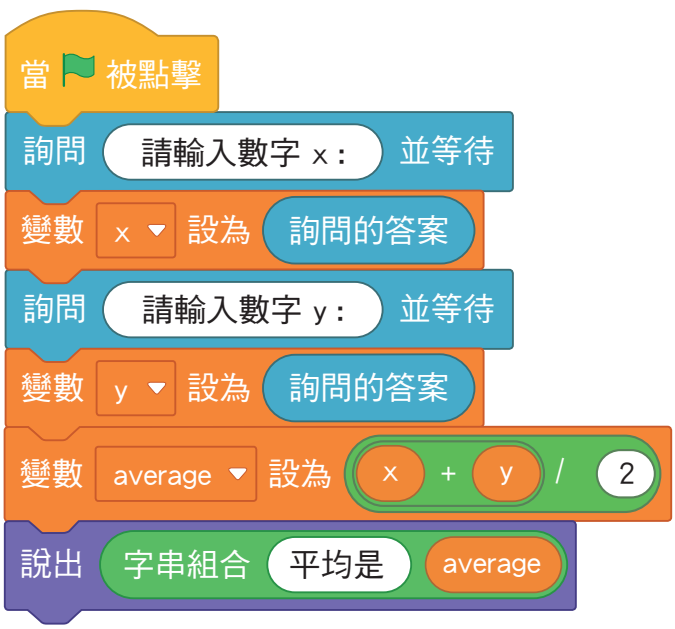
```
print(15%7)   # 結果為 1。
print(15//7)  # 結果為 2。
```

學習完資料型態轉換的概念之後，接下來我們用「求平均數」的程式為例，學習數字與字串運算的概念吧！

範　例－求平均數程式

請設計一個程式，讓使用者輸入兩個數字後，再呈現兩個數字的平均值。

Scratch 積木



概念　數字與字串間的運算

加號不僅可用於數值運算，也可用於字串組合，但同學們必須謹記「只有資料型態相同的變數才能進行運算」。因此，使用時需注意其所應用的資料型態，如下：

例如

```
23 + 64              # 正確，結果為 87。
23 + '64'            # 錯誤，無法進行運算。
'hello' + 34         # 錯誤，無法進行運算。
'hello' + '34'       # 正確，結果為 hello34。
'hello' + str(34)    # 正確，結果為 hello34。
```

Python 程式碼

```
1  x = int(input('請輸入數字 x:'))
2  y = int(input('請輸入數字 y:'))
3  average = (x + y) / 2
4  print('平均是' + str(average))
```

範例程式說明

1. 第 1 行：將使用者輸入的第一個數字存到變數 x。
2. 第 2 行：將使用者輸入的第二個數字存到變數 y。
3. 第 3 行：計算平均值。
4. 第 4 行：因為 average 是浮點數，所以須強制轉換成字串，才能進行字串相加，並將結果輸出。

學習完數字與字串運算的概念之後，接下來我們用「計算學期成績」的程式為例，學習關係運算符號與選擇結構的概念吧！

範　例－計算學期成績程式

請設計一個程式，讓使用者輸入各項成績後，再將各項成績轉換為學期成績，並判斷學期成績是否及格？（其中，作業成績占 40%，測驗成績占 40%，平時成績占 20%，學期成績 60 分為及格分數。）

Scratch 積木

中、英文對照

保留字		保留字		變數 / 串列名稱	
if	如果	False	假	grade	成績
else	否則	True	真		
		elif（else if）	否則		

Python 程式碼

```
1  score1 = int(input('請輸入作業成績:'))
2  score2 = int(input('請輸入測驗成績:'))
3  score3 = int(input('請輸入平時成積:'))
4  grade = score1*0.4 + score2*0.4 + score3*0.2
5  print('學期成績是' + str(grade))
6  if grade < 60:
7      print('不及格')
8  else:
9      print('及格')
```

範例程式說明

1. 第 1 行：將使用者輸入的第一個數字存到變數 score1。
2. 第 2 行：將使用者輸入的第二個數字存到變數 score2。
3. 第 3 行：將使用者輸入的第三個數字存到變數 score3。
4. 第 4 行：計算學期成績。
5. 第 5 行：因為 grade 是浮點數，所以須強制轉換成字串，才能進行字串相加，並將結果輸出。
6. 第 6 ～ 9 行：使用雙向選擇結構判斷，如果 grade 小於 60，呈現出不及格，否則呈現出及格。

保留字是文字式程式語言中，已經具有特定意義的英文單字，不可用來作為使用者自訂的變數名稱。

概念 關係運算符號

Python 中，可使用的關係運算符號有 6 種，如下表所示。

關係運算符號	意義	範例	運算結果
==	相等	8==5	False
		2+6==3+5	True
!=	不相等	8!=5	True
		2+6!=3+5	False
>	大於	8>5	True
		2>6	False
<	小於	8<5	False
		2<6	True
>=	大於或等於	8>=5	True
		2+3>=5	True
		2>=6	False
<=	小於或等於	8<=5	False
		2+3<=5	True
		2<=5	True

例如

```
print(8==5)   # 結果為 False。
print(8!=5)   # 結果為 True。
```

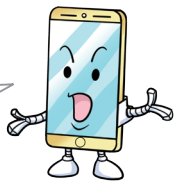
概念 單向選擇結構

在日常生活中，我們經常會遇到要作決策的情況，例如：如果下雨了，就帶把傘出門。程式的執行也是類似的情況，根據條件式運算結果，執行不同的程式區塊。最簡單流程控制是單向選擇結構，語法如下：

```
if 條件式:
    程式區塊
```

if 敘述後面接著條件式，條件式的運算結果是布林值，當條件式為 True 時，就會執行程式區塊的敘述；當條件式為 False 時，不會執行程式區塊。在條件式之後必須有冒號，底下的程式區塊必須縮排。

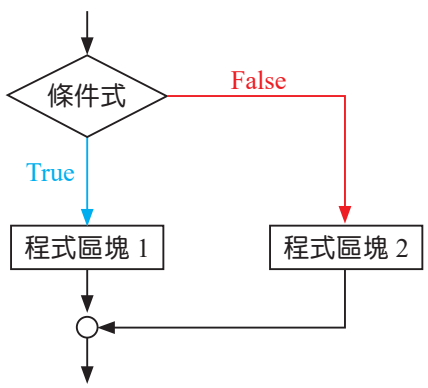
在 Python 中縮排程式碼，可以使用 1 個 Tab 鍵或 4 個空白字元，但須注意同一個程式只能使用其中一種方式，不可同時使用 Tab 鍵及空白字元。



概念 雙向選擇結構

如果我們希望遇到情況為 True 和 False 時，分別做不同的事，就需要使用雙向選擇結構，語法如下：

```
if 條件式:
    程式區塊 1
else:
    程式區塊 2
```

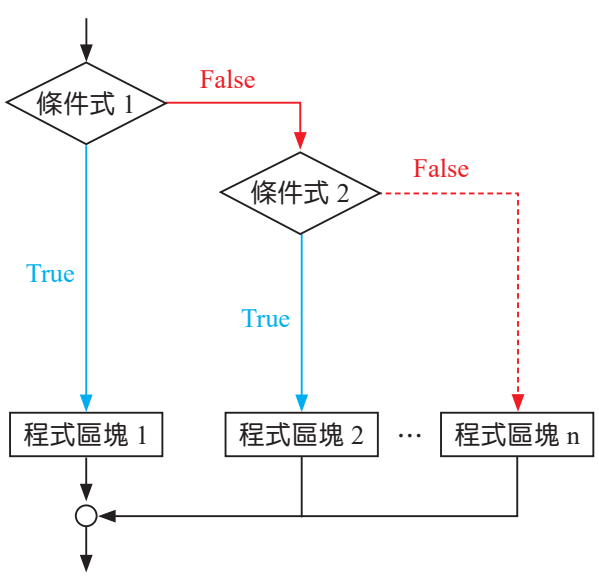


當條件式為 True 時，就執行程式區塊 1，否則就執行 else 裡面的程式區塊 2。

概念 多向選擇結構

如果發生的情況不只兩種，就可以透過使用 elif 指令，做更進一步的選擇，語法如下：

```
if 條件式 1:
    程式區塊 1
elif 條件式 2:
    程式區塊 2
elif 條件式 3:
    ......
else:
    程式區塊 n
```



當其中一個條件式為 True 時，就執行相對應的程式區塊，並且當所有條件式皆為 False 時，就執行 else 裡面的程式區塊。

學習完關係運算符號與選擇結構的概念之後，接下來我們用「累加計算」的程式為例，學習串列、range 函式與 for 迴圈的概念吧！

範　例－累加計算程式

請設計一個程式，讓使用者輸入數字 n 後，再計算 1 ＋ 2 ＋ 3 ＋…＋ n 的值。

Scratch 積木

中、英文對照

變數 / 串列名稱		保留字		函數名稱 / 指令	
sum	總和	for	對於	range	範圍
		in	在…之內	list	串列

Python 程式碼

```
1  n = int(input('請輸入數字 n:'))
2  sum = 0
3  for i in range(1, n+1):
4      sum = sum + i
5  print('1+2+...+' + str(n) + '=' + str(sum))
```

範例程式說明

1. 第 1 行：將使用者輸入的數字存到變數 n。
2. 第 2 行：將變數 sum 設為 0，用以儲存累加的總和。
3. 第 3 ～ 4 行：將 1 到 n 串列中的數值代入到變數 i ，並執行計次式迴圈，過程中每次都將變數 i 加到累加的總和。
4. 第 5 行：呈現出總和。

概念 串列

程式中的資料通常使用變數來儲存，但是對於大量的資料則可以用串列簡化變數的需求量。每一個串列擁有一個識別名稱，串列中的每一個資料稱為元素，每一個元素相當於一個變數，要存取串列中特定元素，是以元素在串列中的位置做為索引值，語法如下：

```
串列名稱 = [元素 1, 元素 2, ...]
```

串列中的元素資料型態可以相同，也可以不相同。要注意的是 Python 的串列索引值從 0 開始，第一個元素的索引值是 0，第二個元素的索引值是 1，第三個元素的索引值是 2，以此類推。此外，索引值也可以是負數，代表從串列最後算起第幾個元素，例如：最後一個元素的索引值即為－1。

例如

```
list1 = [1, 2, 3, 4, 5]              # 元素皆為整數
list2 = ['John', 'Marry', 'Tom']     # 元素皆為字串
list3 = ['Hellen', 18, True]         # 包含不同資料型態元素
print(list1[0])                      # 1
print(list2[1])                      # Marry
print(list3[-1])                     # True
```

概念 range 函式

計次式迴圈中經常使用 range 函式來建立整數循序串列，語法如下：

```
range([起始值,] 終止值 [,累加值] )
```

產生的串列是由起始值開始（預設值為 0），每次遞增累加值（預設值為 1），直到終止值的前一項。舉例來說，range(1 , 5) 會產生一個從 1 開始，小於 5 （不包含 5）的整數串列，因此回傳的串列是 [1 , 2 , 3 , 4]，而若要產生從 1 開始，20 結束的整數串列，則要寫成 range(1 , 21)。

例如

```
list1 = range(5)      # 執行後，list1 的內容是 [0, 1, 2, 3, 4]。
list2 = range(1, 5)     # 執行後，list2 的內容是 [1, 2, 3, 4]。
list3 = range(1, 5, 2)   # 執行後，list3 的內容是 [1, 3]。
list4 = range(5, 1, -1)  # 執行後，list4 的內容是 [5, 4, 3, 2]。
```

概念 for 迴圈

for 迴圈是一種處理重複步驟的語法，它會從一個串列中逐一取出元素，然後指定給迴圈變數，因為串列中的元素個數，即是重複迴圈中程式碼執行的次數，所以稱為計次式迴圈。

for 和 in 之間的迴圈變數可以自己定義變數名稱，而 in 後面接著一個序列，在這個例子中，我們使用了串列（list），它是序列的其中一種。for 迴圈會依序從序列中取得元素，並且將元素指定給前面自訂的迴圈變數，再執行迴圈裡的內容，直到序列每一元素都被取過為止，語法如下：

```
for 迴圈變數 in 串列：
    程式區塊
```

舉例來說，有一個數字串列 [1 , 2 , 3 , 4 , 5]，將此串列放在 in 的後方，並使用 i 來做為迴圈的變數。在程式區塊部分，為了讓 Python 了解這個區塊是屬於迴圈的內容，所以要向內縮排。程式區塊內可以寫入我們希望程式完成的指令，例如：若想輸出每一個迴圈中的迴圈變數值，就可以寫成 print(i)。

例如

```
for i in [1, 2, 3, 4, 5]:
    print(i)
```

另外，因為這個串列是有規則的數列，所以我們可以用 range 函式來簡化程式碼。

例如

```
for i in range(1, 6):
    print(i)
```

學習完串列、range 函式與 for 迴圈的概念之後，接下來我們用「密碼檢查」的程式為例，學習邏輯運算符號與 while 迴圈的概念吧！

範　例－密碼檢查程式

請設計一個電腦系統的密碼驗證機制，條件如下：

1. 若第一次輸入密碼錯誤後，可再重複嘗試輸入兩次。
2. 若三次密碼都錯誤，跳出使用者帳號被鎖定的訊息。

Scratch 積木

中、英文對照

變數 / 串列名稱		保留字		保留字	
passsword	密碼	while	當	not	非；不是
times	次數	and	而且	or	或

Python 程式碼

```
1  password = '137'
2  times = 1
3  password2 = input('請輸入密碼:')
4  while password != password2 and times < 3:
5      print('密碼錯誤!')
6      times = times + 1
7      password2 = input('請輸入密碼:')
8  if password == password2:
9      print('歡迎使用本系統')
10 else:
11     print('輸入密碼錯誤 3 次，帳號已被鎖定')
```

範例程式說明

1. 第 1 行：將變數 password 設為 137，用來當做預設密碼。
2. 第 2 行：將變數 times 設為 1，用來當做輸入的次數。
3. 第 3 行：要求使用者輸入密碼，並存到變數 password2。
4. 第 4 ～ 7 行：
   (1) 使用條件式迴圈判斷，當預設密碼 password 與使用者輸入的密碼 password2 不同，且輸入次數仍小於三次時，顯示密碼錯誤，此時輸入的次數增加 1，並再次要求使用者輸入密碼。
   (2) 當預設密碼與使用者輸入的密碼相符，或預設密碼與使用者輸入的密碼不符的次數達到第三次，則會跳出迴圈，執行雙向選擇結構。
5. 第 8 ～ 11 行：使用雙向選擇結構判斷，如果預設密碼與使用者輸入的密碼相同，呈現歡迎使用本系統，否則呈現輸入密碼錯誤 3 次，帳號已被鎖定。

概念 邏輯運算符號

Python 中，可使用的邏輯運算符號有 3 種，如下表所示。

邏輯運算符號	意義	範例	運算結果
not（反向）	傳回與運算結果相反的值。	not (2<3)	False
		not (3>5)	True
and（且）	左右兩邊同時成立，運算結果才會成立。	(2<3) and (5<9)	True
		(2<3) and (5>9)	False
		(2>3) and (5<9)	False
		(2>3) and (5>9)	False
or（或）	左右兩邊任一個成立，運算結果就會成立。	(2<3) or (5<9)	True
		(2<3) or (5>9)	True
		(2>3) or (5<9)	True
		(2>3) or (5>9)	False

例如

```
print(not(5<8))          # 結果為 False。
print((5>2)and(6<8))     # 結果為 True。
print((5>2)or(6>8))      # 結果為 True。
```

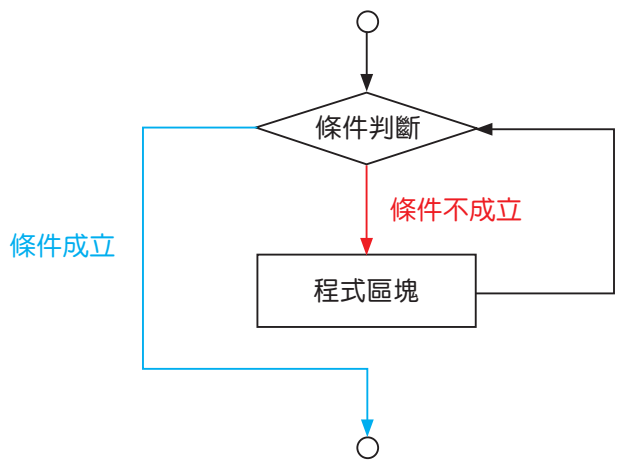
概念 while 迴圈

當我們不知道要重複執行幾次，而只知道繼續重複執行的條件，這時候就要使用條件式迴圈。Python 的條件式迴圈使用 while，後面接的是條件式。若條件式成立，則執行迴圈中的程式區塊，若條件式不成立，則跳出迴圈，語法如下：

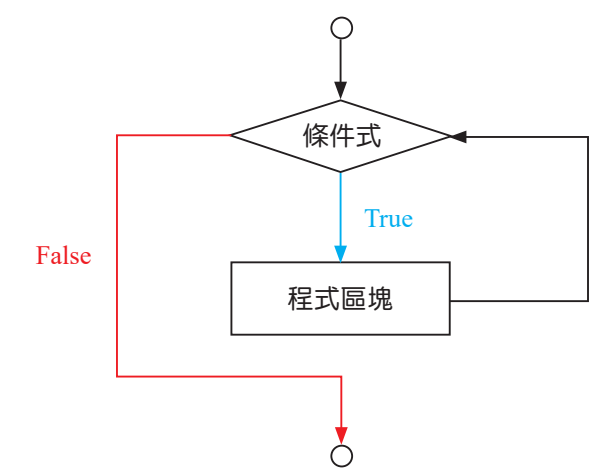
```
while 條件式:
    程式區塊
```

Scratch 條件式迴圈積木是重複執行直到條件成立為止，所以這時侯的條件式是終止狀況，不同的是 Python 的 while 迴圈，其條件式並不是終止狀況，而是符合條件式的情況下執行迴圈內的程式區塊。我們可以從下方表格中的流程圖看出其中的差異。

Scratch（條件式迴圈）



Python（while 迴圈）



學習完邏輯運算符號與 while 迴圈的概念之後，接下來我們用「任意數的所有因數」的程式為例，學習串列的進階概念吧！

範　例－任意數的所有因數程式

請設計一個程式，讓使用者輸入一個數字後，再找出該數字的所有因數。

Scratch 積木

中、英文對照

變數 / 串列名稱		函式名稱 / 指令		函式名稱 / 指令	
factor	因數	append	新增	insert	插入
		remove	移除		

Python 程式碼

```
1  n = int(input('請輸入數字 n:'))
2  factors = []
3  for i in range(1, n+1):
4      if n % i == 0:
5          factors.append(i)
6  print(factors)
```

範例程式說明

1. 第 1 行：將使用者輸入的數字存到變數 n。
2. 第 2 行：定義一個名稱為 factors，且沒有任何元素的空串列。
3. 第 3 行：將 1 到 n 串列中的數值代入到變數 i ，並執行計次式迴圈，過程中每次都執行相對應的動作。
4. 第 4 ～ 5 行：因為這個程式碼目的是找出所有因數，而因數就是可以整除的數字，所以 n 除以 i 的餘數為 0 時，將 i 的值新增到串列 factors。
5. 第 6 行：呈現出所有因數。

概念 串列進階用法

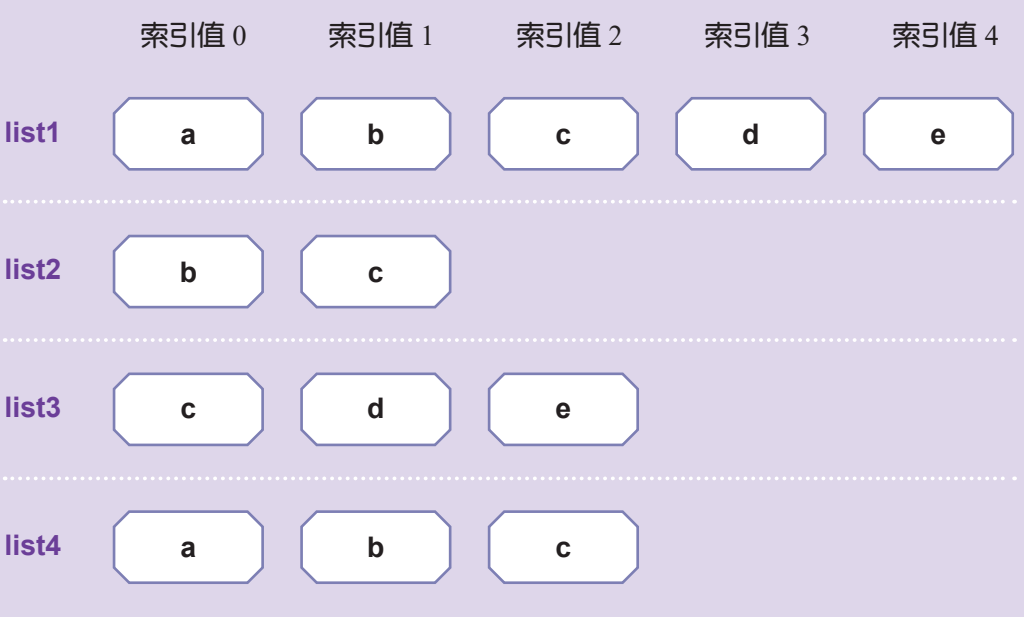
在前面我們已經認識了串列的基本概念，現在要來認識串列的進階用法。對於一個串列，可以使用索引值去取用串列中的元素，索引值可以是一個數字，一次取用串列中的一個元素。不僅如此，串列的中括號裡面也可以是一個數字範圍，一次取用串列中多個元素，形成另外一個子串列。

例如

```
list1 = ['a', 'b', 'c', 'd', 'e']
list2 = list1[1:3]
# 從索引值 1 的位置開始取用，直到不包含索引值 3 的位置，將之間的元素
  取出來。執行後，list2 的內容是 ['b', 'c']。

list3 = list1[2:]
# 從索引值 2 的位置開始取用，直到最後一個元素位置，將之間的元素取出
  來。執行後，list3 的內容是 ['c', 'd', 'e']。

list4 = list1[:3]
# 從索引值 0 的位置開始取用，直到不包含索引值 3 的位置，將之間元素取
  出來。執行後，list4 的內容是 ['a', 'b', 'c']。
```



此外，串列的內容可以新增元素、移除元素，或是將元素插入到指定的位置上。

1. 在串列 list1 中，新增元素 c，可以寫成 list1.append('c')。

例如

```
list1 = ['a','b']
list1.append('c')    # 執行後，list1 的內容是 ['a', 'b', 'c']。
```

2. 在串列 list2 中，移除元素 b，可以寫成 list2.remove('b')。

例如

```
list2 = ['a','b']
list2.remove('b')    # 執行後，list2 的內容是 ['a']。
```

3. 在串列 list3 中，插入元素 b 到第 0 項，可以寫成 list3.insert(0, 'b')。

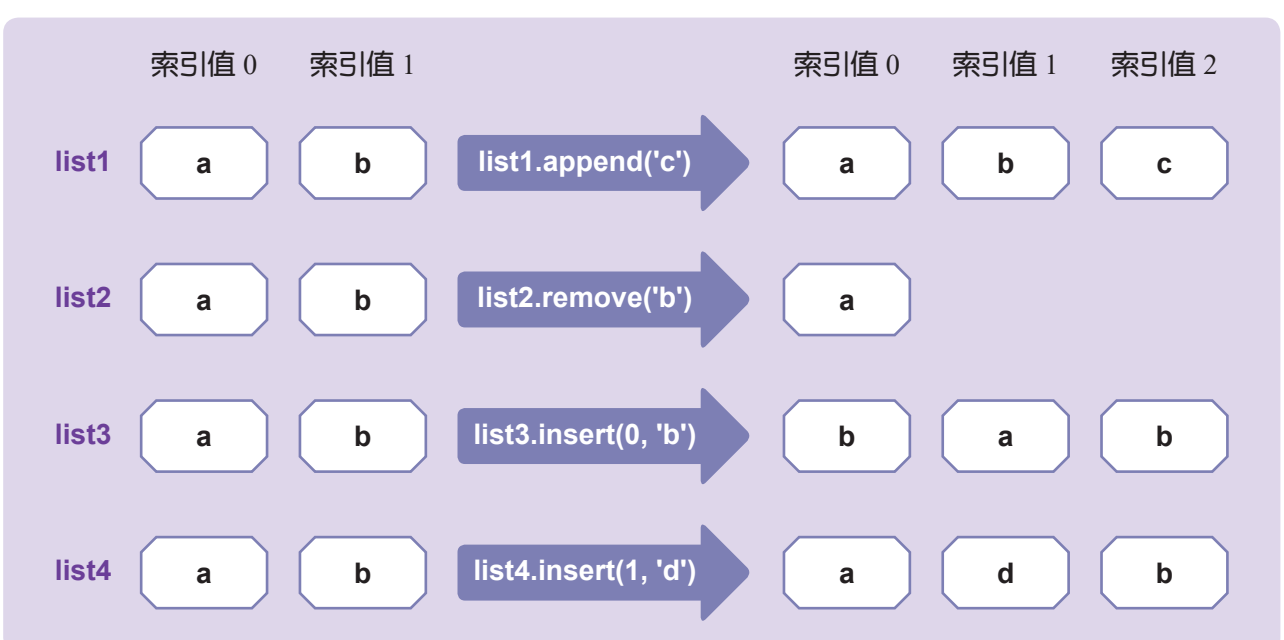
例如

```
list3 = ['a','b']
list3.insert(0, 'b')  # 執行後，list3 的內容是 ['b', 'a', 'b']。
```

4. 在串列 list4 中，插入元素 d 到第 1 項，可以寫成 list4.insert(1, 'd')

例如

```
list4 = ['a','b']
list4.insert(1, 'd')  # 執行後，list4 的內容是 ['a', 'd', 'b']。
```

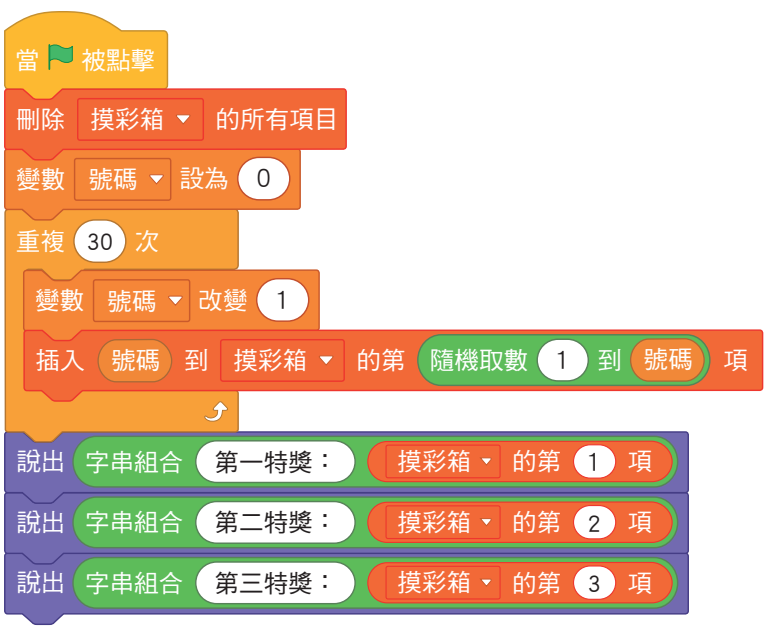


學完串列的進階用法後，接著我們用「抽獎」的程式為例，學習亂數的概念吧！

範　例－抽獎程式

有三十人參加摸獎活動，分別有編號 1 ～ 30 的摸獎券，透過電腦進行抽獎，從中抽出三位幸運兒頒發獎品。

Scratch 積木



概念　亂數

Python 有各式各樣的模組，例如：匯入亂數模組可以寫成 import random，之後就能使用亂數模組的多種指令。但如果只是要用亂數模組裡面的取整數亂數這一個功能而已，使用 from random import randint 的語法就可以，不需要使用 import random 把整個模組都匯進來。其中，randint(a, b) 就像 Scratch 中的隨機取數，會隨機挑選出一個大於等於 a，且小於等於 b 的整數。

例如

```
from random import randint   # 從 random 模組匯入 randint 函式。
number = randint(1, 100)     # 產生 1～100 整數亂數，將它存到變數
                               number 中。
print(number)                # 輸出變數 number 儲存的整數。
```

中、英文對照

保留字		變數 / 串列名稱		函式名稱 / 指令	
from	從	position	位置	random	隨機
import	匯入	box	箱子	randint（random integer）	隨機整數

Python 程式碼

```
1  from random import randint
2  n = 30
3  box = []
4  for i in range(1, n+1):
5      position = randint(0, i-1)
6      box.insert(position, i)
7  print(box)
8  print('第一特獎：' + str(box[0]))
9  print('第二特獎：' + str(box[1]))
10 print('第三特獎：' + str(box[2]))
```

範例程式說明

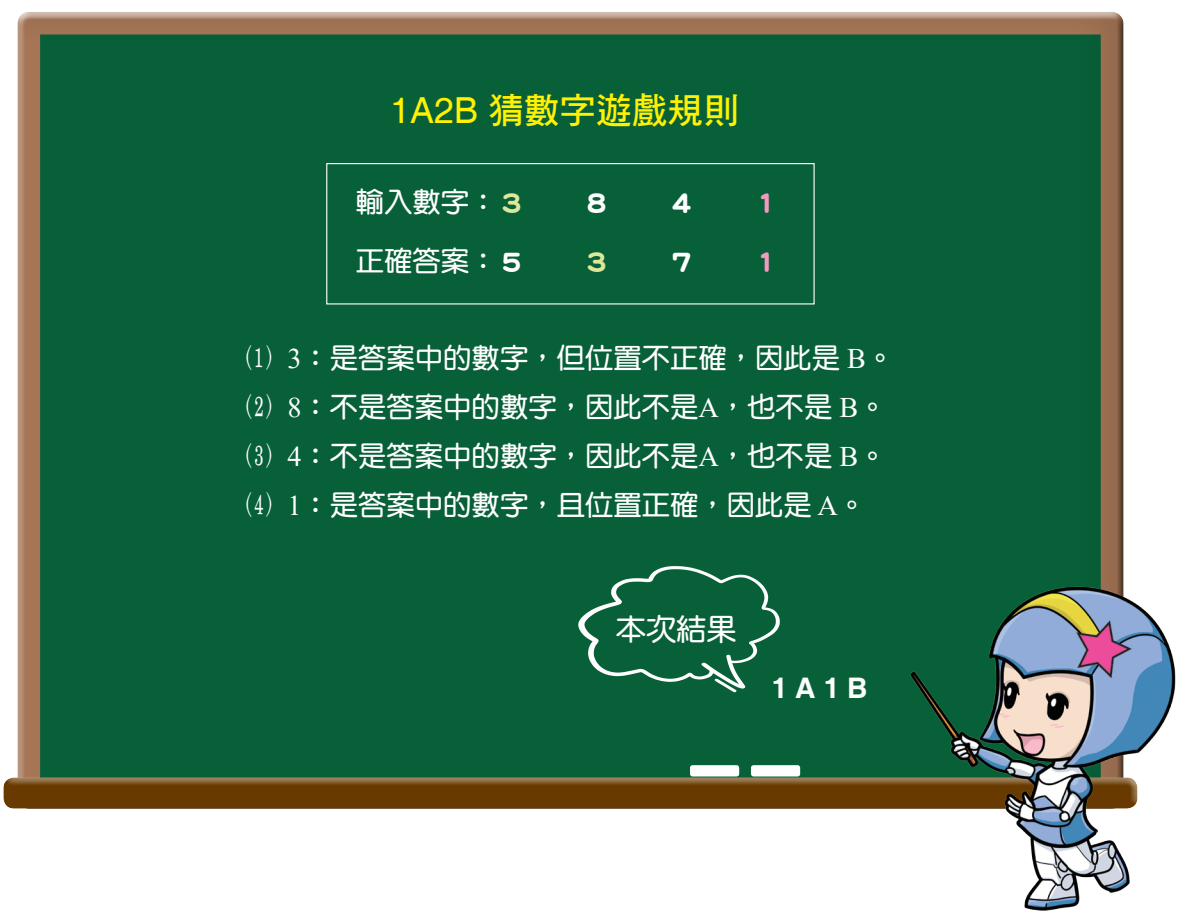
1. 第 1 行：用 from random import randint 的語法，取用整數亂數的功能。
2. 第 2 行：設定變數 n 為 30。
3. 第 3 行：定義一個名稱為 box，且沒有任何元素的空串列。
4. 第 4 ～ 6 行：將 1 到 n 串列中的數值代入到變數 i，並執行計次式迴圈，過程中隨機將數字插入串列索引值 0 ～ 29 的位置。
5. 第 7 行：呈現 box 的所有串列。
6. 第 8 ～ 10 行：呈現第一、第二、第三特獎的結果。

2-3 Python 程式設計－專題

經過前面的課程，相信同學們對 Python 已經更加熟悉，接下來讓我們使用學習過的概念來完成一個程式。

範　例－1A2B 猜數字遊戲

請設計一個程式，先隨機產生 4 個 0 ～ 9 之間，且不重複的數字當正確答案，接著使用者在畫面上輸入 4 個不重複的數字後，程式會將輸入的數字與答案進行比對，再用「幾 A 幾 B」的形式告訴使用者正確與否。其中，A 代表數字正確，且位置正確；B 代表數字正確，但位置錯誤。如果在 8 次機會中，使用者輸入的數字完全正確，則出現「您答對了，正確答案是…」；沒猜到正確數字，則出現「作答已達 8 次，遊戲結束，正確數字是…」。



問題分析

我們可以將這個程式拆解為幾個部分如下：

① 如何讓程式隨機產生 4 個 0 ～ 9 之間，且不重複的數字？

② 如何將答案串列變為字串？

③ 使用者輸入數字後，如何判斷結果？

④ 如何讓程式重複執行，直到使用者輸入的數字正確，或輸入錯誤的次數已達 8 次？

⑤ 如何決定要輸出何種結果？

解題步驟

問題拆解 1 如何讓程式隨機產生 4 個 0 ～ 9 之間，且不重複的數字？

步驟 1 產生一個 0 ～ 9 的數字串列及儲存答案的空串列。

數字串列 0 1 2 3 4 5 6 7 8 9
答案串列

步驟 2 以下動作重複進行四次。

① 從 0～9 的串列隨機取一個數字加到答案的串列中。

② 刪除數字串列中，已經被取出的數字。

數字串列 0 1 2 3 4 5 6 8 9
答案串列 7

↓

數字串列 0 1 2 3 4 5 6 9
答案串列 7 8

↓

數字串列 0 1 2 3 4 5 6
答案串列 7 8 9

↓

數字串列 0 1 2 3 4 5
答案串列 7 8 9 6

問題拆解 2 **如何將答案串列變為字串？**

可以先創造一個空的字串，再將答案串列中的數字，一一加入空字串中，即可將答案串列變為字串。

問題拆解 3 **使用者輸入數字後，如何判斷結果？**

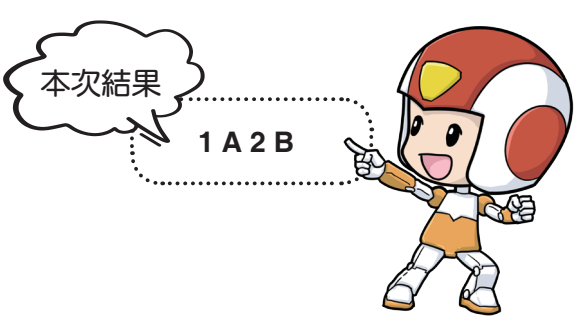
步驟 3 **取得使用者猜測的數字。**

步驟 4 **將輸入的數字串列與答案一一拿出來比對，並重複完以下動作後，再判斷結果。**

③ 若位置相同，且數字相同，表示 A。

④ 若位置不同，但數字相同，表示 B。

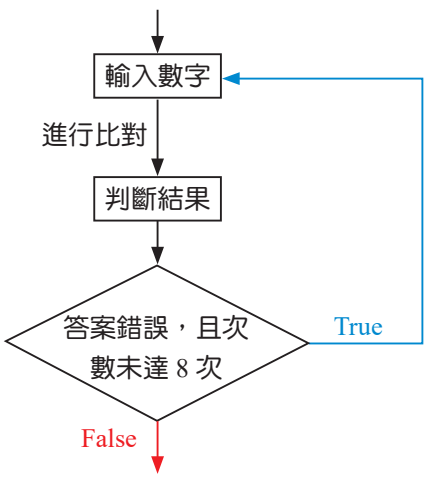
輸入數字 1 6 9 7 答案串列 7 8 9 6	A 的計數器：0 B 的計數器：0
輸入數字 1 6 9 7 答案串列 7 8 9 6	A 的計數器：0 B 的計數器：0＋1＝1
輸入數字 1 6 9 7 答案串列 7 8 9 6	A 的計數器：0＋1＝1 B 的計數器：1
輸入數字 1 6 9 7 答案串列 7 8 9 6	A 的計數器：1 B 的計數器：1＋1＝2



問題拆解 4 如何讓程式重複執行，直到使用者輸入的數字正確，或輸入錯誤的次數已達 8 次？

步驟 5 當次數未達 8 次且答案錯誤時，重複執行 問題拆解 3 的動作。

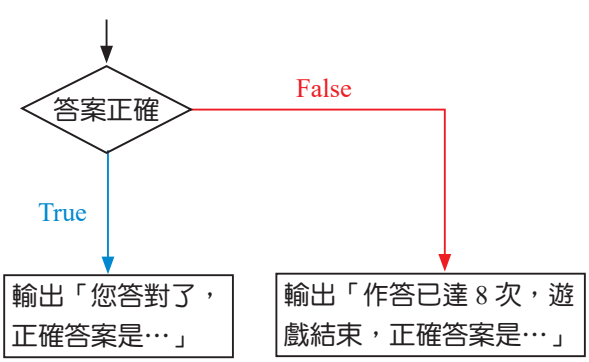
步驟 6 當錯誤次數已達 8 次或答案正確時，跳出迴圈。



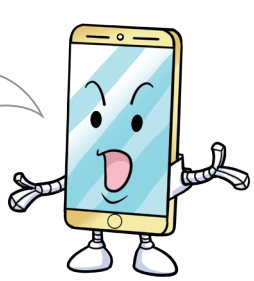
問題拆解 5 如何決定要輸出何種結果？

步驟 7 當答案正確時，輸出「您答對了，正確答案是…」。

步驟 8 當答案錯誤次數已達到 8 次時，輸出「作答已達 8 次，遊戲結束，正確答案是…」。



最後，請同學利用 問題拆解 1 ～ 問題拆解 5 ，完成 1A2B 猜數字遊戲的程式。



重點回顧

本章選用 Python 來學習文字式程式語言，並導入學習過的 Scratch 範例，讓同學能夠將問題從積木式語言銜接到文字式語言。Python 是一種廣泛使用且功能強大的通用型程式語言，很適合做為第一個用來學習的文字式程式語言。下表為本章學習過的概念、指令及語法：

概念		語法
輸入		變數 = input(['提示字串']) print(項目 1 [, 項目 2, ..., sep=分隔字元, end=結束字元])
輸出		
選擇結構	單向選擇結構	if 條件式: 　　程式區塊

	雙向選擇結構	if 條件式:
		　　程式區塊 1
		else:
		　　程式區塊 2

	多向選擇結構	if 條件式 1:
		　　程式區塊 1
		elif 條件式 2:
		　　程式區塊 2
		elif 條件式 3:
		　　.....
		else:
		　　程式區塊 n

重複結構	產生有規律串列	range([起始值,] 終止值 [,累加值])
	計次式迴圈	for 迴圈變數 in 串列： 　　程式區塊
	條件式迴圈	while 條件式: 　　程式區塊

資料型態	轉換指令	使用時機
整數（int）	int( )	強制轉換為整數。
浮點數（float）	float( )	強制轉換為浮點數。
布林值（bool）	bool( )	強制轉換為布林值。
字串（str）	str( )	強制轉換為字串。

符號類型	符號	意義	符號	意義
算術運算符號	+	加法	/	除法
	-	減法	%	除法取餘數
	*	乘法	//	除法取商數
關係運算符號	==	相等	<	小於
	!=	不相等	>=	大於或等於
	>	大於	<=	小於或等於
邏輯運算符號	not	傳回與運算結果相反的值。		
	and	左右兩邊同時成立，運算結果才會成立。		
	or	左右兩邊任一個成立，運算結果就會成立。		

其他概念	語法
變數	變數名稱 = 變數值
串列	串列名稱 = [元素 1, 元素 2, ...]
亂數	import random
取整數亂數	from random import randint