

3 資料與變數



挑戰題目: 電影魔球的相關資訊

電影魔球預告片

魔球這部電影到底值不值得看？老闆請你把和這部電影相關的資訊整理出來讓他可以做判斷。

[運算思維 (抽象化) => 程式設計 (IPO)] => 程式 (資料處理要分門別類)

```
1 #|echo: true
2 #|eval: false
3 #|includes: false
4 #|
5
6 ## 1. 從IMBD電影資料庫找出魔球電影的相關資訊
7 ## 網址: https://www.imdb.com/title/tt1210166/
8 ## 相關資訊: 電影名稱、放映日期.....
9
10 ## 2. 在Python用變數儲存資訊
11 movie_title = "money ball" # 電影名稱、文字
12 release_year = "2011"      # 放映日期、文字
13 movie_time_min = 133       # 片長、數字
14 movie_rating = 7.6         # 評分、數字
15
16 director = "Bennett Miller" # 導演、文字
17 stars_1 = "Brad Pitt"      # 主要演員、文字
18 stars_2 = "Robin Wright"   # 主要演員、文字
19 stars_3 = "Jonah Hill"     # 主要演員、文字
20
21 like = True # 喜不喜歡      # 按個讚、布林值
22
23 ## 3. 印出魔球電影的相關資訊
24 print(movie_title)          # 螢幕列印
25 print(release_year)
26 print(movie_time_min)
```

無論是日常生活，還是職場專業，我們會遇到許多問題需要處理。

例如某部電影值不值得看？紅襪隊有哪些明星球員、守備位置在哪裡、近來表現如何？

台灣的某某上市公司，近來股價表現如何？這個公司與產業的近況。

少子化的情況嚴重，台灣的出生率情況？各大學的招生狀況，又有幾間學校會倒....。

我們蒐集了一些資料，希望電腦幫我們處理。那麼這些資料得先分門別類，並且儲存在電腦的記憶體中才行！如何把這些資料輸入電腦？電腦儲存的資料怎樣叫出來查看呢？這些資料可以怎樣運用呢？

3-1. 資料類型

學習目標

資料處理需要分門別類，資料的類型不同、輸入的方式不同。

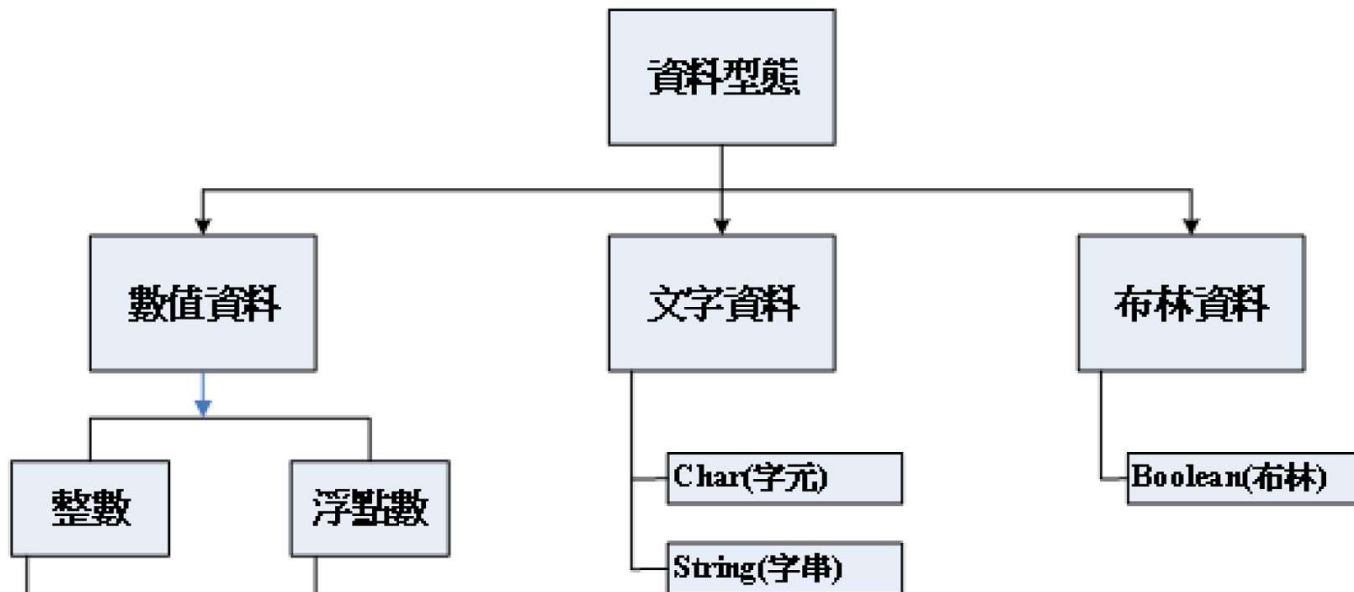
- 資料分成數值numeric（整數integer、浮點數float）、文字character（字串string）與布林Boolean（True與False）三種類型。
- Python資料的輸入語法。

3-1-1 數值、文字和布林

資料分成數值、文字與布林三種類型。

範例：三種常見的資料類型

```
1 ## 常見的資料類型有三種：數值（整數、浮點數）、文字（字串）與布林（邏輯值）。
2 123      # 整數integer
3 123.456  # 浮點數float
4 "Python" # 字串string
5 'Python' # 字串string
6 True     # 邏輯值Logic
7 False    # 邏輯值Logic
```



3-1-2 數值資料

數值：數值資料型態分成整數(integer)與浮點數(float)兩種。

- 整數就是不含小數點的整位數。
 - 程式語言會使用固定大小的空間來儲存一個資料。
- 浮點數類似於小數，是帶有小數點的數字。
 - 浮點數不同於小數。
 - 電腦因為記憶體容量有限，無法表示無窮位數的小數點，如 π 只取有限位數近似值。
 - 許多計算結果，採用二進位與十進位的答案相近，但不完全一樣。如 $0.1+0.1+0.1=?$
 - 浮點數可以用科學記號來表示，如 $a * 10^n$ 。在Python， $7.823e5 = 782300$ 、 $12e-4 = 0.00012$ 。

範例：數字

```
1 ## 數字（電腦輸入數值的方式和日常生活相同）
2 100      # 整數integer
3 100.001  # 浮點數float
```

3-1-3 文字資料

文字：字串(string)是由一連串有順序的字元所組成的序列(sequence)。

- 字元(character)是英文、數字或符號，可以是中文、簡體，或日語、法文等。
- 字串(string)是字元的組合，如電話號碼、姓名、地址、標點符號等。

在Python中，我們使用單引號或雙引號來建立文字字串 (str)。

例：123和'123'(或"123")不同，一個是數值、一個是文字

通常使用單引號或者雙引號不會有任何分別。

- 要成雙成對的出現
 - 單引號開頭，需要單引號結尾。
 - 雙引號開頭，需要雙引號結尾。
- 單雙引號不能混用。
 - 單引號內使用雙引號，可以正確顯示雙引號。
 - 雙引號內使用單引號，也可以正確顯示單引號。

範例：文字字串

```
1  ## 文字字串 ( 要加單雙括號，電腦輸入文字的方式和日常生活不同 )
2  'Bigflower Francis'    # 成雙成對的單引號
3  "田弘華"              # 成雙成對的雙引號
4  "I'm Francis."        # 雙引號內使用單引號
5  '你是 "約翰"'         # 單引號內使用雙引號
```

3-1-4 布林資料

1. 布林：布林值(Boolean value)就是邏輯值。

- 在1800年代中期，英國數學家George Boole發明布林代數，奠定了數位電腦邏輯的基礎。
- 例如電燈的開與關、投籃球進還是沒進、是非題敘述的對與錯等等都是0和1。

2. 布林值只有 True 與 False 兩個值。

Python以 True表示真、以False表示假。

- 程式語言對於英文的大小寫是敏感的。
- 只有True/False 會被視為布林值。

範例：布林值 (一)

```
1  #|echo: true
2  #|eval: false
3  #|includes: false
4  #|
5
6  ## 直接鍵入True或False
7  True
8  False
9
10 ## 大小寫不對會有錯誤訊息
11 true
12 false
13 TRUE
14 FALSE
```

3. 布林值可以視為數值資料。

布林為整數的子類別

- True跟數值1相等
- False跟數值0相等

範例：布林值（二）

```
1 ## 布林值可以視為數值資料
2 bool(0) # False為0
3 bool(1) # True為1
4
5 ## 數值運算中可以納入了布林。
6 True + 3 # False為0
7 False + 3 # False為0
```

3-2 變數賦值

將輸入的資料，儲存在電腦的記憶體中，供日後處理資料與輸出資料之用。

學習目標

- Python變數賦值的語法：變數名字 = 變數值
- 變數一二三：用一個特殊符號等號 = 建立變數；等號的兩邊要給東西，左邊給名字、右邊給資料；名字、等號、資料三樣物件表示Python的變數。

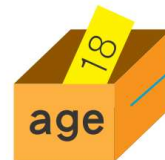
3-2-1 觀念: 用變數儲存資料

「變數」是用來儲存程式中的資料，以提供程式中各種運算之用。

【觀念】將「變數」想像成一個「容器」，它是專門用來「儲放資料」的地方。

- 我們可以把電腦的記憶體想成一個個盒子。盒子裡面放資料，盒子外面貼標籤，那變數就是一個裡面放資料且有標籤的盒子。盒子的名稱是變數名稱(name)，盒子裡面的資料是變數的值(value)。

變數名稱 → age ← 指定成 ← 18 ← 變數的值



變數像一個有標籤的盒子，你能將資料放入盒子中，再利用盒子上的標籤找到你的資料

圖 4-2.1 變數的定義


- 依照不同性質的資料，給予不同的記憶體空間；小東西用小盒子裝，大東西用大箱子裝；以有效地利用主記憶體空間，提高程式的可讀性。


小東西用小盒子裝	大東西用大箱子裝

- 例子：旅館中有房間供客人住宿，房間會有房號，也會有位置、大小與類型的差異。

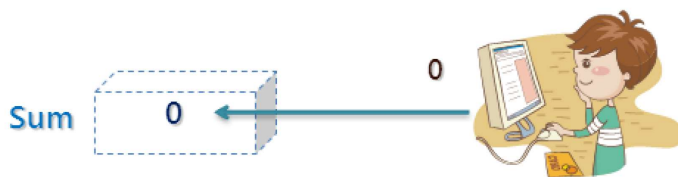
3-2-2 建立變數

用等號 = 建立變數，要給名字與資料 (變數名稱 = 變數的值)。



運算元	指定運算子	運算式的結果 
Sum	=	0

【圖解說明】



- 等號「=」是指派、賦值(assignment)的意思。
 - 名字放在 = 的左邊，要符合命名原則；
 - 資料放在 = 右邊，可以指定變數一個特定值，也可以是一個運算式。
 - 等號前後建議加入空白字元，方便程式閱讀與除錯。
 - 變數賦值時要給初始值，沒有指派就會出現錯誤訊息。

範例：變數賦值

```

1  a           # 光有名字不行
2  a = 1       # 要有名字、等號和資料
3  a = "Python" # 名字在等號左邊、資料在等號右邊
4  a = 1 + 1    # 運算式也可以

```

補充說明：電腦的變數和數學的變數定義不同。

- 電腦的變數是儲存資料的容器
 - 從「程式設計觀點」， $a = a + b$ 這行程式，是把原來a容器中的東西先拿出來($a = 10$)，然後放回a容器的東西是 $a + b$ 的值 ($a = b = 10$)，所以資料容器裡面的東西是20($a+b=20$)，也就是說變數a的值變成是20。
- 數學的變數也代表資料 (未知數)，但與儲存位置無關。
 - 從「數學觀點」， $a = a + b$ ，這個數學式子有問題，根本不可能成立，因為 $0 \neq b$ 。

【隨堂練習1】請判斷下列程式是否正確？如果有誤，請說明其原因。

```

1  #|echo: true
2  #|eval: false
3  #|includes: false
4  #|
5
6  Sum = 0
7  A = 1
8  2 = B # 「=」的左邊是名字，不能放常數。
9  A + B = Sum # 「=」的左邊不能是運算式。

```

```
10 A, B = 1, 2 # 這樣寫可以，是Pythonic的寫法。
11 ## 在電腦上打一次，也可以知道答案
```

3-2-3 選個好名字

Question: 簡述Python語言命名的規則。

1. 變數不能任意命名，要符合規定。

- 變數的名字可以用數字、底線或英文字母。
 - 不能以數字開頭。
 - 除了底線，不能使用【特殊符號】做為變數名稱。
 - 特殊符號例如!、@、#、\$...
 - 使用英文字母
 - 注意大小寫英文字母不同，即A與a被視為不同的變數。
- 不能有空格。
- 不能用Python【保留字】做為變數名稱。
- 不建議用中文命名變數。

Python將一些字串保留起來，指定它們功能，稱為關鍵字或保留字。
Python的關鍵字一共35個，可以用下面的指令匯入 keyword模組查看。

```
import keyword
print(keyword.kwlist)
```

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

2. 變數名稱要有意義。

由英文單字組合而成有意義的名字。

- Snake case:** 小寫英文字、底線（橫線）連結
 - 數學成績變數，以`math_score`表示。
 - `score_sum = math_score + english_score`
- 其他常見的變數命名方式
 - (小)駝峰式命名法：Camel case: `mathScore`
 - 大駝峰命名法：Upper Camel case: `MathScore`
 - Doted.case: `math.score`
 - Kebab case: `math-score`

【隨堂練習2】請判斷下列變數名稱是否正確？如果有誤，請說明其原因。

```
1 #|echo: true
2 #|eval: false
3 #|includes: false
4
5
6 _index
7 data01
8 width
9 department_no
10
11 3_pass      # 不能以數字開頭
12 while      # 不能使用關鍵字
```

```
13 $money      # 不能使用特殊符號
14 pass word   # 不能有空格
```

Ans: 前半部正確，後半部錯誤。

3-3 其他

3-3-1 資料類型的判斷(Optional)

使用 `type()` 函數可以判斷到底資料是哪一種型別。

範例：用 `type()` 判斷資料類型

```
1 #|echo: true
2 #|eval: false
3 #|includes: false
4 #|
5
6 my_int = 87 ; type(my_int);type(87)
7
8 my_float = 8.7 ; type(my_float); type(8.7)
9
10 my_str = "8.7" ; type(my_str); type("8.7")
11
12 bool_true = True ; type(bool_true); type(True)
```

3-3-2 程式註解

程式的好壞在於是否每一個人都看得懂你寫的程式。要自己和他人能看懂所寫的程式，加註解是個好方法。



註解敘述(comment statements) 很重要

- 註解是幫助大家看懂程式。
- 註解不是寫給電腦看的，是寫給人看的（未來的自己和他人）。
- 註解越清楚詳細越好。
- 程式碼越多，越需要加入相關註解，將來除錯才會比較容易。

二種註解方式：單行註解與多行註解

- 單行註解以井字號 `#` 開頭，在同一行後面的文字即為註解敘述；
 - 第一行可以寫「問題是什麼」，或是「解題思路」的關鍵。
 - 在程式的開頭之前，說明程式設計的用意，**用意註解**
 - 以兩個井字號，依照Input-Process-Output架構加註。
 - 以兩個井字號與數字，說明解決問題的步驟（即加上虛擬碼）。

- 在一行程式結束的後面，說明指令的用法，**用法註解**。
 - 以一個井字號，說明Python程式語言的語法。
- 多行註解以三個單引號或三個雙引號開頭與結尾，之間的文字即為註解，適用於補充內容較多時。