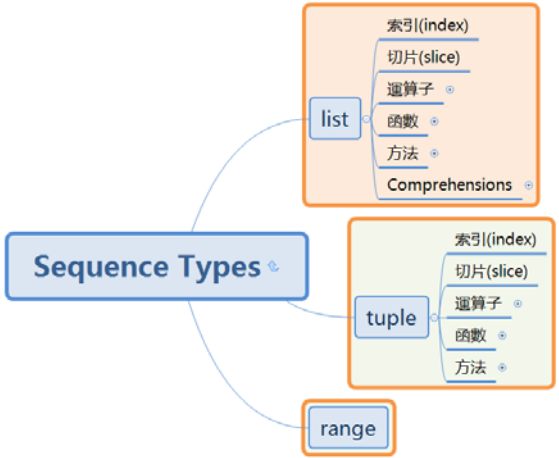
串列(List)與元組(Tuple)

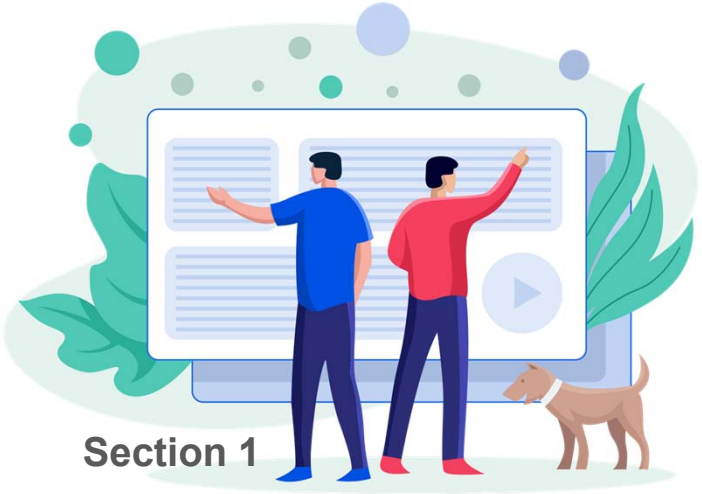
學習目標

- 元組(Tuple) 宣告與運算
- 元組(Tuple) 相關的函數與方法
- 串列(List) 的 Comprehensions



元組(Tuple)的用法

- 介紹元組(Tuple) 的特性
- 如何存取元組(Tuple) 的內容



元組(Tuple)

- Tuple 是有順序(sequence)，內容不可改變(immutable)
- 使用小括號括起來，每一個項目以逗點分隔
- 項目的資料型態可以不同
  - tup1 = ('bdse', 'aien', 2015, 2018)
  - tup2 = (1, 2, 3, 4, 5)
  - tup3 = "a", "b", "c", "d"
  - tup4 = () # empty tuple
  - tup5 = (50,) # single value tuple

取得元組的資料

- 要取得 Tuple 內容，使用中括號配合註標或切片(slice) 來獲得
-
```
>>> tup1 = ('aiot', 'python', 23.5, 2019)
>>> tup2 = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
>>> tup1[0]
'aiot'
>>> tup2[1:5]
(2, 3, 4, 5)
>>> tup2[1:10:2]
(2, 4, 6, 8, 10)
>>>
```

更新元組內容 -1

- Tuple 內容是不可改變的(immutable)，亦即無法更新或刪除項目
- 可以利用已存在的 Tuple 來產生新的 Tuple

```
>>> tup1 = (12, 34.56)
>>> tup2 = ('java', 'python')

# tuples are immutable
# tup1[0] = 100

# create a new tuple
>>> tup3 = tup1 + tup2
>>> tup3
(12, 34.56, 'java', 'python')
>>>
```

```
>>> tup[0]
10
>>> tup[0]=100
Traceback (most recent call last):
  File "<pyshell#5>", line 1, in <module>
    tup[0]=100
TypeError: 'tuple' object does not support item assignment
```

- tup1[0] = 100，會發生 “TypeError: ‘tuple’ object does not support item assignment” 的錯誤，因為Tuple的內容是不可改變的(immutale)。
- Tuple是不可改變其數值的，只能新增一個全新的Tuple

更新元組內容 -2

- Tuple 內容也可以是 List、Tuple 或 Set
-
```
>>> tup = (33, 'python', [11, 12, 13], 77)
>>> tup[0]
33
>>> tup[2]
[11, 12, 13]
>>> tup[2][1]
12
>>> tup[2][1] = 40   # update list, not tuple
>>> tup
(33, 'python', [11, 40, 13], 77)
>>>
```

更新元組內容 -3

- 常見的不可變物件：
  - 數字類型：int，float，complex
  - 字串(String)
  - 元組(Tuple)
  - 冷凍集(Frozenset)
- 常見的可變物件：
  - 串列(List)
  - 字典(Dict)
  - 組(Set)
  - byte array

```
>>> lst=[10,20,30]
>>> tup=10,20,30
>>> lst[2]=tup
>>> lst
[10, 20, (10, 20, 30)]
>>> lst[2][0]=100
Traceback (most recent call last):
  File "<pyshell#17>", line 1, in <module>
    lst[2][0]=100
TypeError: 'tuple' object does not support item assignment
```

- 改動與否，視最後改動的對象為準

元組索引與切片

- 索引與切片的方式同 String 或 List
- 假設 tp 資料如下
  - tp = ('banana', 'apple', 'grape')

運算	結果	說明
tp[0]	'banana'	索引從零算起
tp[-2]	'apple'	倒數第二個元素
tp[1:]	('apple', 'grape')	切片取子元組

元組基本運算

運算	結果	說明
len((10, 20, 30))	3	求元組個數
(1, 2, 3) + (4, 5, 6)	(1, 2, 3, 4, 5, 6)	元組結合
('Hi!',) * 4	('Hi!', 'Hi!', 'Hi!', 'Hi!')	元組重複
30 in (10, 20, 30)	True	成員運算
for x in (10,20,30): print (x,end = ' ')		配合迴圈處理元組元素

```
>>> len((10,20,30))
3
>>> len(10,20,30)
Traceback (most recent call last):
  File "<pyshell#19>", line 1, in <module>
    len(10,20,30)
TypeError: len() takes exactly one argument (3 given)
```

- len((10,20,30))的Tuple的小括號不可以省略，因為len()函數內只能有一個變數。須注意，有些地方小括號可以省略，有些地方小括號不能省略。

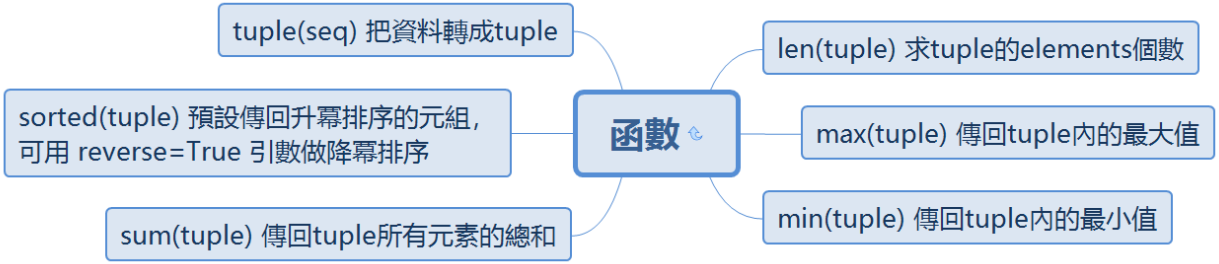
元組相關函數與方法

- 使用元組相關的函數與方法

Section 2

內建元組相關函數 -1

- len(tuple)
  - 取得 tuple 項目個數
- max(tuple)
  - 傳回 tuple 內的最大值
  - 所有項目必須是相同的資料型態
- min(tuple)
  - 傳回 tuple 內的最小值
  - 所有項目必須是相同的資料型態



內建元組相關函數 -2

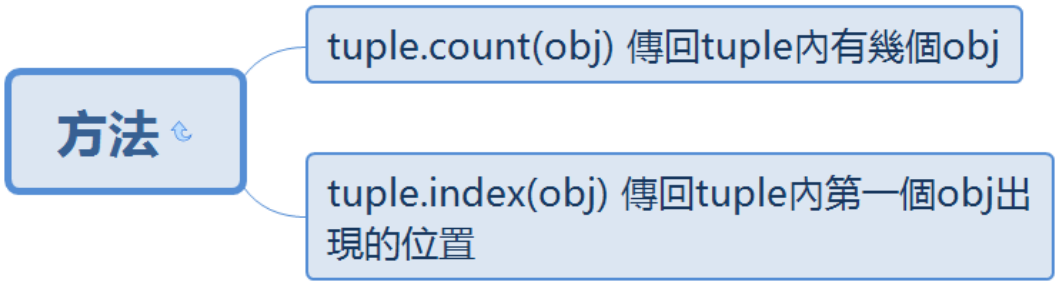
- sum(tuple)
  - 傳回 tuple 所有元素的總和
  - 所有項目必須是數值資料型態
- sorted(tuple)
  - 預設傳回升冪排序的串列(List)，可用 reverse=True 引數做降冪排序
  - 所有項目必須是相同的資料型態
- tuple(seq)
  - 將資料轉成 tuple

```
>>> tuple(range(6))
(0, 1, 2, 3, 4, 5)
```

  - Tuple的速度比List快，在處理大數據時，使用Tuple會比較有效率

元組相關方法

- tuple.count(obj)
  - 傳回 tuple 內有幾個 obj
- tuple.index(obj)
  - 傳回 tuple 內第一個 obj 出現的位置



元組(tuple)的用法(Demo)

- 如何使用元組(Tuple)

List Comprehensions

- 介紹 List comprehensions 的語法
- 如何使用 List comprehensions

何謂List Comprehensions

- Python支援一種稱為 “list comprehensions” 的概念，以一個非常直接簡單的方式來建構串列(List)，感覺跟數學的公式很像
- 語法
  - [expression for item in iterable]
  - [expression for item in iterable if condition]
- 數學公式
  - $S = \{x^2 : x \text{ in } \{0 .. 9\}\}$
  - $V = (1, 2, 4, 8, ..., 2^{10})$
  - $M = \{x \mid x \text{ in } S \text{ and } x \text{ even}\}$

Python List Comprehensions -1

- Python寫法
  - >>> S = [x**2 for x in range(10)]
    >>> S
    [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
  - >>> V = [2**i for i in range(11)]
    >>> V
    [1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]
  - >>> M = [x for x in S if x % 2 == 0]
    >>> M
    [0, 4, 16, 36, 64]
- 也可以應用到字典(Dict) 與集合(Set) 的建置

Python List Comprehensions -2

- 如果不使用Comprehension來實做，就需使用到for loop或while loop來實現。但其實Python的設計邏輯可以在不使用loop的情況下，製作出所需的序列資料。這也是資料科學家喜歡使用Python的原因之一。
- 不需寫迴圈
- Dict 與 Set 的 List comprehensions 後續章節會介紹
- Tuple沒有Comprehension的觀念

```
>>> S = [x**2 for x in range(10)]
>>> S
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
>>> V = [2**i for i in range(11)]
>>> V
[1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]
>>> M = [x for x in S if x%2==0]
>>> M
[0, 4, 16, 36, 64]

>>> N = [x for x in S if x%2]
>>> N
[1, 9, 25, 49, 81]
```

List Comprehensions(Demo)

- 使用 List comprehensions 簡化程式

本章重點精華回顧

- 元組(Tuple) 的特性與用途
- List comprehensions 的用法

Lab: 串列與元組

- Lab01: 使用元組(Tuple)
- Lab02: 使用 List comprehensions

Lab01: 使用元組(Tuple)

- 啟動Python互動式執行環境，做以下練習

```
>>> tup1 = ()
>>> tup1
>>> type(tup1)
>>> tup2 = (50)
>>> tup2
>>> type(tup2)
>>> tup3 = (50,)
>>> tup3
>>> type(tup3)
```

```
>>> tup4 = (1,2,3,4,5)
>>> tup4
>>> tup5 = ('eng',70,'math',82,'comp',63)
>>> tup5
>>> tup5[1]
>>> len(tup5)
```

```
>>> t1 = (20,40)
>>> t2 = (60,80,100)
>>> t1
>>> t2
>>> t1 + t2
>>> t1 * 3
>>> 60 in t2
>>> 40 not in t1
```

```
>>> tup = tuple('Hello Python')
>>> tup[1:4]
>>> tup[1:6]
>>> tup[:8]
>>> tup[3:]
>>> tup[2:-1]
>>> tup[6:-2]
>>> tup[1:10:3]
>>> tup[1:len(tup):3]
```

```
>>> tup = (33, 'python', [11, 12, 13], 77)
>>> tup[0]
>>> tup[2]
>>> tup[2][1]
>>> tup[2][1] = 40
>>> tup[2][1]
>>> tup
```

Lab02: 使用List comprehensions

- 啟動Python互動式執行環境，做以下練習
- >>> list1 = [x for x in range(10)]
- >>> list1
- >>> list2 = [x+1 for x in range(10)]
- >>> list2
- >>> list3 = [x for x in range(10) if x % 2 == 0]
- >>> list3
- >>> list4 = [x*2 for x in range(10) if x % 2 == 0]
- >>> list4