在前面的第二章，我們已經學習到，**程式中的資料都是以變數的方式來儲存**。但試想，若今日有 3000 筆資料，難道真的需要宣告 3000 個變數嗎？因此**容器資料型別非常適合處理大量資料的儲存**。

Python 中，提供四種容器資料型別，(1)**串列(List)**、(2)**序對(Tuple)**、(3)**字典(Dict)**、(4)**集合(Set)**。容器資料型別的特色如下：

(1) 每個容器資料型別**擁有一個名稱**。 (即指派變數名稱)

(2) 每個容器資料型別中可以有**多個元素，且每個元素可以是不同的資料型態**。

(3) 每個元素可透過容器中的**索引值**或**鍵值**，來取得元素值。

一、串列(List)

1. 一維串列

(1) 串列以**中括號[]**存放元素，各個元素的資料型態可以相同，也可以不同，其**宣告一維串列**的語法為：

```
a = [1,2,3,4,5]                 # a 串列，分別有 1,2,3,4,5 的元素，且皆為整數
b = ["蘋果","香蕉","橘子"]       # b 串列，分別有"蘋果","香蕉","橘子"的元素，且皆為字串
c = [100,"香蕉",True]            # c 串列，分別有 100,"香蕉",True 的元素，且為不同資料型態
d = []                           # d 串列，空串列。
e = [0]*4                        # e 串列，分別預設 0,0,0,0 的元素。
f = [""]*4                       # f 串列，分別預設四個空字串的元素。
g = [i for i in range(1,6)]      # g 串列，分別有 1,2,3,4,5 元素。
```

(2) **取得一維串列元素**時，需透過索引值，**索引值由 0 開始**，其語法為：

```
b = ["蘋果","香蕉","橘子"]
print(b[0])      #輸出蘋果
print(b[2])      #輸出橘子
print(b[3])      #程式錯誤，索引值超過範圍
print(b[-1])     #輸出橘子 (取倒數第一個)
print(b[-3])     #輸出蘋果 (取倒數第三個)
print(b[-4])     #程式錯誤，索引值超過範圍
print(b[1:3])    #輸出["香蕉","橘子"]，此為串列的資料型態。
```

(3) **設定一維串列元素**時，需透過索引值，**索引值由 0 開始**，其語法為：**(僅適用於非空串列)**

```
b = ["蘋果","香蕉","橘子"]
b[1] = "火龍果"
print(b)                    #輸出 ['蘋果', '火龍果', '橘子']
```

(4) **刪除一維串列元素**時，使用 **del 指令**，其語法為：

```
b = ["蘋果","香蕉","橘子"]
del b[1]
print(b)                    #輸出 ['蘋果', '橘子']
```

2. 二維串列

(1) 串列的元素也可以是另一個串列，稱為**多維串列**，其**宣告二維串列**的語法為：

```
a = [[1,2,3] , [4,5,6] ]          # a 為 2*3 的二維串列，分別有[1,2,3] , [4,5,6]的元素。
b = [[]]                           # b 為空的二維串列。
c = [ [""]*5 for i in range(4) ]   # c 為 4*5 的二維串列，元素皆為空字串。
d = [ [0]*5 for i in range(4) ]    # d 為 4*5 的二維串列，元素皆為 0。
```

(2) **取得二維串列**元素，需透過索引值，**索引值由 0 開始**，其語法為：

```
a = [ ["joe","1234"], ["mary","abcd"], ["david","5678"] ]
print(a[1])          #輸出["mary","abcd"]
print(a[1][1])       #輸出 abcd
print(a[1][0:2])     #輸出["mary","abcd"]
```

(3) **設定二維串列元素**時，需透過索引值，**索引值由 0 開始**，其語法為：**(僅適用於非空串列)**

```
a = [ ["joe","1234"], ["mary","abcd"], ["david","5678"] ]
a[0] = "kkbox"
a[1][0]="john"
print(a)             #輸出  ['kkbox', ['john', 'abcd'], ['david', '5678']]
```

(4) **刪除二維串列元素**時，使用 **del 指令**，其語法為：

```
a = [ ["joe","1234"], ["mary","abcd"], ["david","5678"] ]
del a[1]
print(a)             #輸出  [['joe', '1234'], ['david', '5678']]
del a[0][1]
print(a)             #輸出  [['joe'], ['david', '5678']]
```

3. 常用的運算子

運算子	範例	範例結果
+	[2,4,6] + [8,10,12]	[2,4,6,8,10,12]
*	[2,4,6] * 3	[2,4,6,2,4,6,2,4,6]
>、<、>=、<=	[2,4,6] > [2,3,8]	True
==	[2,4,6] == [2,3,8]	False
in	2 in [2,4,6]	True
not in	2 not in [2,4,6]	False

4. 常用的方法

方法	說明	範例	範例結果	其他說明
append()	將新元素加到末端。	a = ["C","A","B"] a.append("D") print(a)	["C","A","B","D"]	※適用於空串列 / 非空串列。
insert()	插入新元素到指定的**索引編號**。	a = ["C","A","B"] a.insert(1,"D") print(a)	["C","D","A","B"]	※適用於空串列 / 非空串列。
pop()	移除指定**索引編號**的元素，若未指定則刪除最後一個。	a = ["C","A","B"] a.pop(1) print(a)	["C","B"]	
remove()	移除指定**元素值**。	a = ["C","A","B"] a.remove("C") print(a)	["A","B"]	
count()	計算出現次數。	a = ["C","A","B"] print(a.count("C"))	1	
sort()	將元素進行**遞增排序**。	a = ["C","A","B"] a.sort() print(a)	["A","B","C"]	※原串列順序會被更改。 ※遞減排序: sort(reverse=True)
reverse()	**反轉**元素順序。	a = ["C","A","B"] a.reverse() print(a)	["B","A","C"]	
index()	搜尋元素的**索引值**。	a = ["C","A","B"] print(a.index("A"))	1	※若找不到，程式會發生錯誤

5. 常用的函數

方法	說明	範例	範例結果	其他說明
sum()	計算**總分**	a = [10,20,30] print(sum(a))	60	
max()	傳回串列中的**最大值**	a = [10,20,30] print(max(a))	30	
min()	傳回串列中的**最小值**	a = [10,20,30] print(min(a))	10	
len()	傳回串列中的元素**個數**	a = [10,20,30] print(len(a))	3	
sorted()	將元素進行**遞增排序**。	a = [30,20,10] b = sorted(a) print(b)	[10,20,30]	※原串列順序不會被更改。 ※遞減排序: sorted(a,reverse=True)

二、序對(Tuple)

1. 序對(Tuple)與串列(List)的概念類似，差別在於：

(1) 序對以**小括號()**存放元素。

(2) **元素一旦設定後，便無法更改值或位置**。因此若有牽涉更動元素值或位置的相關方法、函數皆無法使用。例如：append()、insert()、pop()、remove()、sort()、reverse()等皆無法使用。

```
a = (1,2,3,4,5)      # a 序對，分別有 1,2,3,4,5 的元素，且皆為整數
b = ((1,2),(4,5),(7,8))  # b 序對，為 3*2 的二維序對
c = ()               # c 序對，為空序對
```

三、字典(Dict)

1. 字典

(1) 字典以**大括號{}**存放元素，每個元素是由一對「鍵值(key):元素值(value)」組合而成，**若鍵值重複出現，其值會被覆蓋**，其**宣告字典**的語法為：

```
a = {"id":1081416, "name":"黃健崴"}   # a 字典，鍵值有 id、name。
b = {}                                # b 字典，空字典。
```

(2) **取得字典**元素，需透過**鍵值**，其語法為：

```
print(a["id"])      #輸出 1081416
print(a["name"])    #輸出黃健崴
```

(3) **設定字典**元素，需透過**鍵值**，其語法為：

```
a["id"] = 1081401
print(a)            #輸出{'id': 1081401, 'name': '黃健崴'}
```

(4) **刪除字典元素**時，使用 **del 指令**，其語法為：

```
del a["id"]
print(a)            #輸出{'name': '黃健崴'}
```

2. 常用的運算子

運算子	範例	範例結果
!=	a["id"] != 1081401	True
==	a["id"] == 1081401	False
in	"id" in a	True
not in	"id" not in a	False

3. 常用的方法

方法	說明	範例	範例結果
clear()	清空字典。	a = {"name":"健崴","city":"台北"} a.clear() print(a)	{}
copy()	複製字典。	a = {"name":"健崴","city":"台北"} b = a.copy() print(b)	{"name":"健崴","city":"台北"}
get()	取得指定 key 的元素值。	a = {"name":"健崴","city":"台北"} b = a.get("name") print(b)	健崴
pop()	移除指定 key 的元素值。	a = {"name":"健崴","city":"台北"} a.pop("name") print(a)	{"city":"台北"}
update()	合併字典。	a = {"name":"健崴","city":"台北"} b= {"name":"佑宗","id":1} a.update(b) print(a)	{'name': '佑宗', 'city': '台北', 'id': 1}
items() keys() values()	取鍵值與元素值 取鍵值 取元素值	a = {"name":"健崴","city":"台北"} print(a.items()) print(a.keys()) print(a.values())	dict_items([('name', '健崴'), ('city', '台北')]) dict_keys(['name', 'city']) dict_values(['健崴', '台北'])

4. 常用的函數

方法	說明	範例	範例結果
len()	傳回字典中的 key:value **組數**	a = {"name":"健崴","city":"台北"} print(len(a))	2

四、集合(Set)

1. 集合(Set)與字典(Dict)的概念類似，差別在於：

(1) 其值重複時，僅會保留一個。

(2) 只有元素值(value)，**沒有鍵值(key)**。

```
a = {1,2,3,4,5}      # a 集合，分別有 1,2,3,4,5 的元素，且皆為整數
b = set()            # b 集合，空集合
```

2. 常用的運算子

運算子	集合(Set)	範例	範例結果
聯集(|)	a = {1,2,3,4,5} b = {2,4,6,8,10}	a | b	{1, 2, 3, 4, 5, 6, 8, 10}
交集(&)		a & b	{2, 4}
差集(-)		a - b	{1, 3, 5}
互斥或(^)		a ^ b	{1, 3, 5, 6, 8, 10}

3. 常用的方法

方法	說明	範例	範例結果
add()	新增元素。	a = {1,2,3,4,5} a.add(6) print(a)	{1,2,3,4,5,6}
remove()	移除元素。	a = {1,2,3,4,5} a.remove(3) print(a)	{1,2,4,5}
update()	合併集合。	a = {1,2,3,4,5} b = {2,4,6,8,10} a.update(b) print(a)	{1, 2, 3, 4, 5, 6, 8, 10}
clear()	清空集合。	a = {1,2,3,4,5} a.clear() print(a)	set{}

五、強制轉換容器資料型別

- **list()**：強制**轉換為串列**資料型態。
- **tuple()**：強制**轉換為序對**資料型態。
- **set()**：強制**轉換為集合**資料型態。
- **dict(zip(key 串列, value 串列))**：強制**轉換為字典**資料型態。