04：進階串列操作

2018.1.30

串列操作 – 取值

- list1 = [1, 2, 3, 4, 5, 6]

	方法	意義	範例	範例結果
	list1*n	串列重覆 n 次	list2=list1*2	list2=[1,2,3,4,5,6,1,2,3,4,5,6]
★	list1[n1:n2]	取出 n1 到 n2-1 元素	list2=list1[1:4]	list2=[2,3,4]
	list1[n1:n2:n3]	同上，取出間隔為 n3	list2=list1[1:4:2]	list2=[2,4]
★	del list1[n1:n2]	刪除 n1 到 n2-1 元素	del list1[1:4]	list1=[1,5,6]
	del list1[n1:n2:n3]	同上，刪除間隔為 n3	del list1[1:4:2]	list1=[1,3,5,6]

串列操作 – 串列屬性

- list1 = [1, 2, 3, 4, 5, 6]

	方法	意義	範例	範例結果
★	n=len(list1)	取得串列元素數目	n=len(list1)	n=6
★	n=min(list1)	取得元素最小值	n=min(list1)	n=1
★	n=max(list1)	取得元素最大值	n=max(list1)	n=6
★	n = sum(list1)	取得串列元素的和	n=sum(list1)	n=21

串列操作 – 進階1

- list1 = [1, 2, 3, 4, 5, 6]; x = [8, 9]

	方法	意義	範例	範例結果
	n=list1.index(n1)	第 1 次 n1 元素的索引值	n=list1.index(3)	n=2
	n=list1.count(n1)	n1 元素出現的次數	n=list1.count(3)	n=1
★★	list1.append(n1)	將 n1 做為元素加在串列最後	list1.append(8)	list1=[1,2,3,4,5,6,8]
★	list1.extend(x)	將 x 中元素逐一做為元素加在串列最後	list1.extend(x)	list1=[1,2,3,4,5,6,8,9]
★	list1.insert(n,n1)	在位置 n 加入 n1 元素	list1.insert(3,8)	list1=[1,2,3,8,4,5,6]

串列操作 – 進階2

- list1 = [1, 2, 3, 4, 5, 6]; x = [8, 9]

方法	意義	範例	範例結果
n=list1.pop()	取出最後 1 個元素並由串列中移除元素	n=list1.pop()	n=6, list1=[1,2,3,4,5]
list1.remove(n1)	移除第 1 次的 n1 元素	list1.remove(3)	list1=[1,2,4,5,6]
list1.reverse()	反轉串列順序	list1.reverse()	list1=[6,5,4,3,2,1]
list1.sort()	將串列由小到大排序	list1.sort()	list1=[1,2,3,4,5,6]

增加串列元素

- 串列設定初始值後，要如何增加串列元素？
  - list1 = [3, 5, 7]
  - 直接用 list1[4] = 9 ???

增加串列元素的方法

insert()

append()

extend()

insert() vs append()

- insert() 方法是將元素加在串列的指定位置
  - 注意索引值不能超過串列元素個數，否則會產生「索引超過範圍」的錯誤
- append() 方法是將元素加在串列最後面

insert() vs append()

```
lst = [3, 5, 7]

lst.insert(2, 100)
print(lst)

lst.insert(10, 99)
print(lst)

lst.append(888)
print(lst)
```


extend() vs append()

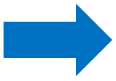
- 這兩個方法都是將資料加在串列最後面，不同處在於
  - append() 方法的參數可以是元素，也可以是串列
    - 如果是串列，會將整個串列當成一個元素加入串列
  - extend() 方法的參數只可以是串列，不可以是元素
    - extend() 方法會將串列中的元素做為個別元素逐一加入串列

extend() vs append()

```
3 lst = [3, 5, 7]
4 lst1 = [2, 4, 6, 8]
5
6 lst.append(lst1)
7 print(lst)
```

```
[3, 5, 7, [2, 4, 6, 8]]
```

```
3 lst = [3, 5, 7]
4 lst1 = [2, 4, 6, 8]
5
6 lst.extend(lst1)
7 print(lst)
```



```
[3, 5, 7, 2, 4, 6, 8]
```


pop()

- pop() 方法的功能是由串列中取出元素，同時串列會移除該元素
  - pop() 方法可以有參數，也可以沒有參數：
    - 沒有參數 ➔ 取出最後 1 個元素
    - 有參數且為整數 ➔ 取出以參數為索引值的元素

```
3 lst = [1, 2, 3, 4, 5, 6]
4
5 print(lst.pop())
6 print(lst.pop(3))
```

存取串列內容

- 要存取串列內部的元素，傳統上使用索引
  - 假設有一個串列叫做 lst，長度為 10，則它的所有元素為 lst[0], lst[1], lst[2], ..., lst[8], lst[9]
  - Python 除了傳統的索引值外，提供了一個像瑞士小刀一樣的東西叫分割(slice)

串列分割

- lst[m:n] ➔ 代表從第 m 個元素開始取，取到第 n-1 個元素為止
  - lst[m:] → 從第 m 個元素開始取到最後
  - lst[:n] → 從開頭取到第 n-1 個元素
  - lst[:] → 全取
- lst[m:n:t] ➔ 從第 m 個到第 n-1 個元素，每隔 t 個取一次

範例

```
2 lst = [2, 3, 5, 2, 33, 21, 99]
3
4 print(lst[2:4])
5 print(lst[:3])
6 print(lst[5:])
7 print(lst[-4:-2])
8 print(lst[1:5:2])
```

```
[5, 2]
[2, 3, 5]
[21, 99]
[2, 33]
[3, 2]
```


想想看

- 如何將串列以反轉的順序輸出？
  - lst[2, 3, 5, 2, 33, 21, 99]

- 如何將使用者輸入的字串以反轉的順序輸出？

+, *, in/not in 運算子

- 連結運算子(+)用來結合兩個字串
- 重複運算子(*)可複製串列的元素
- in/not in 可判斷某元素是否在串列中

```
[2, 3, 5, 9, 33, 99]
[2, 3, 5, 9, 33, 99, 33, 99]
False
```

```
2 lst1 = [2, 3, 5, 9]
3 lst2 = [33, 99]
4
5 print(lst1 + lst2)
6 print(lst1 + lst2*2)
7
8 print(3 in lst2)
```


追蹤串列元素

```
2 lst = [2, 3, 5, 9, 33, 21, 99]
3
4 for i in lst:
5     print(i)
6
7 for i in range(0, len(lst), 3):
8     print(lst[i])
```

串列解析(List Comprehension)

- 串列解析提供一個簡潔方法建立一個循序串列
  - 由中括號裡面包含 for 子句後接運算式所組成
  - 可以零個或多個 for 或 if 子句
  - 串列解析後將產生一個新串列

```
lst = [x for x in range(5)]
```

```
[a for a in lst if a%2]
```


簡化程式碼

```
week_day = eval(input("Enter a week number (1~7) : "))

if week_day == 1:    print("The week is Mon.")
elif week_day == 2:  print("The week is Tues.")
elif week_day == 3:  print("The week is Wed.")
elif week_day == 4:  print("The week is Thu.")
elif week_day == 5:  print("The week is Fri.")
elif week_day == 6:  print("The week is Sat.")
else:                print("The week is Sun.")
```

```
week = ["Mon.", "Tues.", "Wed.", "Thu.",
        "Fri.", "Sat.", "Sun."]

week_day = eval(input("Enter a week number (1~7) : "))
print("The week is", week[week_day-1])
```

PY3-0002

四、數字反轉判斷

2. 設計說明：
   (1) 請撰寫一程式，讓使用者輸入一個正整數，將此正整數以反轉的順序輸出，並判斷如輸入的正整數為 0，則輸出為 0。

TO BE CONTINUED...