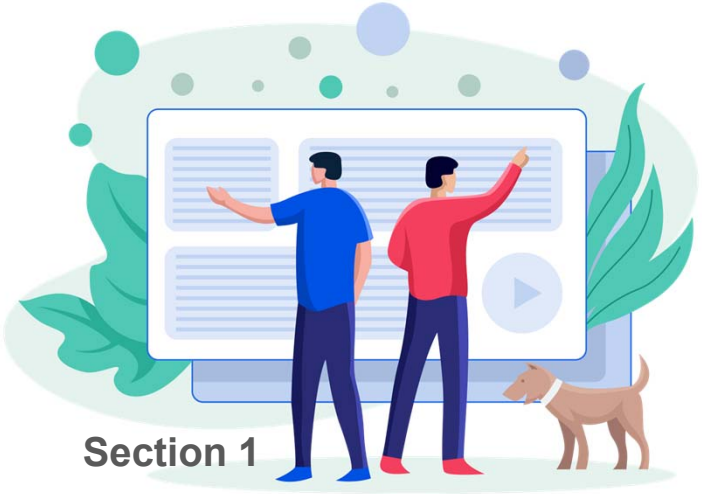
# 函數(Function)設計

# 學習目標

- 函數(Function) 的定義方式
- 函數(Function) 傳引數(Arguments) 與回傳值(Return value)
- 如何使用函數

# 函數設計

- 介紹函數(Function) 的定義方式
- 如何使用函數(Call function)



# 函數(Functions)

- 函數是一個程式碼區塊組成的，可重複使用，一般是要執行某個特定功能
- 函數提供應用程式更佳的模組化能力，以及更高的可重用性(reusable)
- Python 已經提供一些內建的函數，例如 print()，但是也可以自訂函數，這些自訂的函數稱為使用者自訂函數(User-defined functions)

# 定義函數 -1

- 函數定義以 “def” 關鍵字開頭，後面加上函數名稱及一對小括號 “( )”
- 函數命名法則依照 PEP-8 文件，小寫字元配合底線以增加可讀
- Function name建議全部使用英文小寫，如果真的要區隔不同單字，可以使用下底線 _
- 小括號內可以定義要接收的參數(Formal parameters)
- 小括號後面接冒號(:)，底下就開始縮排，撰寫函數程式內容
- “return [expression]” 敘述可以結束函數，也可以傳回一個運算結果給呼叫者(Caller)
  - 只有 return 敘述而沒有引數可以視為 “return None”
  - 如果函數內沒有明確的使用 return 敘述，預設為 “return None”
  - 可以使用元組(tuple) 格式傳回多個資料

# 定義函數 -2

- 語法
  - def functionname(parameters):
    function_suite
    [return [expression]]
- 範例
  - def make_a_sound():
    print('quack')
  - def printme(pstr):
    print(pstr)
  - def sum_nums(n1, n2):
    return n1+n2

# 使用函數

- 一旦函數定義完成，就可以呼叫函數
- 
```
# Function definition
def make_a_sound():
    """This function print duck sound"""
    print('quack')

# Call function
make_a_sound()
for i in range(3):
    make_a_sound()
```

變數 i 若沒有使用到，可使用底線 _ 代替，表示沒有使用到這個變數

# Document string

- 三個雙引號可用來定義document string。
- 三個雙引號可以註記函數、模組、類別的document string。
- #也是註解，但#的註解是給自己看的。而document string則是要讓其他使用此函數的人看的。說明該函數、模組、類別的功能。
- 可以這樣理解：我要自己看的用#，要給別人參考的使用document string。

# 函數設計(Demo)

- 如何定義函數(Function)
- 如何使用函數(Call function)

# 函數的引數

- 熟悉函數傳引數(Arguments) 的方式

Section 2

# Pass By Assignment -1

- Python的引數(Arguments) 是以 “Pass by assignment” 方式傳遞
- Assignment 只是建立物件的參考(References to objects)
- 呼叫者(Caller) 與被呼叫者(Callee) 的引數彼此之間沒有別名(Alias) 的關係，所以沒有傳參考呼叫(Call-by-Reference) 的方式

# Pass By Assignment -2

```
# Function definition
def changeme(myvar):
    print('In function, before change:', myvar)
    myvar = 50
    print('In function, after change:', myvar)

# Call function
myvar = 20
changeme(myvar)
print('Outside function:', myvar)
```

# Pass By Assignment -3

```
# Function definition
def changeme(mylist):
    print('In function, before change:', mylist)
    mylist[2] = 50
    print('In function, after change:', mylist)

# Call function
mylist = [10,20,30]
changeme(mylist)
print('Outside function:', mylist)
```

# 函數的引數(Demo)

- 介紹函數傳引數(Arguments) 的方式

# 本章重點精華回顧

- 函數(Function) 的定義與使用
- 函數如何接引數(Arguments) 與傳回值

# Lab: 函數(Function)設計

- Lab01: 定義與使用函數(Function)
- Lab02: 使用函數的引數(Arguments)

# Lab01: 定義函數(Function)

1. 啟動Python IDLE環境
2. 使用 “File/Open…” 開啟 “func_def1.py” 程式，了解函數的設計與使用
3. 關閉func_def1.py程式視窗
4. 使用 “File/Open…” 開啟 “func_def2.py” 程式，了解函數的設計與使用
5. 關閉func_def2.py程式視窗

# Lab02: 使用函數的引數(Arguments)

1. 啟動Python IDLE環境，做以下練習
2. 使用 “File/Open…” 開啟 “argument1.py” 程式，了解pass by assignment的風格
3. 關閉argument1.py程式視窗
4. 使用 “File/Open…” 開啟 “argument2.py” 程式，了解pass by assignment的風格
5. 關閉argument2.py程式視窗