# 補充學習

## 函式的文件字串

我們可以在函式區塊的第一行，加上**文件字串**（docstring）來說明函式的功能及用法，這個文件會被存入函式的程式編碼中，未來若用 Python 內建函式 help() 來查看函式說明，就會連同文件字串一起顯示，例如：

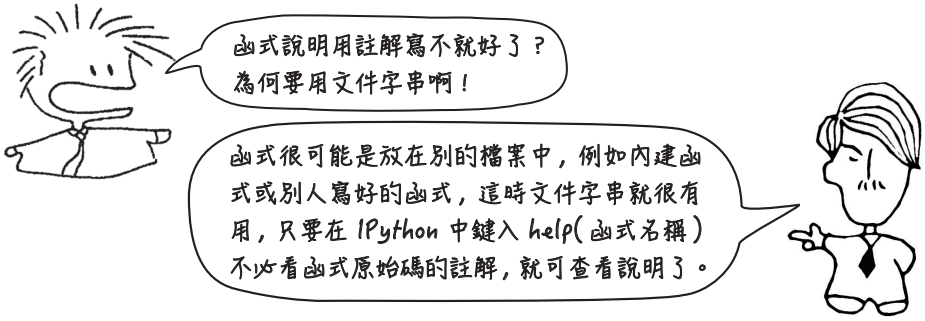
```
def calc(w, h, d=1):
    '計算體積，若 d 省略則可計算面積'  ← 說明字串必須加在第一行
    return w * h * d

help(calc)  ← 用 help() 查看函式的參數定義及文件字串
```

輸出

```
Help on function calc in module __main__:

calc(w, h, d=1)  ← 函式的參數定義
    計算體積，若 d 省略則可計算面積  ← 文件字串
```




文件字串其實是儲存在函式的 __doc__ 屬性中，因此也可以用例如 calc.__doc__ 來讀取。

## Python 的心理學!!!

初學 Python 最有趣的是，因為變數不用事先宣告，所以常會發生讓人疑惑的狀況，例如在看別人的程式時，突然出現一個奇怪的變數，例如：

```
my_level = game_level('John')
```

其中 game_level() 是程式中已 def 過的函式，所以不會奇怪，但 my_level 是突然跑出來的一個變數，它怎麼來的呢？前前後後都找不到線索！其實它就只是一個名牌，用來綁住 game_level() 函式的傳回值而已。因為變數不用事先宣告，隨手可以新創一個來用，往往會突然讓人摸不著頭緒，這尤其是學過其他 "需要事先宣告變數型別" 的靜態程式語言（例如 C/C++、Java 等）的人的心理障礙！

## 不只是心理學!!!

變數的值其實是取決於等號右邊的資料，如果等號右邊是一個函式，那麼就必須先了解該函式會傳回什麼資料，才能知道變數會綁定到什麼樣的資料上。例如底下的 AI 深度學習程式：

這裡只是用來示範，有些地方看不懂沒關係，也不用跟著操作喔！(from... import... 將在第 5 章介紹)

```
from keras.datasets import mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
print(type(train_images))
print(train_images)
```

呼叫 load_data() 來下載 MNIST 手寫數字圖庫

依照傳回的資料結構，在等號左邊用 2 層 tuple 的 4 個變數來承接傳回值

顯示第一個變數的型別

顯示第一個變數的內容

輸出

```
<class 'numpy.ndarray'>
[[[0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  ...
  [0 0 0 ... 0 0 0]]
  ...
  [0 0 0 ... 0 0 0]]]
```

它是一個特殊的類別！

內容相當多而且複雜，其中的 … 表示很多所以省略

有關此程式可參考旗標出版的 Deep Learning 深度學習必讀一書第 2 章。

令人驚訝的是，train_image 不是表面上看到的那麼簡單。所以變數的結構是由等號右邊的敘述所決定，而變數本身只是一個 "標籤名字" 而已。