

# Matlab概論

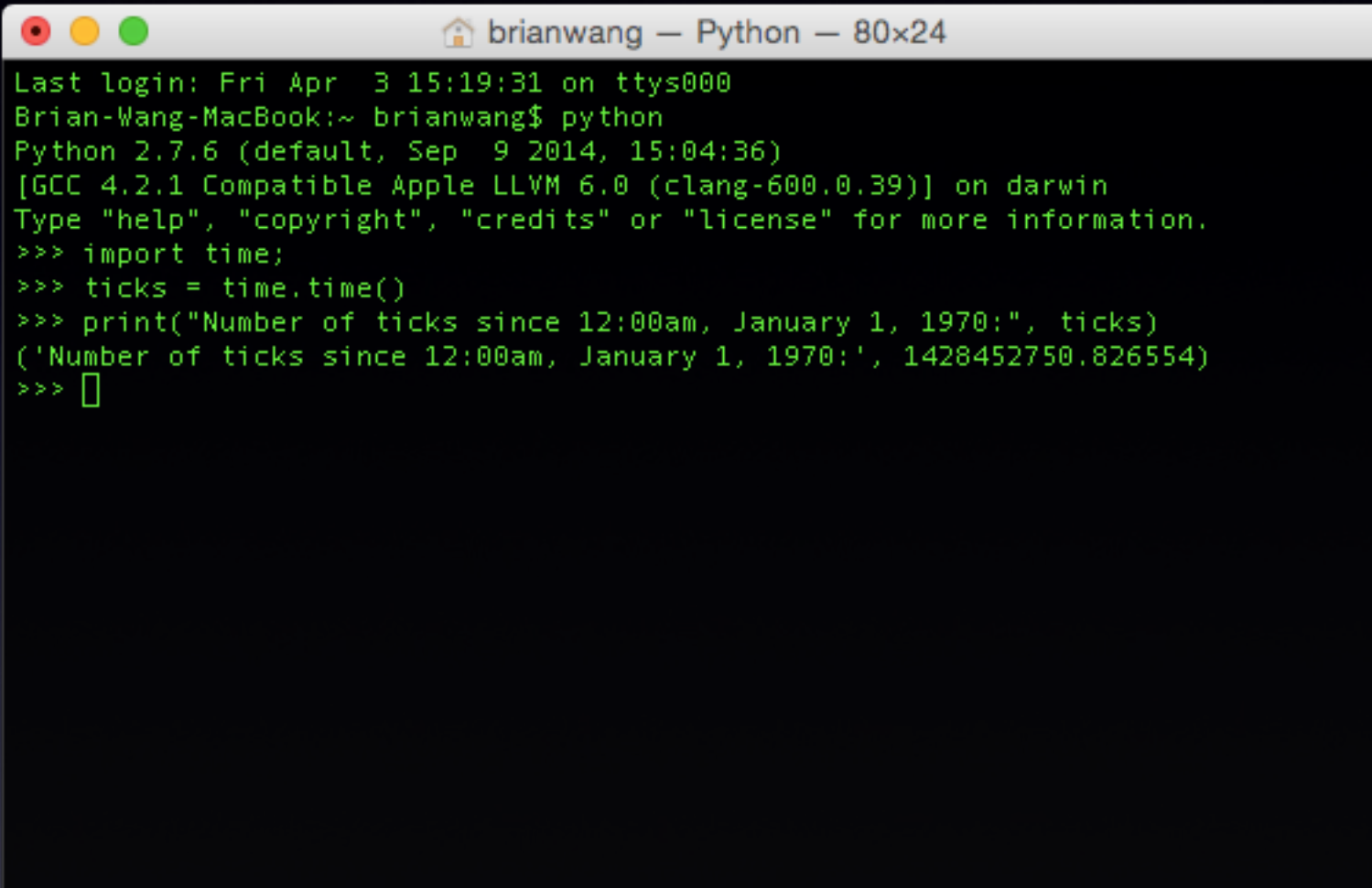
## Python 日期、函數

Brian Wang 王昱景  
[brian.wang.frontline@gmail.com](mailto:brian.wang.frontline@gmail.com)

# 時間模組

- 在撰寫程式時需要當下的時間，可以使用 Python 標準函數庫中所提供的時間模組
- Python 是以 tick 做為時間的計數單位
- Python 的時間準確度可以到百萬分之一秒，也就是一個 tick 時間
- tick 是以微秒為單位的 float 數值

- 1 秒 = 1,000,000 ticks
- 時間的算法是從 1970 年 1 月 1 日 0 點起算到現在總共經歷多少秒的時間
- 使用 time 模組的 `time.time()` 函數可以取得秒數，然後便能知道現在的時間
- 在此取得的 ticks 總數是依照格林威治標準時間來計算，並不是依照使用者所在地的時區計算



```
Last login: Fri Apr 3 15:19:31 on ttys000
Brian-Wang-MacBook:~ brianwang$ python
Python 2.7.6 (default, Sep  9 2014, 15:04:36)
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.39)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import time;
>>> ticks = time.time()
>>> print("Number of ticks since 12:00am, January 1, 1970:", ticks)
('Number of ticks since 12:00am, January 1, 1970:', 1428452750.826554)
>>> 
```



- time 模組的屬性包括：
  - timezone - 代表所在的時區跟格林威治標準時間的差異
  - 以秒數為單位，資料型態是 int
  - 以台灣為例，在 GMT+8 時區呼叫這個屬性會回傳 -28800

- altzone - 代表現在所在的時區跟格林威治標準時間的 DST（日光節約時間）的差異
- 以秒數為單位，資料型態是 int
- 以台灣為例，在 GMT+8 時區呼叫這個屬性會回傳 -32400

- daylight - 會回傳本地是否有使用日光節約時間
- 型態為 int，0 代表沒有，1 代表有

- time 模組的函數：
  - time() - 以 float 型態回傳從 1970 年 1 月 1 日 0 點到現在總共經過多少秒
  - sleep() - 接受 int 型態的參數，叫 Python 休息的函數
  - clock() - 回傳 float 值，第一次呼叫時會回傳此程式實際運行時間，第二次以後會回傳與第一次的時間差



- `gmtime()` - 可以選擇性給予 `int` 或 `float` 型態的參數
- 會回傳 1970 年 1 月 1 日 0 點開始經過所傳入秒數之後的時間，回傳的資料型態是 `tuple`
- `localtime()` - 和 `gmtime()` 相似，所傳回的 `tuple` 資料都會有相同時差的時數

- `asctime()` - 接受 `tuple` 型態的時間資料當作傳入參數，並將其轉換為字串形式回傳 `str` 值
- `ctime()` - 接受 `int` 或 `float` 型態的資料為傳入參數，會以 1970 年 1 月 1 日 0 點為起點開始加上傳入的秒數，並以 `str` 型態回傳加總之後的當地時間

- `mktime()` - 以具有時間之九個屬性值的 `tuple` 為傳入參數，並回傳 `float` 型態的數值
- `strftime()` - `strf` 是 `string format` 的簡稱，接收 `tuple` 型態的傳入參數，並可以依照所需的格式輸出
- `strptime()` - `strp` 是 `string parse` 的簡稱，接收一個字串參數與一個格式參數，然後會回傳時間的 `tuple`

# 函數

- 函數是經過組織且可重複使用的程式碼
- 函數 (function) 物件 (object) 可以執行一些工作，或是進行計算



- Python 中定義函數使用關鍵字 (keyword) `def`，其後空一格接函數的識別字 (identifier) 名稱加小括弧，然後冒號，如

```
def function_name():  
    pass
```

- `pass` 為關鍵字之一，其為甚麼事情都不做的陳述 (statement)

- 函數內容的部份須縮排 (indentation) ，縮排的區塊 (block) 專屬於函數，如下定義的 fun() 函數，印出遞增的數字

```
1  def fun():
2      i = 0
3      while i < 6:
4          print(i)
5          i += 1
6
7  fun()
8  fun()
9  fun()
```

- `fun()` 函數從 0 逐行遞增 1 印到 5，底下沒有縮排的地方一共呼叫 (call) `fun()` 三次，因此一連印出 18 個數字
- `i` 為 `fun()` 函數內的區域變數 (local variable)，`fun()` 之外的地方無法存取 `i` 的值，例如

```
1  def fun():
2      i = 0
3      while i < 6:
4          print(i)
5          i += 1
6
7  fun()
8  print(i)
```



- 這是因為直譯器 (interpreter) 在 fun() 函數的區塊內才認得變數 i，離開函數的地方，直譯器便不認識這個名稱
- 模組 (module) 也就是 .py 檔案中定義函數，需注意先有函數定義，才可進行函數呼叫。如將上例函數定義與呼叫的順序顛倒，如下

```
1 fun()  
2 fun()  
3 fun()  
4  
5 def fun():  
6     i = 0  
7     while i < 6:  
8         print(i)  
9         i += 1
```

- 因為 Python 直譯器從頭一行一行的解譯程式原始碼 (source code)
- `fun()` 既非 Python 的內建名稱，也還沒解譯到底下定義的部份
- 因此直譯器直接發生 `NameError`，終止程式的進行