

入門概論（三）：電腦系統與Python

1. 電腦系統

電腦系統的組成

電腦主要是由硬體與軟體兩部分，構成電腦系統架構，可以提供使用者在其中執行應用程式和處理資料。硬體是具體可見的零件，而軟體則是指揮硬體運作的靈魂。我們可以使用「輸入-處理-輸出」電腦系統處理程序模型，來說明電腦的工作方式。

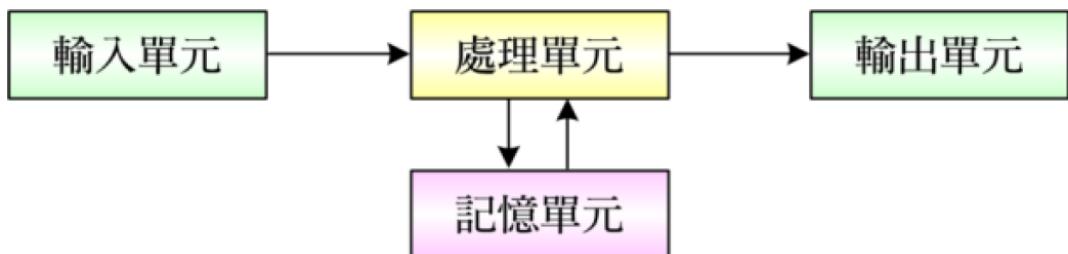
1-1 電腦硬體

電腦硬體是輸出入資料和處理資料的實體，負責執行解決問題所必須的基本運算和處理。

一、四個基本單位

電腦硬體的基本組成包括下列四個單元：

1. 輸入單元 (input unit)
2. 處理單元 (processing unit)
3. 記憶單元 (memory unit)
4. 輸出單元 (output unit)



1. 輸入單元(input unit)

用來接收外面的資料，包括文字、圖形、聲音與視訊，然後將這些資料轉換成電腦懂的格式，傳送給處理單元做運算。

例如鍵盤、滑鼠、觸控版、數位相機、數位攝影機、掃描器、搖桿。

- 鍵盤：是電腦主要的輸入設備，可以用來接收使用者的指令和外界的資料等，以做為電腦運作的依據。
- 滑鼠：滑鼠也是電腦主要的輸入設備，使用者只要移動滑鼠，螢幕上的指標就會移動到目標的文字或圖示，再透過滑鼠的按鍵，即可將訊息傳遞給電腦。



2. 處理單元 (processing unit)

「中央處理器」(CPU)，負責電腦的運算，其中包含控制單元和算術邏輯單元兩大部份。



3. 記憶單元(memory unit)

記憶單元是用來儲存CPU運算時所需要的資料或程式，以及儲存CPU的運算結果。記憶單元又分為記憶體和儲存裝置兩種類型。

- 記憶體用來暫時儲存資料，例如暫存器、快取記憶體、主記憶體等。
- 儲存裝置用來長時間儲存資料，例如硬碟、光碟、隨身身碟、記憶卡與固定硬碟。

～ 主記憶體中的資料一關機就會消失，如果要保留就要使用硬碟、隨身碟...等儲存裝置。



4. 輸出單元 (output unit)

將CPU運算完畢的資料轉換成使用者能夠理解的文字、圖形、聲音與視訊，然後顯示出來。

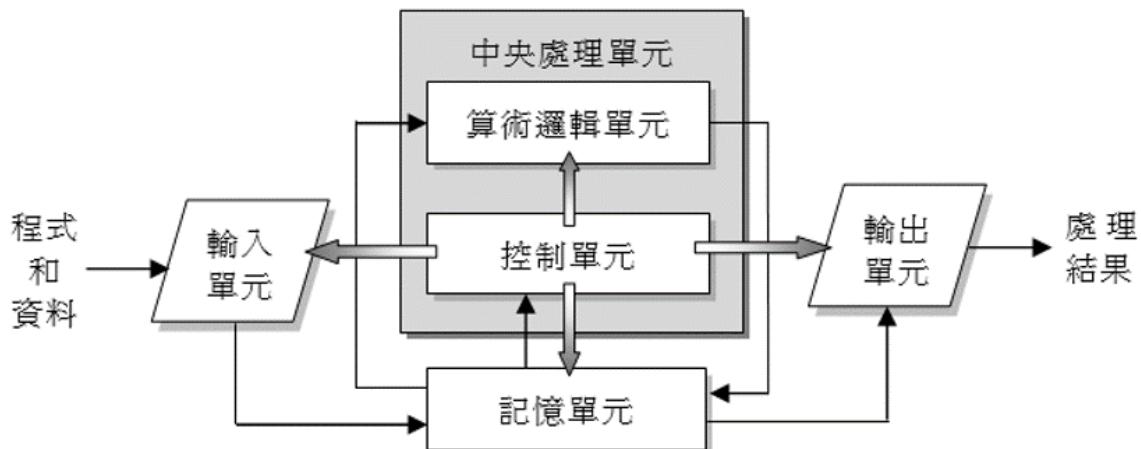
例如螢幕、印表機、喇叭、投影機等。

- 顯示器：又稱為「螢幕」，是電腦最主要的輸出設備，可以將電腦處理後的結果如文字或影像顯示出來。一般來說，螢幕都有畫面調整扭，可以讓使用者用來調整螢幕的亮度、大小、垂直及水平的中心點等。
- 鍵盤：是電腦主要的輸入設備，可以用來接收使用者的指令和外界的資料等，以做為電腦運作的依據。
- 滑鼠：滑鼠也是電腦主要的輸入設備，使用者只要移動滑鼠，螢幕上的指標就會移動到目標的文字或圖示，再透過滑鼠的按鍵，即可將訊息傳遞給電腦。
- 硬碟機：硬式磁碟機是電腦的工作磁碟，可以用來開機啟動作業系統，及存取主要的應用程式與使用者的資料。硬式磁碟機通常被電腦的主機外殼所蓋住，要拆開來才看的見。硬式磁碟機採用密閉的裝置設計，所以不容易受到污染。
- 光碟機：一般的光碟機，僅能讀取光碟片上的資料，而無法將資料再寫入，也有可讀寫光碟機常見的有MO和CD燒錄器。
- 印表機：常見的印表機種類有點矩陣式、噴墨式和雷射印表機等。印表機是將電腦上的文字和圖形，輸出列印到紙上的一項重要設備。
- 喇叭：喇叭是多媒體電腦中，不可或缺的一項重要設備，它可以將電腦的聲音或音樂效果輸出，讓使用者有身歷其境的感覺效果。



上述的電腦硬體又被稱為「主機」與「週邊設備」。

- 電腦的主機就像人類的大腦一樣，是電腦中最重要的部分。電腦的「主機」包含中央處理器、記憶體、及介面卡等，放在主機板上，而且外面以外殼保護著。
- 相對於電腦主機以外的硬體，都稱為電腦的「週邊設備」。常見的週邊設備有顯示器、鍵盤、滑鼠、磁碟機、光碟機、印表機、數據機等。



二、計算、儲存與連接三種功能

1. 計算用硬體：CPU、FPU、GPU



- 定義：負責計算的硬體元件



中央處理器 (CPU)
Central Processing Unit



浮點運算器 (FPU)
Floating Point Unit



圖形處理器 (GPU)
Graphic Processing Unit

(1) 中央處理器(Central Processing Unit , CPU)

中央處理器是電腦的主要裝置之一，主要功能是進行算術運算與邏輯運算，由控制單元（control unit）、算術邏輯單元（arithmetic logic unit）及部分記憶單元的暫存器（register）所組成。

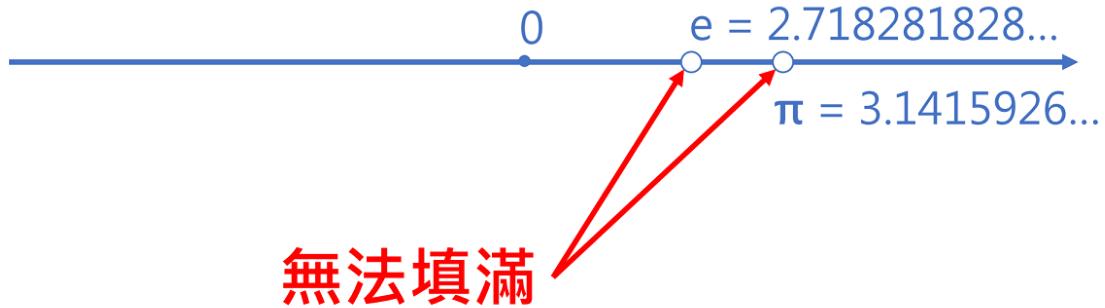
幾乎所有的CPU都使用二進位系統來表示數位。

- 位元（Bit），亦稱二進制位，指二進位中的一位，以0或1來表示，是資訊的最小單位。
- 將八個位元合成一個單元來表示單一數字或字母，稱為位元組（Byte）。
 $1\text{Byte} = 8\text{ Bits}$ ，一個位元組可表示256個符號， $2^8 = 256$ 種組合。英文以一個位元組來表示，中文字則以兩個位元組來表示。一個8位元的CPU表示一次可以處理八個二進位的資料。

平行處理：一部電腦裡面有多個處理器，每個處理器都像一個CPU，可以獨立執行工作，至於主記憶體及I/O 則共用。常見多核心中央處理器有四核心、八核心等。

(2) 浮點運算器 (Floating Point Unit , FPU)

浮點運算器是用來執行浮點運算。在現在的電腦架構中，浮點數運算跟整數運算是分開地，而浮點運算器大多被集成在CPU上，比較老的電腦硬體並不支援浮點數的運算。



• 沒有 FPU 幫忙



• 有 FPU 幫忙



(3) 圖形處理器 (Graphic Processing Unit , GPU)

圖形處理器是一種專門處理圖像運算工作的微處理器，能執行複雜的數學和幾何計算，擁有2D或3D圖形的加速功能，在浮點運算、平行計算上速度優越。有GPU，CPU就從圖形處理的任務中解放出來，可以執行其他更多的系統任務，大大地提高電腦的整體效能。

影片：[GPU v.s.CPU的速度比較](#)

2. 儲存用硬體

儲存體 = 儲存程式碼（指令 + 資料）

(1) 儲存體單位

Bit : 位元，0或1
 Byte : 位元組，8 Bits
 KB (Kilo Byte) : 千位元組，常用於表示檔案大小， $1KB=1024Bytes$ 。
 MB (Mega Byte) : 百萬位元組，常用於表示電腦主記憶容量量， $1MB=1024KB$ 。
 GB (Giga Byte) : 十億位元組，常用於表示硬碟容量量， $1GB=1024MB\ Bytes$ 。
 TB (Tera Byte) : 兆位元組， $1TB=1024GB$ 。
 PB (Peta Byte) : 千兆位元組， $1PB=1024TB$ 。

(2) 儲存體材料



A. 一級儲存體：速度快、關機資料會消失

靜態隨機存取記憶體 (Static Random Access Memory, SRAM) 動態隨機存取記憶體 (Dynamic Random Access Memory, DRAM)

B. 二級儲存體：速度慢、關機資料不消失

快閃記憶體 (Flash) USB 安全數位卡 (Secure Digital Memory Card，SD 卡) 固態硬碟 (Solid State Disk) 其他 機械硬碟 (Hard Disk) 光碟 (Optical Disk) 磁帶 (Tape)

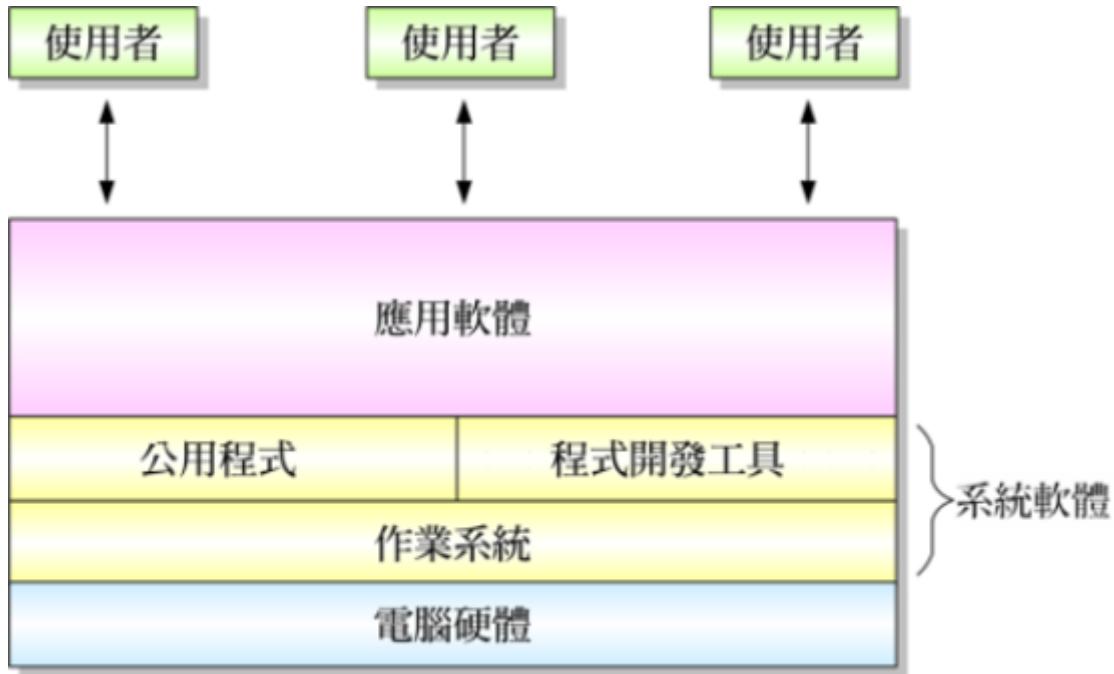
隨機存取記憶體 (Random Access Memory, RAM)，也叫主記憶體，是電腦內部最主要的記憶體，用來載入各式各樣的程式與資料，以供CPU直接執行與運用。它可以隨時讀寫，而且速度很快，通常作為作業系統或其他正在執行中的程式的臨時資料儲存媒介。

當使用硬體電源關閉後，儲存於記憶體中的資料會消失，因此，又稱為動態隨機存取記憶體。如果保持通電，記憶體儲存的資料可以恆常保持，就是靜態隨機存取記憶體。然而，當電力供應停止時，SRAM儲存的數據還是會消失，這與在斷電後還能儲存資料的ROM或快閃記憶體是不同的。在電源關閉後，原本寫入的資料仍可保存於快閃記憶體中，可以非常快速的存取資料。再加上小體積、大容量量的特性，快閃記憶體廣泛應用於許多可攜式的3C產品

3. 連接用硬體

匯流排(Bus) 電腦周邊設備

軟體是許多指令的集合，可以指揮電腦硬體運作，用來解決我們的問題，而這些指令的集合就是程式。一般將軟體分成系統軟體和應用軟體兩大類。



系統軟體 (system software)：管理電腦內各種硬體單元必備的程式，負責做硬體與軟體間溝通的橋樑，支援電腦運作的程式，包括作業系統 (operating system)、公用程式 (utility)、程式開發工具 (program development tool)。

- 作業系統是介於電腦硬體與應用軟體之間的程式，除了提供執行應用軟體的環境，還負責分配系統資源。
- 公用程式是用來管理電腦資源的程式。
- 程式開發工具是協助程式設計人員開發應用軟體的工具，包括文字編輯器等。

應用軟體 (application software): 針對特定事務或工作所撰寫的程式，是因應用戶需求而設計的程式，來完成特定的工作，目的是協助使用解決問題。



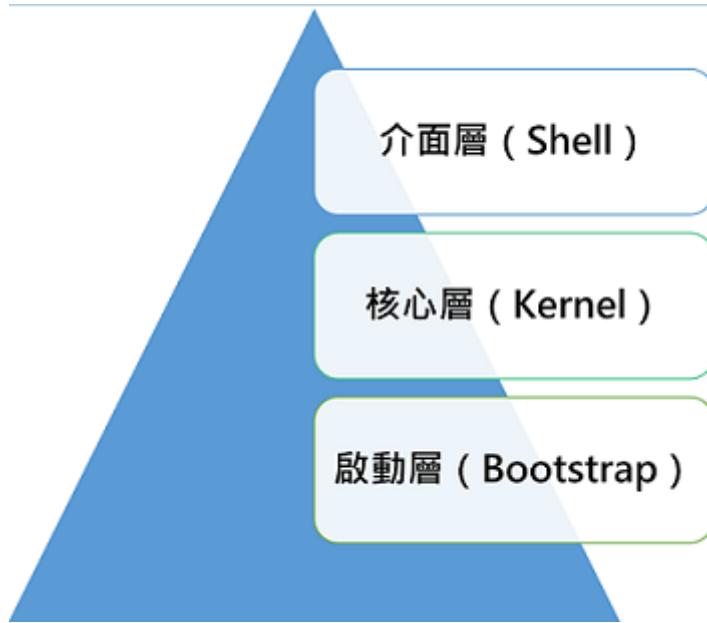
Windows 作業系統屬於系統軟體



Word 文書處理軟體屬於應用軟體

作業系統

作業系統是管理電腦硬體與軟體資源的系統軟體，同時也是電腦系統的核心與基石。作業系統層分為三層：

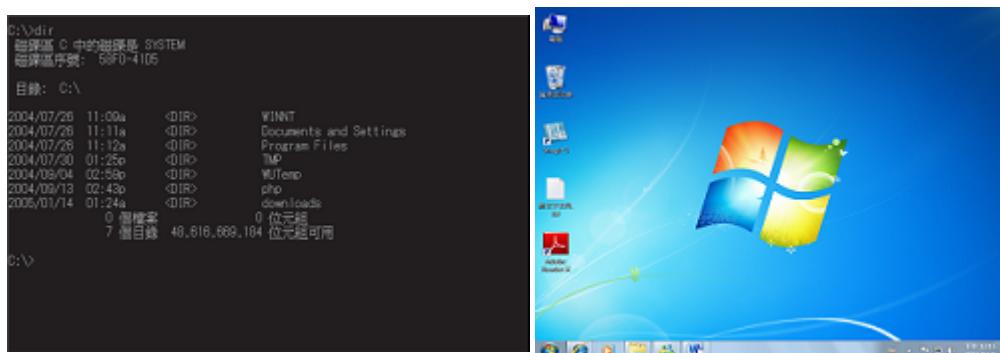


(1) 啟動層Bootstrap = 火星塞 開機程式是系統啟動後的第一個執行的程式，可將硬體（CPU與記憶體）初始化，讀進作業系統，直到可以迎接核心層的程度。

(2) 核心層Kernel 核心是一個電腦程式，進行的是應用軟體和電腦硬體的互動工作。核心層下管硬體層，任何程式要用硬體都得跟核心層申請。核心層上管程式層，負責監督、協調與控制作業系統中的處理程式，讓所有執行中的程式和平相處正常運作。

(3) 介面層Shell 一個接受並處理使用者的指令，讓使用者與系統互動的操作介面。

- 命令列介面(CLI, command-line interface)：用鍵盤在命令列上鍵入文字指令。
- 圖形化介面 (GUI, graphical user interface)：用滑鼠按下一圖形鈕。



常見的作業系統

UNIX電腦作業系統是由美國AT&T公司的貝爾實驗室，在1969年開發成功的，具有多工、多用戶的特徵。所有作業系統或多或少衍伸自Unix系統。例如最受歡迎的Unix Shell是Bash (the Bourne Again Shell)，可在Linux系統上使用；Apple電腦的OSX系統為BSD (Berkeley Software Distribution)，是目前商業化最成功的Unix作業系統；Window電腦則是採Unix-like的Power Shell。

(1) Linux系統

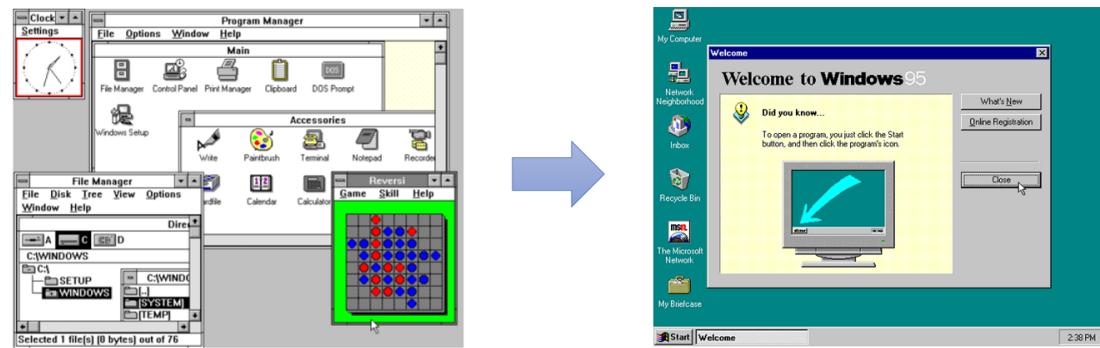
Linux是由芬蘭大學生林納斯·托華斯 (Linus Torvalds)，於1991年以UNIX為基礎，開發出來安裝在個人電腦上的作業系統。近年來開放原始碼系的Linux越來越受歡迎，是市佔率最高的系統。

(2) Windows系統

MS-DOS (Microsoft disk operating system)採用命令列使用者介面，是Microsoft公司於1975年針對IBM PC所推出的作業系統，使用者必須透過鍵盤輸入指定的指令集，才能指揮電腦完成工作，為Unix的仿作。



MS-Windows (Microsoft Windows)則是微軟在MS-DOS基礎上設計的圖形作業系統。1985年、1987年所推出的Windows 1.0和Windows 2.0順利轉型成為IBM兼容PC的標準圖形使用者介面系統，1990年推出的Windows 3.0獲得了空前迴響。



1990 MS-Windows 3.0

1995 MS-Windows 95

1995年推出的Windows 95不再包含MS-DOS，Windows從殼層程式轉變為真正的作業系統。之後Microsoft公司不斷推出新版的Windows作業系統，包括Windows Me、Windows XP、Windows Vista 和Windows 7/8/10。現在的Windows系統皆是建立於 Windows NT核心，可以在32位元和64位元的Intel和AMD的處理器上運行。

(3) MAC系統

macOS，稱「Mac OS X」或「OS X」，是一套執行於蘋果Macintosh系列電腦上的作業系統。Apple公司的創始人Steve Jobs意識到圖形使用者介面的重要性及未來來前景，於1983年、1984年推出採用圖形化使用者介面的個人電腦Macintosh(麥金塔)。Mac OS指的就是安裝於Apple Macintosh電腦的作業系統，以卡內基美隆大學開發的 Mach做為核心，是第一個商用化成功的圖形化使用者介面系統，最終版本為1999年推出的Mac OS 9。之後Apple公司改以BSD UNIX為基礎推出OS X，從OS X 10.8開始在名字中去掉Mac，僅保留OSX和版本號。2016年年蘋果公司將OS X更更名為 macOS，現行的最新的系統版本是macOS Mojave。



2. Python 簡介

2-1 為什麼學Python

～Python 成為主流程式語言

IEEE Spectrum 年度程式語言排行榜, Python 禪連衛冕.

Language Rank	Types	Spectrum Ranking
1. Python	🌐💻📱	100.0
2. C++	📱💻📱	98.4
3. C	📱💻📱	98.2
4. Java	🌐📱💻	97.5
5. C#	🌐📱💻	89.8
6. PHP	🌐	85.4
7. R	💻	83.3
8. JavaScript	🌐📱	82.8
9. Go	🌐💻	76.7
10. Assembly	💻	74.5

2-2. 認識Python



- 發明人
 - 吉多·范·羅蘇姆
(Guido van Rossum)
 - 荷蘭人
- 緣由
 - 1989年12月
 - 無聊打發時間
 - 想做個簡單好學又強大的語言

1989年 Guido Van Rossum為了打發聖誕節的空餘時間，就開發了Python 程式語言。



- 喜劇
 - 蒙提·派森的飛行馬戲團
 - 1969 ~ 1974 · 共 45 集
- 和蟒蛇沒啥關係
- 作者說：你們要認為它是蟒蛇也無所謂！

因為吉多是BBC電視劇——蒙提·派森的飛行馬戲團（Monty Python's Flying Circus）的愛好者，所以選擇Python當作這個程式語言的名字。

- 2000年10月16日發布Python 2.0
 - Python 2.0只到Python 2.7, 2020年退休
- 2008年12月 日發布Python 3.0
 - 此版不完全相容於Python 2.0
 - 所有新功能都加入到Python 3.0以後的版本

Guido van Rossum曾為不同的公司工作，如Google、Dropbox等，也因此變成許多著名公司的核心程式語言。

2-3 Python 的特性

- 簡單易學：Python 的哲學是「優雅」、「明確」和「簡單」，語法是接近日常語言的高階語言。
- 具備腳本和程式語言功能：Python語言具備腳本語言功能，可以執行電腦簡單自動化的任務。也可以編寫功能繁複的程式。
- 開源又免費：Python可以免費複製、散佈，原始碼可以自由閱讀甚至修改。Python基於群體知識分享的理念，所以程式會不斷地改進不停地優化。
- 可攜性高：Python是開放源碼軟體，所以已經被修改成能夠在不同平臺順利運作。Python語言開發的程式碼，可以無需修改就能在各種平臺上面運行。
- 良好的移植性：Python語言是採用直譯器，程式碼會先轉換成位元組碼的中間形式，然後再根據平台翻譯成機器語言並運行。
- 膠水語言：Python語言可以將不同功能的程式碼，甚至不同語言的程式碼，結合起來一起執行。
- 可嵌入性：使用Python語言編寫的程式碼，可以嵌入到其他語言的程式中，使該程式也具有腳本的功能。

- 功能強大的函式庫：Python提供強大的標準函式庫，可以協助處理GUI、檔案存取...等各種工作。除標準函式庫外，還有許多功能包羅萬象的第三方函式庫。
- 支援物件導向：Python語言也完整支援物件導向，且秉持簡約的精神，讓設計者可以用簡單的語法撰寫物件導向程式。。

2-4 學Python可以做什麼

Python的用途很多很廣，可用於字串處理、數學運算、科學計算、系統管理、網頁框架、大數據分析與網頁分析等。

總而言之，在此時代變動的趨勢潮流下，要學程式語言就從簡單、重要、用途多的**Python**學起吧！

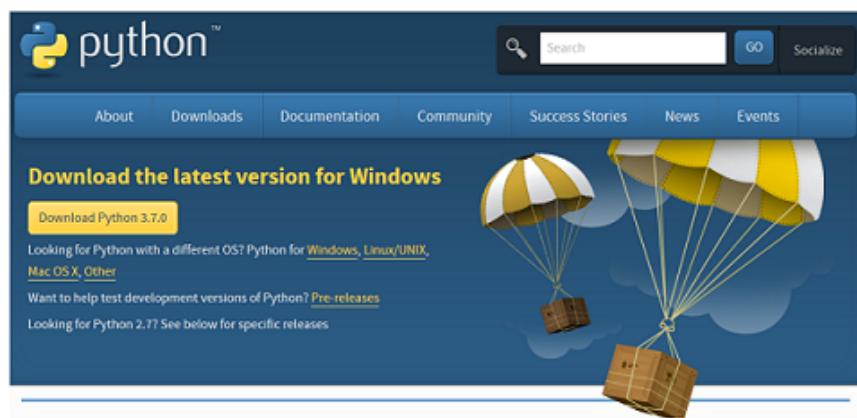
2. Python 官方安裝與使用

2-1 下載與安裝

下載Python 3檔案：<https://www.python.org/>



- <https://www.python.org/downloads/>



然後開始安裝（記得Add Python to PATH要打勾）

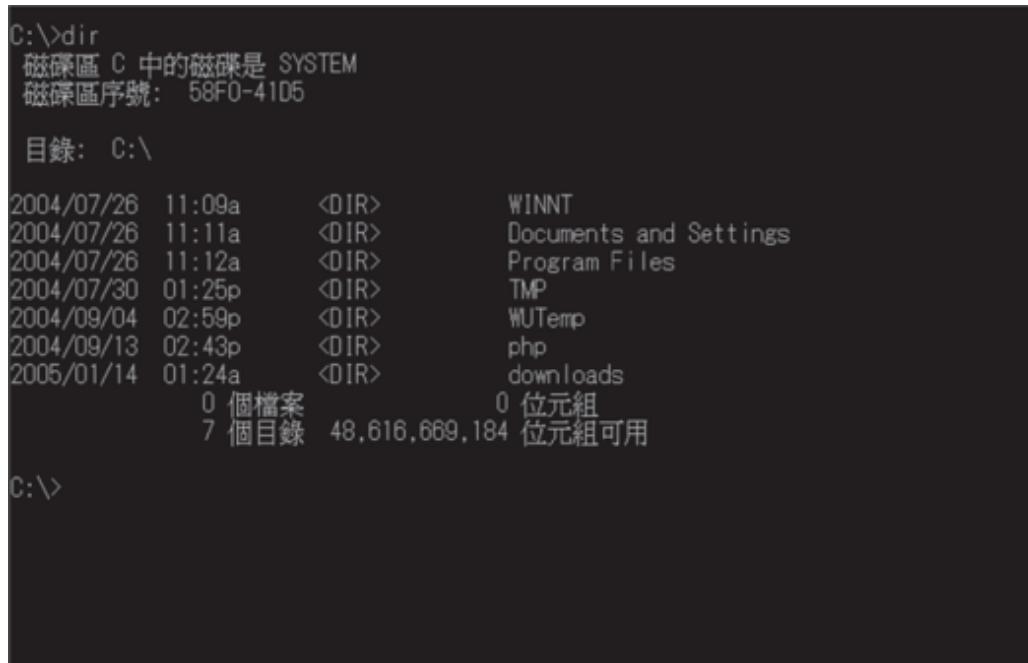
參考資料：網路上安裝教學(約5分鐘)

<https://www.youtube.com/watch?v=pAY3q4KZ-z4>

操作介面

一個接受並處理使用者的指令，讓使用者與系統互動的操作介面。

- 命令列使用者介面(CLI , command-line interface)：用鍵盤在命令提示列上鍵入文字指令。



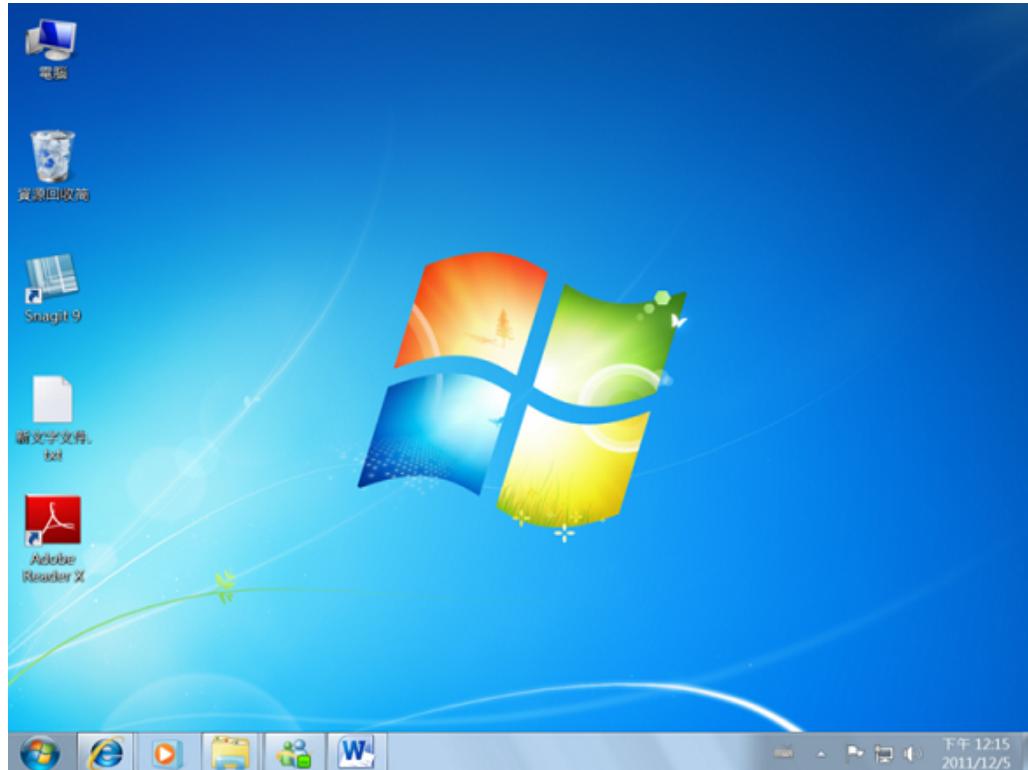
```
C:\>dir
磁碟區 C 中的磁碟是 SYSTEM
磁碟區序號: 58F0-41D5

目錄: C:\

2004/07/26 11:09a    <DIR>        WINNT
2004/07/26 11:11a    <DIR>        Documents and Settings
2004/07/26 11:12a    <DIR>        Program Files
2004/07/30 01:25p    <DIR>        TMP
2004/09/04 02:59p    <DIR>        %UTemp
2004/09/13 02:43p    <DIR>        php
2005/01/14 01:24a    <DIR>        downloads
                           0 個檔案          0 位元組
                           7 個目錄   48,616,669,184 位元組可用

C:\>
```

- 圖形化使用者介面 (GUI , graphical user interface)：用滑鼠按下一按圖形鈕。



2-2 使用模式

兩種常見的使用模式：互動模式與正規模式

使用模式（一）：互動模式（Interactive Mode）

直接在命令列執行程式指令，然後觀看結果。

- (1) Python互動模式

打 **python** 進入 python 互動模式，電腦出現 **>>>** 符號。在 **>>>** 符號後面打入 python 相關指令，電腦就會執行程式、顯示結果。

```
1 $ python
2
3 Python 3.5.2 |Anaconda 4.1.1 (x86_64)| (default, Jul  2
2016, 17:52:12)
4 [GCC 4.2.1 Compatible Apple LLVM 4.2 (clang-425.0.28)] on
darwin
5 Type "help", "copyright", "credits" or "license" for more
information.
6 >>>
```

- (2) Ipython互動模式

IPython 是一種基於 Python 的互動式直譯器。相較於原生的 Python Shell，IPython 提供了更為強大的編輯和互動功能。

打 **ipython** 進入 ipython 互動模式，電腦出現 **In[1]** 符號。在 **In[1]** 符號後面打入 python 相關指令，電腦就會執行程式、顯示結果。

```
1 Python 3.6.8 |Anaconda custom (64-bit)| (default, Dec 30
2018, 18:50:55) [MSC v.1915 64 bit (AMD64)]
2 Type 'copyright', 'credits' or 'license' for more
information
3 IPython 7.2.0 -- An enhanced Interactive Python. Type '?' for help.
4
5 In [1]:
```

～Python 直譯器有 python 與 ipython 兩種不同設計；有 Python 2 與 Python 3 兩種不同的版本；也有不同電腦作業系統的差異，如 windows 與 MAC；還有電腦作業系統版本的差異，如 32bit 與 64bit 的不同。

使用模式（二）：正規模式（Normal Mode）

如果要執行較長的指令，或者有重複執行的可能性時，可以撰寫並儲存程式，此稱為**script**（腳本）。Python腳本的附檔名是".py"。

用文字編輯器，將下面指令儲存在test.py中。

執行時，在提示符號下，鍵入 **python 檔案名稱.py** 即可執行程式。

```
1 $ python test.py
```

2-3 整合開發環境

寫電腦程式需要在有文字編輯器、直譯器與除錯器的環境下進行。

2-3-1 什麼是整合開發環境

整合開發環境（Integrated Development Environment，簡稱IDE）是將撰寫、編譯、除錯、執行等功能合併在一套軟體，讓你在編寫程式時更快速方便。

(1) 寫出原始碼(Source Code)

原始碼（Source code），是指一系列列人類可讀的電腦語言指令。為了編譯出電腦程式，要用好的文字編輯器紀錄下來。

～常用的文字編輯器有：

NotePad++是一套用來取代 Windows 記事本的工具，操作方式與記事本幾乎相同，還可以替許多的程式語言的語法上色，是撰寫程式的好幫手，並支援 Unicode。

Atom是由GitHub開發的自由及開放原始碼的文字與程式碼編輯器，支援macOS、Windows和Linux作業系統，並內建由Github提供的Git版本控制系統，可透過套件（Package）擴充功能。Atom可當作IDE使用。被它的開發者稱為「21世紀的「駭客」文字編輯器（hackable text editor for the 21st Century）」。

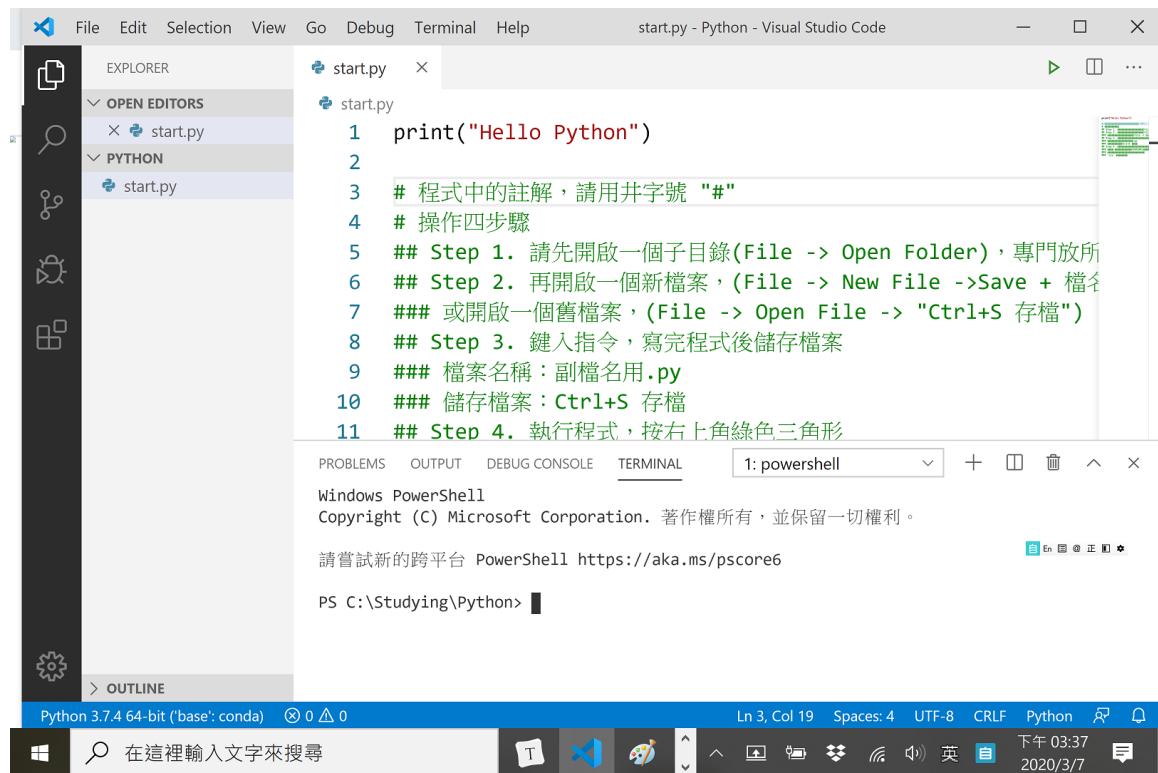
Sublime是一套跨平台的文字編輯器，支援macOS、Windows和Linux作業系統，可透過套件（Package）擴充功能。這門課如果需要文字編輯器，將拿Sublime示範教學、撰寫程式。

Visual Studio Code（簡稱VS Code）是一個由微軟開發，同時支援Windows、Linux和macOS等操作系統且開放原始碼的程式碼編輯器，它支援測試，並內建了Git 版本控制功能，同時也具有開發環境功能，例如代碼補全、代碼片段和代碼重構等。該編輯器支援用戶個性化組態，例如改變主題顏色、鍵盤捷徑等各種屬性和

參數，同時還在編輯器中內建了擴充程式管理的功能。

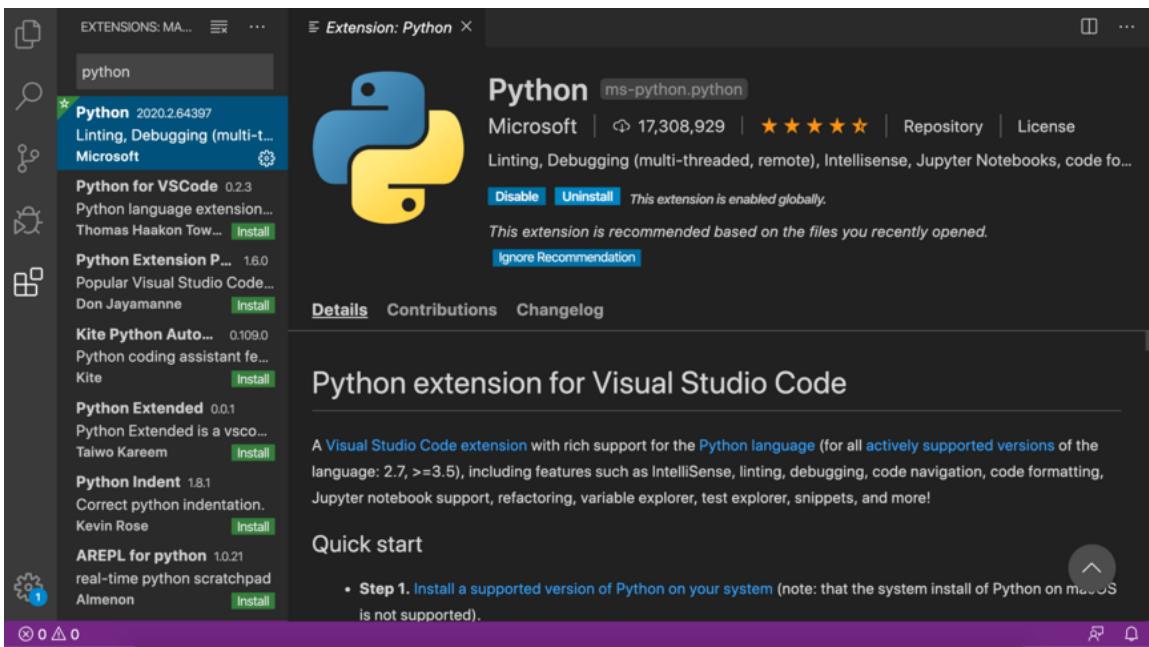
在2019年的Stack Overflow組織的開發者調研中，VS Code被認為是最受開發者歡迎的開發環境，據調查87317名受訪者中有50.7%的受訪者聲稱正在使用VS Code。

VS Code操作四步驟



- Step 1. 請先開啟一個子目錄(File -> Open Folder)，專門放所有檔案
- Step 2. 再開啟一個新檔案，(File -> New File -> Save + 檔名.py)
 - 或開啟一個舊檔案，(File -> Open File -> "Ctrl+S 存檔")
- Step 3. 鍵入指令，寫完程式後儲存檔案
 - 檔案名稱：副檔名用.py
 - 儲存檔案：Ctrl+S 存檔
- Step 4. 執行程式，按右上角綠色三角形
 - 指令:在下面區塊，TERMINAL選項，輸入 **python** 檔案名稱
 - 方向鍵可以記憶使用過的指令
 - **cls** 清空畫面

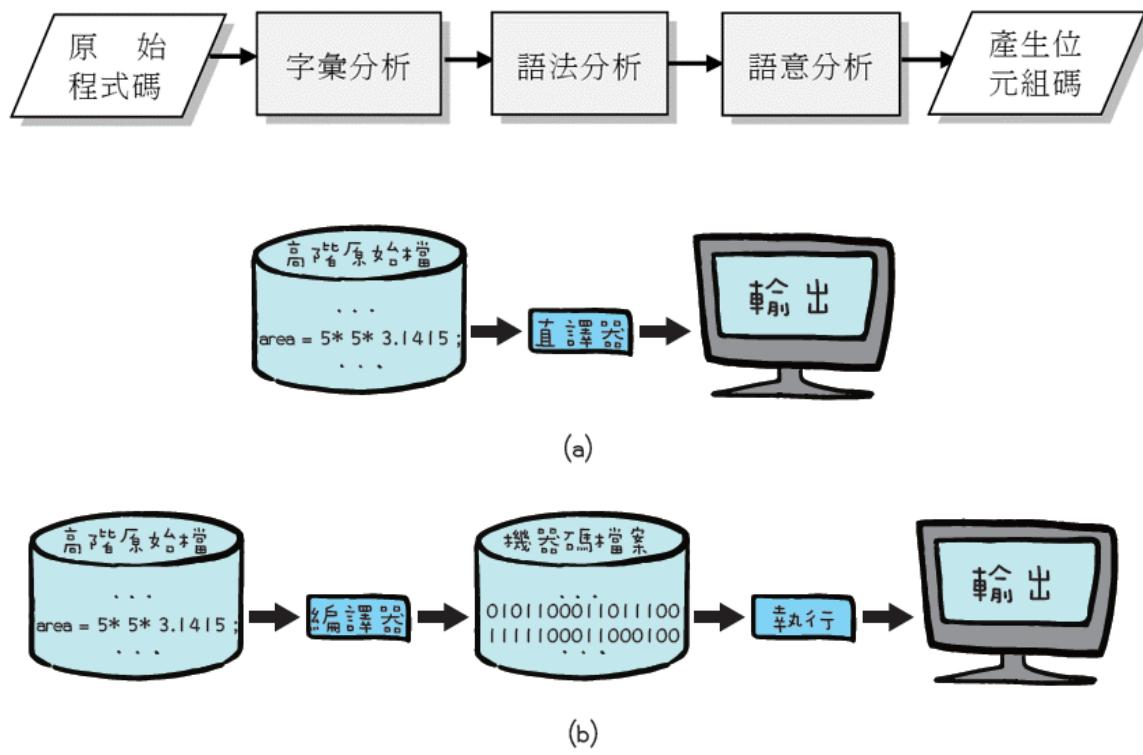
使用Visual Studio Code要記得安裝外掛Python。



利用文字編輯器撰寫Python程式，完成的檔案稱為原始程式檔，Python的程式碼檔案的副檔名為.py。

(2) 執行原始碼

電腦執行程式時，會先由輸入設備將撰寫好的原始程式碼(Source Code)載入到記憶體，程式碼要經過編譯成機器語言才能在電腦中執行。在編譯的過程中，編譯器會對原始程式進行字彙分析、語法分析和語意分析，順利完成後會產生位元組碼。如果編譯時檢查出錯誤，編譯器就會停止編譯。程式設計者必須將產生錯誤的程式碼更正，再重新編譯直到沒有錯誤為止，才能順利執行程式。



● 圖1-2 (a)直譯器一次翻譯與執行一行程式敘述。
(b)編譯器將整個原始程式翻譯成機器語言檔

高階語言的語言翻譯器，有直譯器和編譯器兩種。

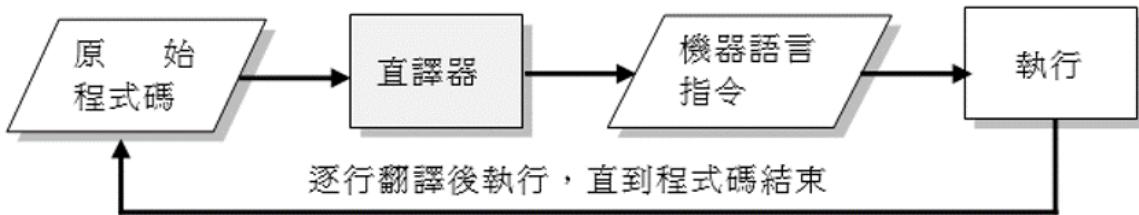
表6-a | 直譯器及編譯器

分類	說明
直譯器 (Interpreter)	<ol style="list-style-type: none">將高階語言逐行翻譯，每翻譯一行敘述後便直接執行，執行完後再翻譯下一行敘述並執行。每次執行時都必須執行翻譯的動作。執行效率較差。 <p>The diagram illustrates the interpreter process. A robot character is shown reading a line of high-level language code: "SUM = SUM + S1". This code is then translated into binary machine code: "10111001 01100110 00101000 10001001 01010110 00...". The machine code is processed by a yellow box labeled "機器碼" (Machine Code), which then leads to a yellow box labeled "執行" (Execution). A feedback loop labeled "處理下一敘述" (Handle next statement) returns from the execution back to the "單一敘述" (Single statement) input.</p>
編譯器 (Compiler)	<ol style="list-style-type: none">將高階語言所寫的原始程式整個轉譯成目的碼(Object Code)。只需將程式編譯過一次，以後便可以直接拿其目的程式碼來使用。執行效率較佳。 <p>The diagram illustrates the compilation process. A robot character is shown reading a high-level language program:<pre>Program Sum(Input, Output); Var N1,N2,SUM:Integer; Begin ReadIn(Input, N1, N2); Sum := N1 + N2; WriteIn(Output,Sum); End.</pre>This program is then translated into binary object code: "10111001 01100110 00101000 10001001 01010110 00...". The object code is processed by a yellow box labeled "目的碼" (Object Code), which then leads to a yellow box labeled "可執行檔" (Executable File).</p>

A. 直譯器 (interpreter)

直譯器的功能是將程式語言所編寫的程式碼，依照程式的邏輯順序，將指令逐行轉為機器語言指令，並且立即執行。程式語言中Python、BASIC、JavaScript、LISP...等，都是使用直譯方式來翻譯程式。直譯器的優點是執行時需要的記憶體較小，存檔時占用較小的磁碟空間。適合開發階段的除錯，方便初學者學習和測試。但是，缺點是每次執行程式都必須重新翻譯，所以執行所需的時間較長效率較差。

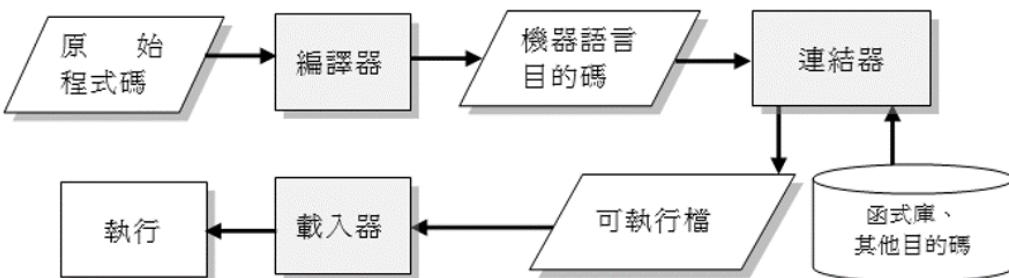
- 使用直譯器時，程式語言變成了一個會和你互動的環境。
 - 在命令列上直接打指令時，直譯器會立刻執行該命令。
 - 把所有指令統統輸入到某個檔案裡面，然後呼叫直譯器讀取該檔案，並執行檔案中的指令亦然。
- 如果所下的指令有錯誤產生，直譯器會進入偵錯模式 (debugger)，並且顯示相關錯誤訊息，以便對程式除錯。
 - 可以立刻看到指令的執行結果，並迅速修正錯誤。
 - 直譯器會用到很多的記憶體，執行效率比編譯器差。



B. 編譯器 (compiler)

編譯器的功能是將高階語言所寫的程式碼，一次翻譯成能直接被電腦接受的機器語言。程式經過編譯後產生的目的碼，可以透過連結器，連結函式庫等相關檔案後產生可執行檔。執行程式時由作業系統的載入器，將可執行檔載入記憶體就能執行。使用編譯器的優點是程式執行時不用再翻譯，所以執行的速度較快。缺點是編譯和連結所需要的時間較長，而且程式若有修改就必須再重新編譯一次。

- 使用編譯器時，必須先把程式碼統統寫到檔案裡面，然後執行編譯器來編譯程式。
 - 在命令列可以執行編譯好的程式，看看它是否可以運作。
 - 如果不行就必須修改程式，直到編譯器接受且把你的程式編譯成執行檔為止。
- 編譯器與直譯器的差別
 - 編譯器不像直譯器可以馬上得到結果。
 - 編譯器執行效率高，在編譯時順便最佳化程式，直譯器不行。
 - 當你想把你寫好的程式拿到另外一台機器上跑時，你只要將編譯器編譯出來的可執行檔，拿到新機器上便可以執行，而直譯器則必須在新機器上重新執行一次。

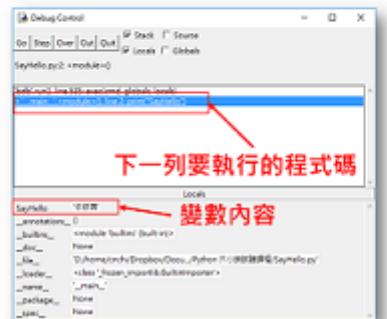


(3) 除錯器

找出錯誤在哪一列

解說方法

• 用「除錯器 (Debugger)」找出錯在哪一行



~ Summary:

當我們有整合開發環境完整的輔助工具時，程式設計就會變得輕鬆些。

會用到的開發工具



當你要寫程式讓電腦執行時,你需要文字編輯器撰寫程式。寫完的程式先讓電腦跑跑看有沒有錯誤,這時你需要翻譯器。沒有錯誤才算寫好程式。不幸的是我們常常寫錯,所以還需要除錯器來幫忙修正。



每個工具的用途



純文字編輯器 (Text Editor)

- 儲存時，除了文字本身，不存任何格式設定的軟體
- 如：NotePad (Word 就不是純文字編輯器)



直譯器 / 編譯器 (Interpreter / Compiler)

- 直譯器：一次翻譯一列成為 0/1 的軟體
- 編譯器：一次翻譯一個檔案成為 0/1 的軟體



除錯器 (Debugger)

- 一次執行一列，讓你慢慢找到底程式寫錯在哪一列的軟體
- 如果程式一次就寫對，那麼就不需要執行除錯這個步驟

另外，整合開發環境和你用哪一種高階語言有關，例如在Rstudio整合開發環境寫R程式語言(用Rstudio學R)，在Jupyter Notebook整合開發環境寫Python程式語言(用Jupyter Notebook學Python、用Pycharm學Python)等等。

2-3-2 常見的IDE

- (1) IDLE
- (2) Spyder
- (3) Pycharm
- (4) Thonny (功能簡單好上手的IDE)

～上面的這些IDE僅儲存程式，副檔名為.py，任何文字編輯器都能開啟。

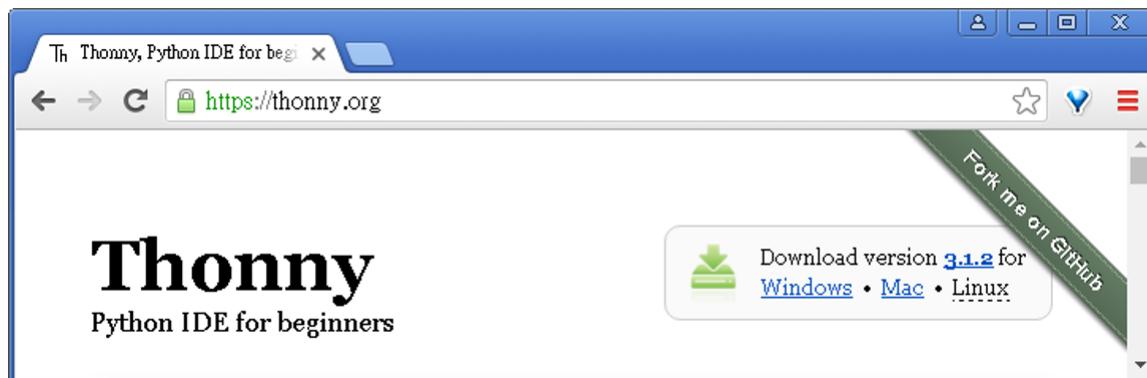
～下面的Jupyter Notebook IDE同時儲存程式與執行結果在一個檔案中，副檔名為.ipnb。得用Jupyter Notebook才能開啟、github可以直接瀏覽，但因為使用網頁，有些特殊功會無法使用。

3. Thonny 使用

運算思維與程式設計課程使用此IDE。

3-1 安裝 Thonny 開發環境

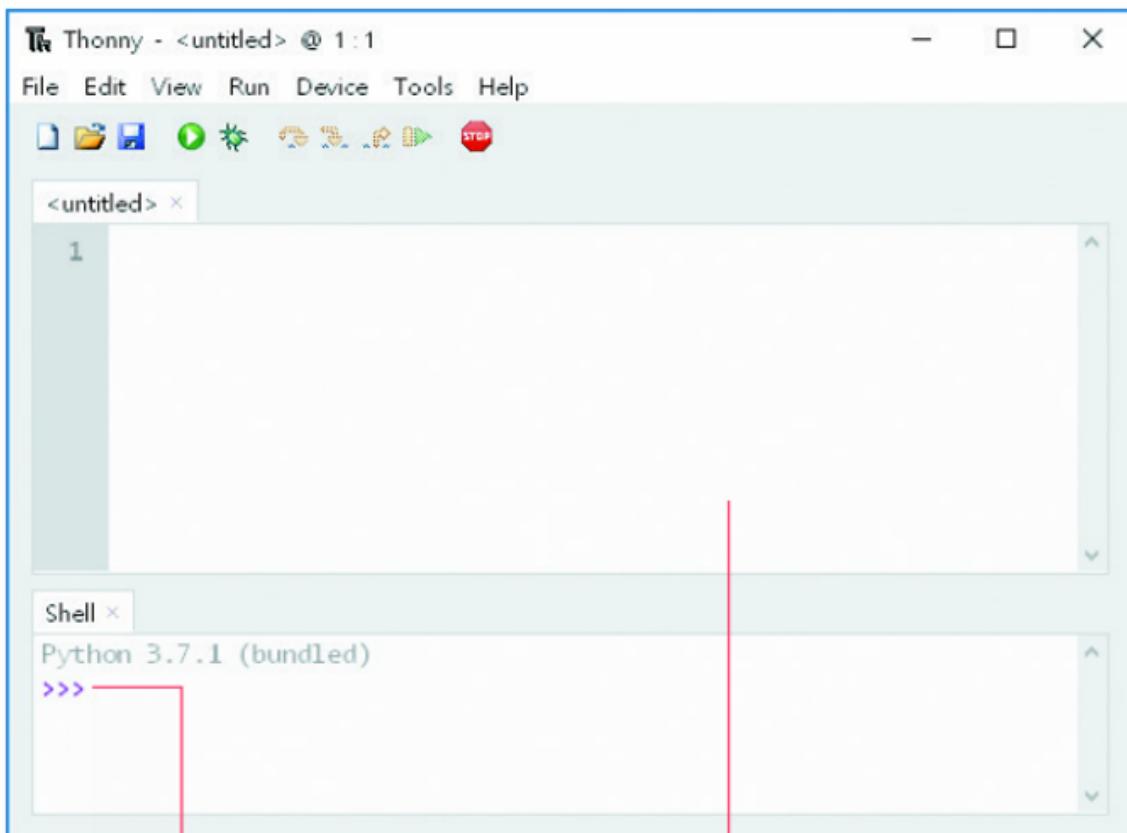
Thonny 下載網址：



下載後請雙按執行該檔案，然後一直按 Next 即可完成安裝。



3-2 開始寫第一行程式



互動性程式執行區

程式編輯區

The image shows two screenshots of a Python shell window. In the first screenshot, the command `>>> print("Hello World")` is entered. A callout bubble explains that this program tells the computer to print "Hello World". In the second screenshot, the output `Hello World` is displayed, with a callout bubble stating that the computer follows the program's instructions to show "Hello World".

1 輸入 `print("Hello World")`,
然後按 `Enter` 鍵

2 電腦依照我們的程
式顯示 `Hello World`

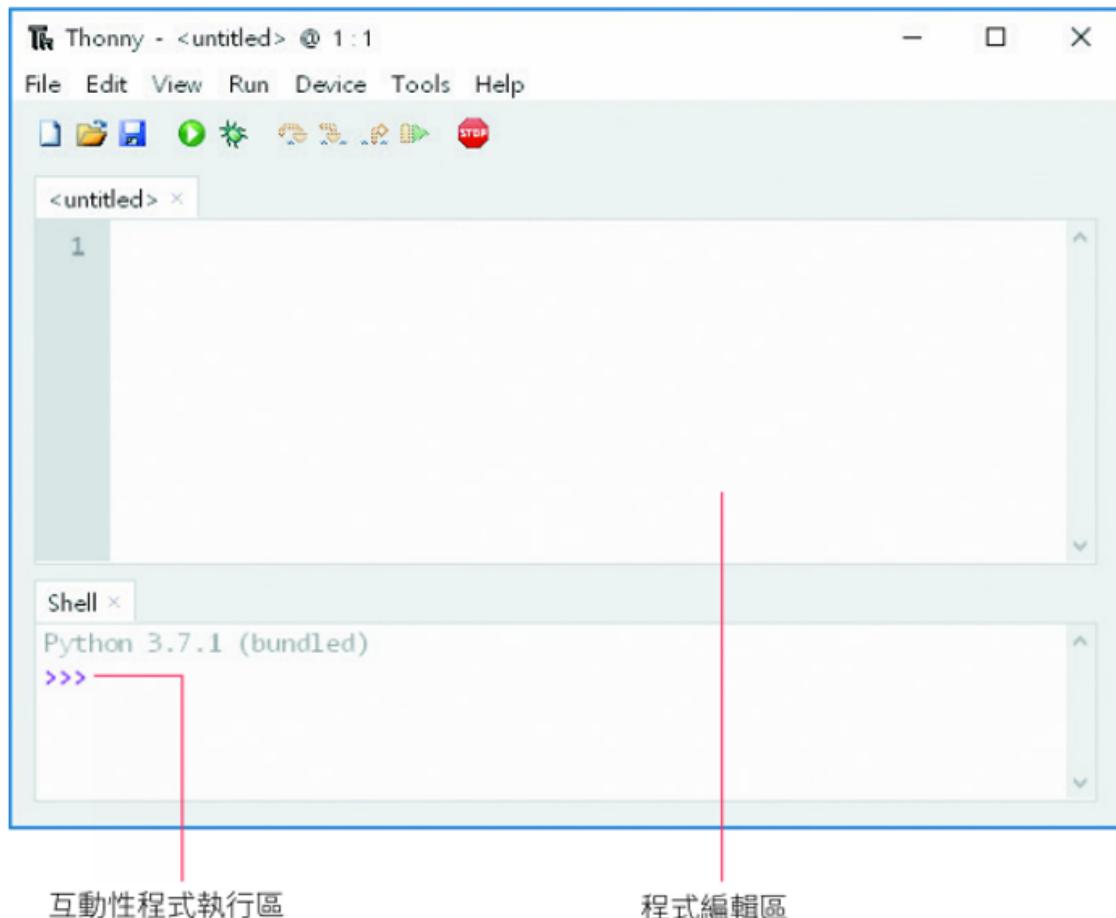
寫程式其實就像是寫劇本，寫劇本是用來要求演員如何表演，而寫程式則是用來控制電腦如何動作。



雖然說寫程式可以控制電腦，但是這個控制卻不像是人與人之間溝通那樣，只要簡單一個指令，對方就知道如何執行。您可以將電腦想像成一個動作超快，但是什麼都不懂的小朋友，當您想要電腦小朋友完成某件事情，例如唱一首歌，您需要告訴他這首歌每一個音是什麼、拍子多長才行。

所以寫程式的時候，我們需要將每一個步驟都寫下來，這樣電腦才能依照這個程式來完成您想要做的事情。

3-3 Thonny 開發環境基本操作



互動性程式執行區

程式編輯區

The screenshot shows the Thonny IDE interface with a program example. The program code is:import time
from machine import Pin

led = Pin(15, Pin.OUT)
led.value(1)
time.sleep(3)
led.value(0)A red bracket on the right points to the 'Interactive Interpreter' area, which displays the output of the program: 'Hello World'. The code editor shows the same code with syntax highlighting.

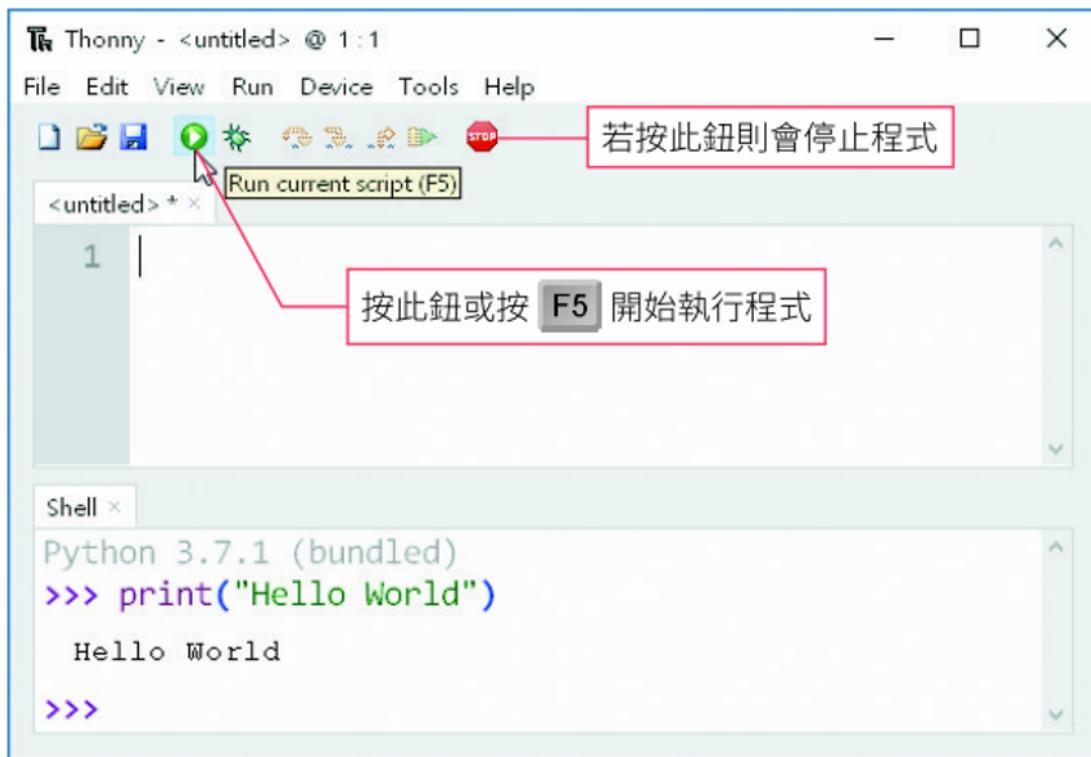
可以說，上半部程式編輯區類似稿紙，讓我們將想要電腦做的指令全部寫下來，寫完後交給電腦執行，一次做完所有指令。

而下半部 **Shell** 窗格則是一個交談的介面，我們寫下一行指令後，電腦就會立刻執行這個指令，類似老師下一個口令學生做一個動作一樣。

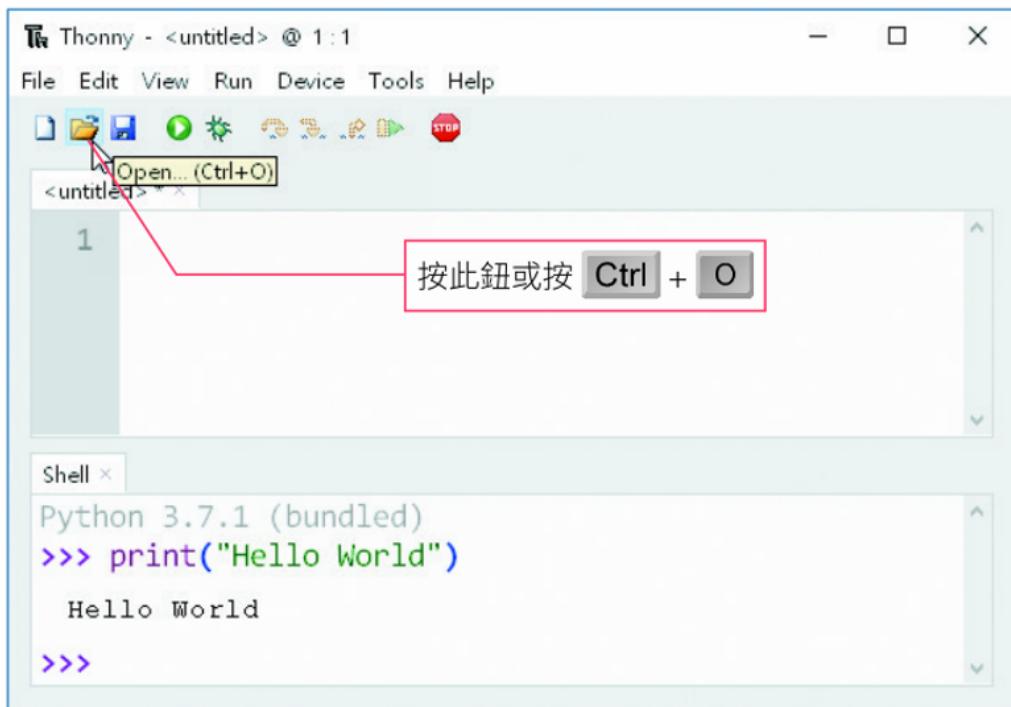
所以 **Shell** 窗格適合用來作為程式測試，我們只要輸入一句程式，就可以立刻看到電腦執行結果是否正確。

⚠ 本書後面章節若看到程式前面有 >>>，便表示是在 **Shell** 窗格內執行與測試。

如果要讓電腦執行或停止程式，請依照下面步驟：



若要打開之前儲存的程式或範例程式檔，請如下開啟：



4. 雲端Colab開發環境

Google Colaboratory（或簡稱Google Colab）是 Google 提供的一項供工具服務！簡單來說，Google Colab 是線上市版的 Jupyter Notebook，只要你有 Google 帳號，請先登錄後，就可免費使用 Google Colab 服務！

Summary:在瀏覽器使用 **Google Colaboratory**

Step 1. 前往 <https://colab.research.google.com/>

Step 2. 點選 NEW NOTEBOOK - 修改檔案名稱 Step 3. 完成 - 執行cell請按
Ctrl+Enter - Ctrl+S 存檔

4-1 第一次使用Google Colab (Optional)

在管院電腦教室使用時，請務必先登錄學校的電腦系統，然後再用**Google Chrome**進行操作。

請注意：若不如此操作，執行常會出問題。或者你可以選擇自備筆電。

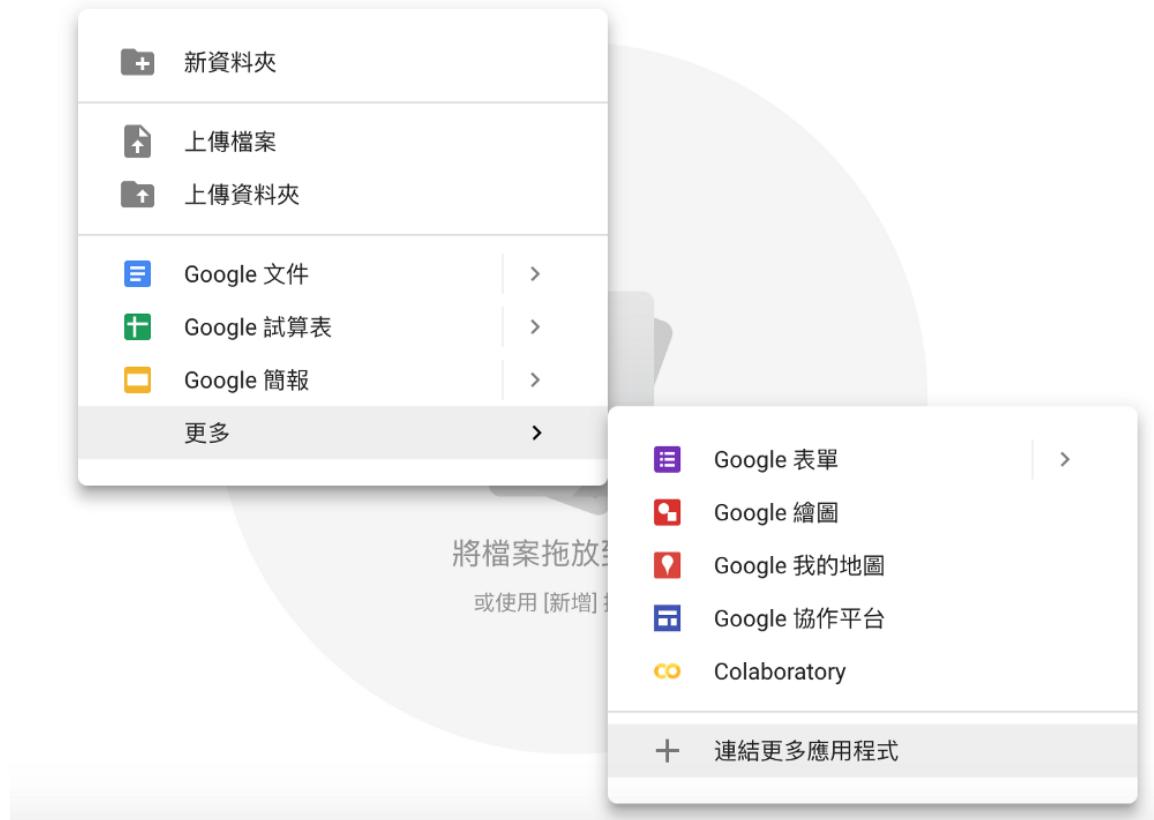
Step 1 :用Gmail登錄Chrome

Step2 :進入你的 [Google 雲端硬碟] 進行連結設定

~第一次使用，請進行以下操作：

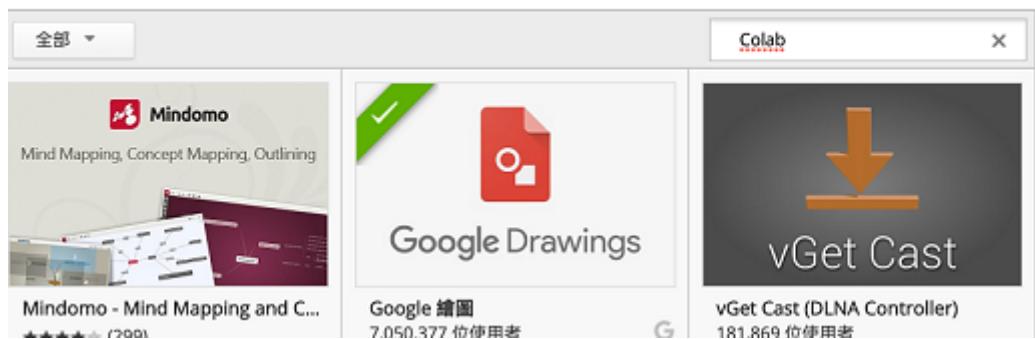
- (1) 點選[新增+] → 按下[更多] → 按下 [連結更多應用程式] (如圖一)

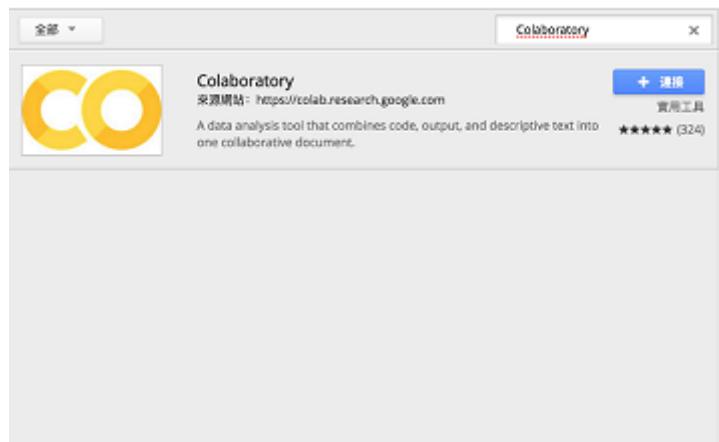
我的雲端硬碟 > Colab ▾



- (2) 在右上角打入 Colaboratory 搜尋之後按下 [連結] (如圖二)

將應用程式連接到雲端硬碟





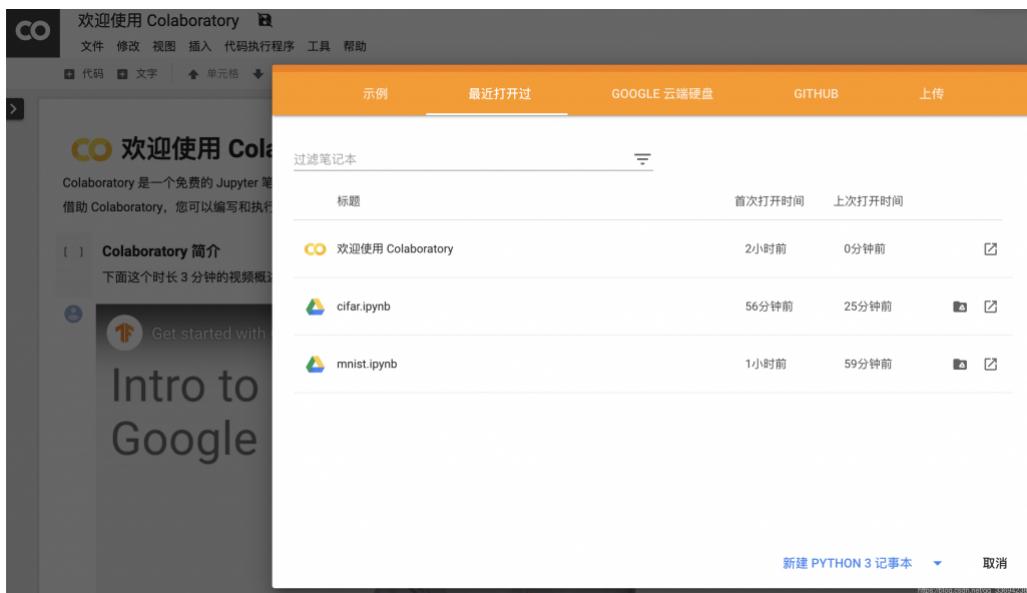
- (3) 你可以回到[連結更多應用程式]那裡，就會看到有[Colaboratory]的選項了。日後有三種方式使用Colab：
 - A. 請直接點選[Colaboratory]選項
 - B. 或先建立一個名叫「範本」檔案，從雲端硬碟點選使用。
 - C. 或直接從<https://colab.research.google.com>進入操作。

4-2 建立一個 Google Colab 文件

Step 1 : 打開一個Google Chrome瀏覽器，輸入網址就可以看到以下頁面。

<https://colab.research.google.com>

- 請在右下方按藍色的[New Python 3 Notebook]，選擇[新建Python 3筆記本]，就能看到下面完全配置好的Python運行環境了。
- 或直接點選開啟舊的Python檔案（之前的存檔）。



Step 2 : 檔案（重新）命名



- 進入 Google Colab 介面後，如果想更改檔案名稱，請點擊左上角檔案名稱，然後進行些改，就可以替檔案重新命名了。

Step3: 開始寫 Code

在文件中只要在單元格中打入 Python 語法之後，按下：

- **Shift + Enter**：執行此單元格後，移至下個單元格（若下面無單元格則會新建一個新的單元格）

4-3 上傳、下載與儲存檔案



～上傳：只要是.ipynb就可以。

Google Colab 的使用介面和 Jupyter Notebook 幾乎是一模一樣的，連文件的副檔名也一樣是 .ipynb。因此，假如你電腦中已經有一份 .ipynb 的文件，只要傳上 Google 雲端硬碟便可以用 Google Colab 開啟執行唷！

- 請選[文件] →[上傳筆記本]
- 然後選擇檔案，上傳

～下載：.ipynb與.py兩種檔案。

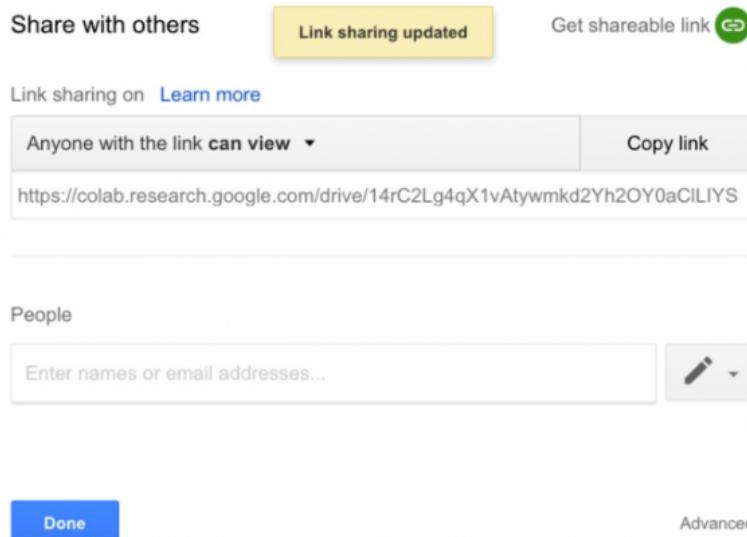
你也可以把Google Colab筆記本，直接下載成為ipynb文件，或py文件，在本地保存副本。要注意選擇需要的儲存路徑，避免不知存到哪裡去了。

- 請選[文件] →[下載.ipynb]
- 請選[文件] →[下載.py]

你也可以把Google Colab筆記本，直接下載放到雲端硬碟或Github帳戶上面，副檔名均為.ipynb。

- 請選[文件] →[在雲端硬碟保存一份副本]
- 請選[文件] →[在GitHub保存一份副本]

4-4 分享、協作與求助



～分享：用Google Colab把筆記本共享出去。

- 先點擊筆記本右上角的Share按鈕。
- 然後選擇「複製鏈接」。
- 在共享對象處，可以不用填入Email或姓名，但在後面要選[可以查看View]選項。

把獲取的共享鏈接連同你的文字描述，直接貼到Python的論壇或者課程討論區裡。別人只需要點擊，就能查看你的全部程式碼、相關資訊，以及運行你的筆記副本，即使使用不同的作業系統、不同的瀏覽器。

～協作：用Google Colab讓大家一起協作筆記。

- 先點擊筆記本右上角的Share按鈕。
- 然後選擇「複製鏈接」。
- 最後在共享對象框裡，選擇給對方「可以編輯」的修改權限。

把獲取的共享鏈接連同你的文字描述，直接貼到Python的論壇或者課程討論區裡。別人只需要點擊，就能與你協作（評論）。

～求助：作為新手，你遇到錯誤和問題，是完全正常的，而Python具有強大的社群，可以提供很多幫助。

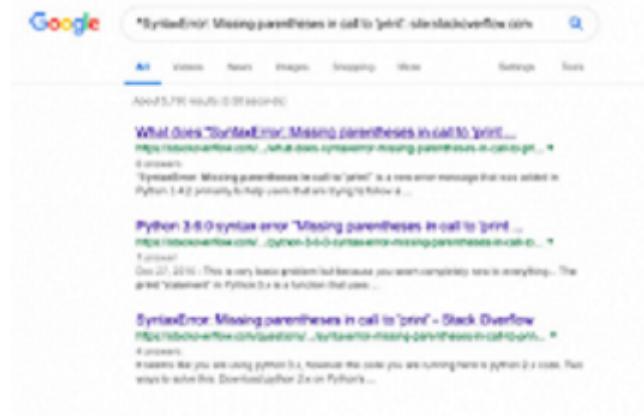
- Google Colab為你主動尋找問題答案提供了工具支持。每當你遇到報錯的時候，你都會看到下方有個按鈕[Search Stackoverflow]。請點擊這個按鈕，Google Colab就會用搜尋引擎，在Stackoverflow這個IT問答站上，幫你尋找相關報錯的已有答案。

```
[1] print "This is my solution!"

File "<ipython-input-1-a09954ca6674>", line 1
    print "This is my solution!"
      ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print("This is my solution"

SEARCH STACK OVERFLOW
```

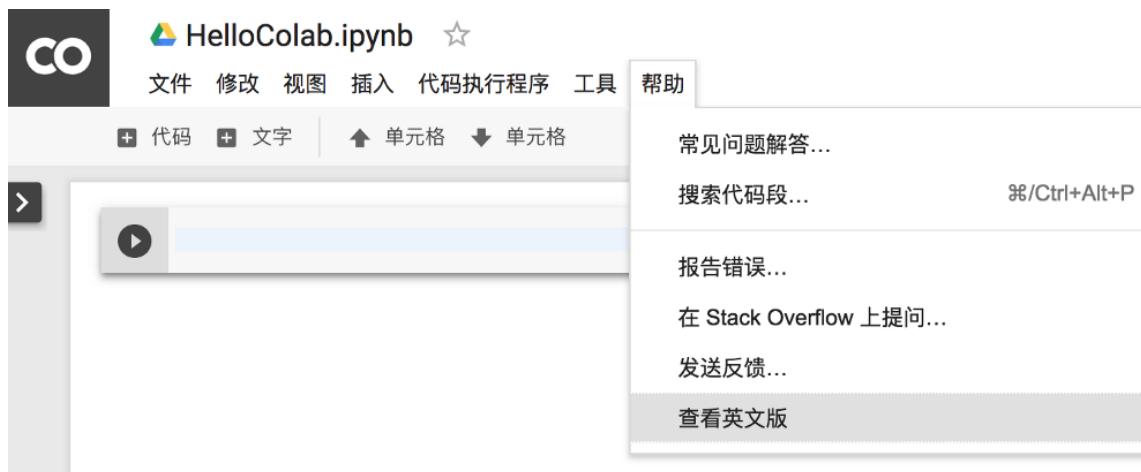
一般來說，點擊前幾條資訊，你就會有收穫。



4-5 其他設定

語言選擇

目前 Google Colab 在中文只有簡體中文介面，如果喜歡使用英文介面的朋友們可以點選 [幫助] → 按下 [查看英文版]，就能切換到英文使用介面囉！



GPU/TPU

若是想使用 GPU 來訓練模型，就必須先選擇含有 GPU 的使用環境，否則只會被分配到沒有 GPU 的虛擬機，而設定的方式為：在代碼執行程序中選擇 [更改運行時類型] → 硬件加速器選項選擇 [GPU]。

+ 代码 + 文字 ↑ 单



- 全部运行 ⌘/Ctrl+F9
 - 运行当前单元格之前的所有单元格 ⌘/Ctrl+F8
 - 运行光标所在的单元格 ⌘/Ctrl+Enter
 - 运行所选单元格的内容 ⌘/Ctrl+Shift+Enter
 - 运行当前单元格以及其后的所有单元格 ⌘/Ctrl+F10
 - 中断执行 ⌘/Ctrl+M I
 - 重新启动代码执行程序... ⌘/Ctrl+M .
 - 重新启动并运行所有单元格...
 - 重置所有代码执行程序...
-
- 更改运行时类型
-
- 管理会话

笔记本设置

运行时类型

Python 3

硬件加速器

GPU

None

 保存此笔记本时将所有单元格输出项

GPU

TPU

取消

保存