

appendix

A

Python 開發環境： Anaconda 與 WinPython 客製化套件

- ▷ A-1 建立 Anaconda 的 Python 開發環境
- ▷ A-2 安裝本書客製化的 WinPython 可攜式套件
- ▷ A-3 使用 Thonny 的 Python IDE



A-1

建立 Anaconda 的 Python 開發環境

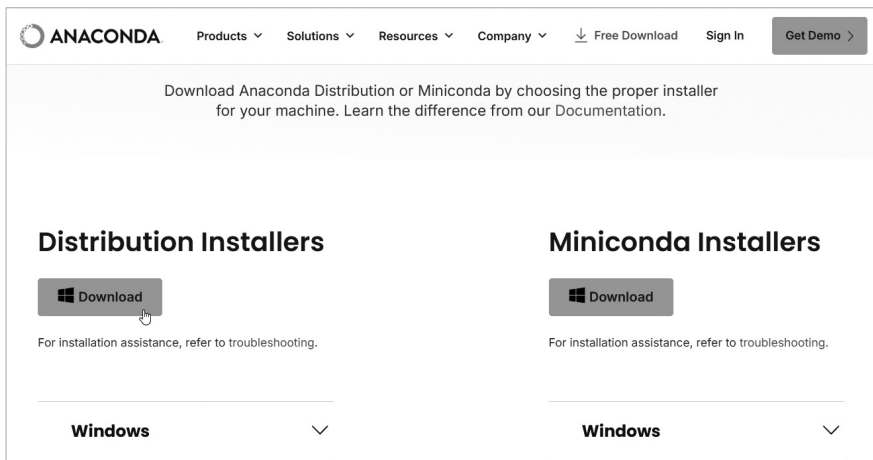
基本上，本書內容可以使用市面上常見的 Python 開發環境。除了第 A-2 節 WinPython 客製化的 Python 開發環境外，也可以自行安裝 Anaconda 整合安裝套件來建立所需的 Python 開發環境（需自行安裝 Python 相關套件）。

Anaconda 是 Python 語言著名的整合安裝套件，內建 Spyder 整合開發環境和 Jupyter Notebook。除了標準 Python 模組外，還預設安裝包含 Scipy、NumPy、Pandas 和 Matplotlib 等資料科學、機器學習與人工智慧的相關套件。

下載 Anaconda

在 Anaconda 官方網站可以免費下載 Windows 版的 Anaconda 整合安裝套件，其網址如下所示：

<https://www.anaconda.com/download/success>



請按 **Download** 鈕下載 Anaconda 安裝程式，在本書下載的檔名是：Anaconda3-2025.06-0-Windows-x86_64.exe。

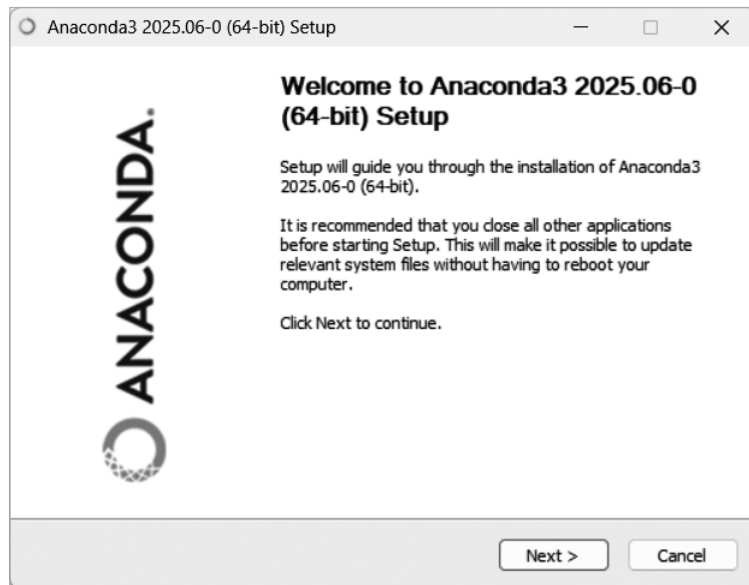
安裝 Anaconda

成功下載 Anaconda 安裝程式後，就可以在 Windows 電腦進行安裝。本書是在 Windows 11 作業系統進行安裝（如果已經安裝舊版 Anaconda，請先解除安裝套件），其步驟如下所示：

Step

1

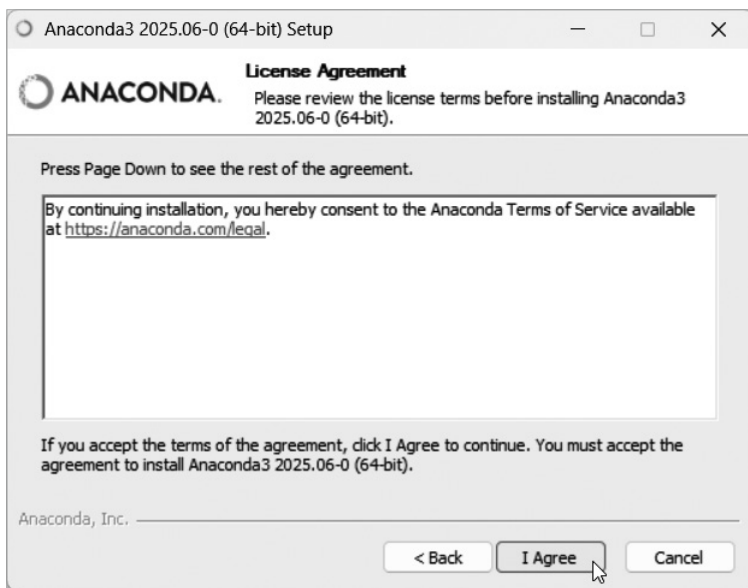
請雙擊 **Anaconda3-2025.06-0-Windows-x86_64.exe** 安裝程式檔案，稍等一下，可以看到歡迎安裝的精靈畫面，按 **Next >** 鈕，可以看到使用者授權書。



Step

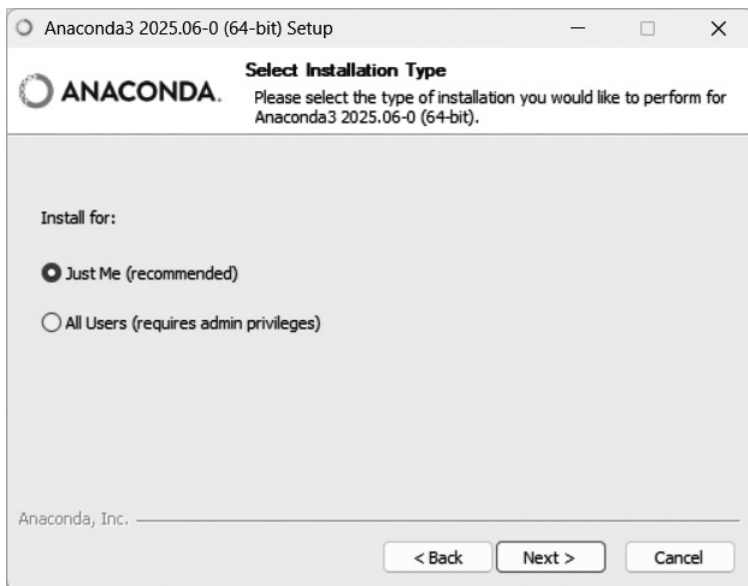
2

點選超連結可閱讀授權。請按 **I Agree** 鈕同意授權，即可選擇安裝類型。



Step
3

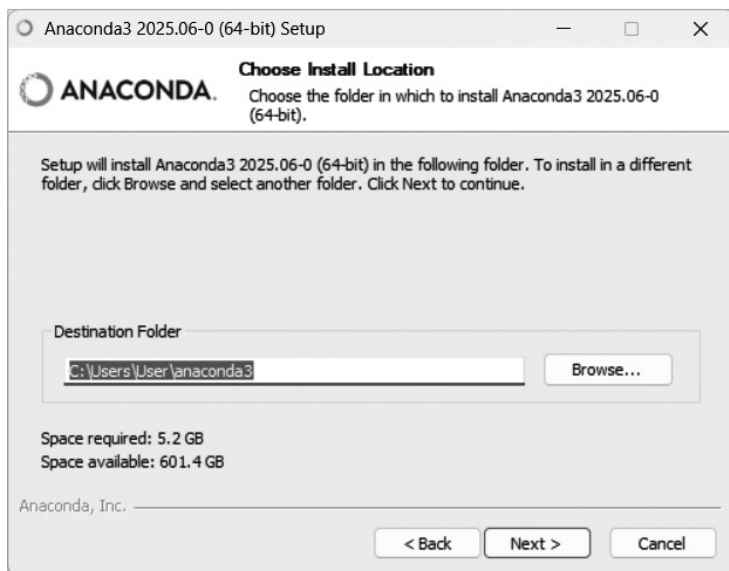
預選 **Just Me** 是安裝給目前使用者（建議），或選 **All Users** 安裝給所有使用者（需管理者權限）。請不用更改，按 **Next >** 鈕選擇安裝目錄。



Step

4

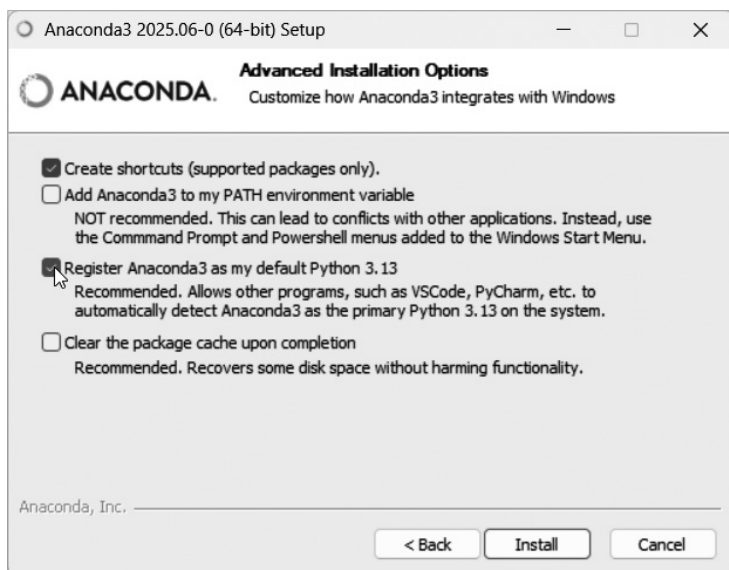
預設是安裝在使用者目錄，可以按 **Browse** 鈕來更改安裝目錄。請不用更改，按 **Next >** 鈕勾選所需的安裝選項。



Step

5

勾選註冊「Anaconda3 為我預設的 Python 3.13 版」後，按 **Install** 鈕開始安裝，可以看到目前的安裝進度。



Step

6

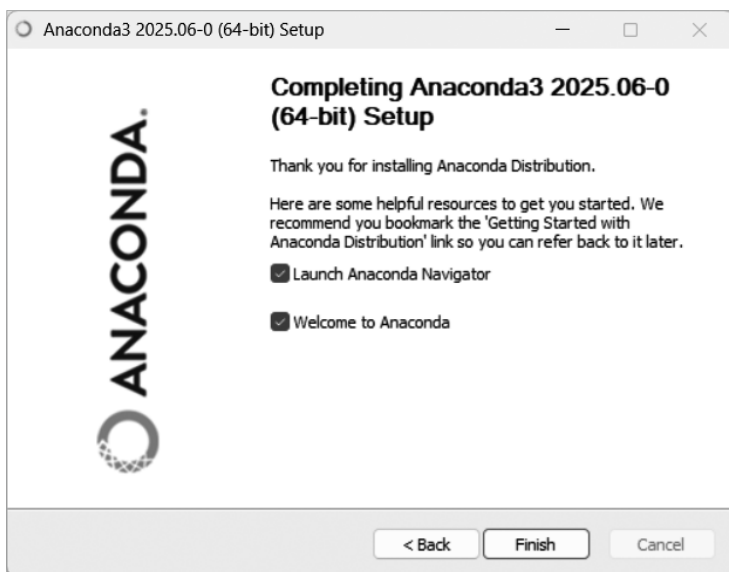
等到完成安裝後，請按 **Next >** 鈕，可以看到 Anaconda Cloud 的廣告畫面；再按 **Next >** 鈕，即可看到完成安裝的精靈畫面。



Step

7

按 **Finish** 鈕完成 Anaconda 整合安裝套件的安裝，同時可以看到瀏覽器開啟的相關說明文件。



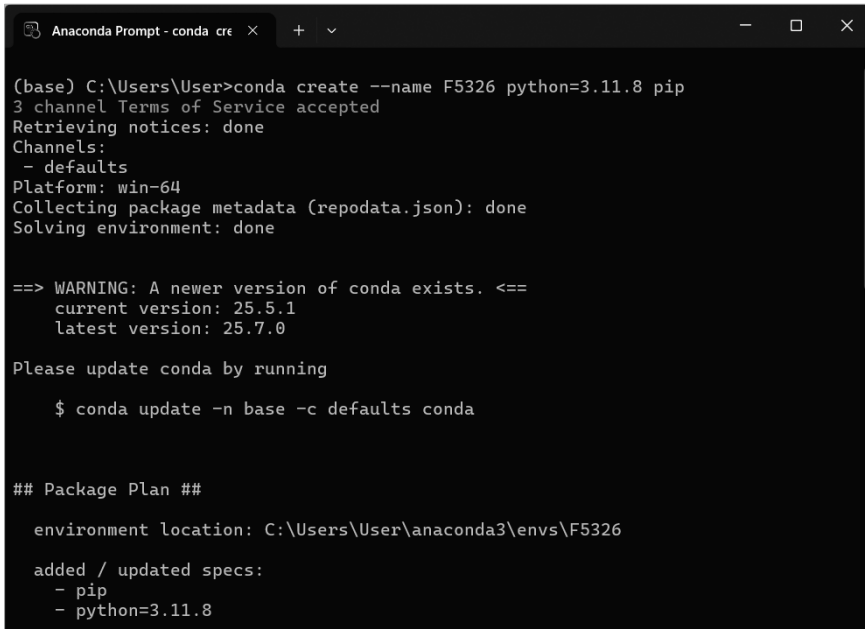
建立 Python 虛擬環境

Python 虛擬環境可以針對不同 Python 專案建立專屬的開發環境，例如：特定 Python 版本和不同套件的需求。特別是那些需要特定版本套件的 Python 專案，我們可以針對此專案建立專屬的虛擬環境，就不會因為特別版本的套件而影響其他 Python 專案的開發環境。

Anaconda 是使用 conda 命令來建立、啟動、刪除與管理 Python 虛擬環境。現在，我們準備在 Windows 11 作業系統新增名為 **F5326** 的虛擬環境，請執行「開始 / 全部 / Anaconda3 (64-bits) / Anaconda Prompt」命令開啟「Anaconda Prompt」命令提示字元視窗，輸入 conda create 指令建立 **Python 3.11.8 版**（本書使用的 Python 版本）的虛擬環境，如下所示：

```
(base) > conda create --name F5326 python=3.11.8 pip 
```

上述指令使用 --name 參數指定虛擬環境名稱為 F5326，python 參數指定 Python 版本 3.11.8 版，最後的 pip 是一併安裝 pip 套件管理，如下圖所示：



```
Anaconda Prompt - conda cre x + v
(base) C:\Users\User>conda create --name F5326 python=3.11.8 pip
3 channel Terms of Service accepted
Retrieving notices: done
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
current version: 25.5.1
latest version: 25.7.0

Please update conda by running

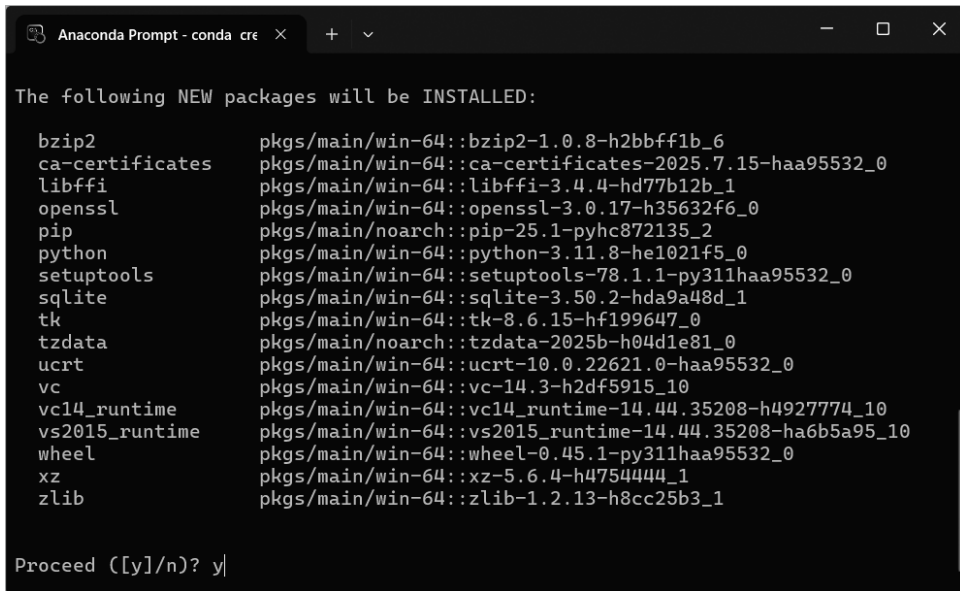
$ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\User\anaconda3\envs\F5326

added / updated specs:
- pip
- python=3.11.8
```

在解析環境後，會顯示套件計劃 Package Plan，如果沒有問題，請按 **[Y]** 鍵建立虛擬環境，如下圖所示：



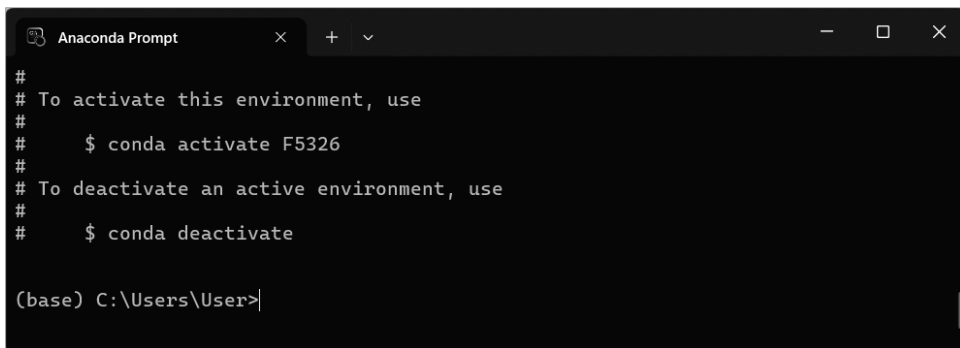
```
Anaconda Prompt - conda cre x + v

The following NEW packages will be INSTALLED:

bzip2                pkgs/main/win-64::bzip2-1.0.8-h2bbff1b_6
ca-certificates      pkgs/main/win-64::ca-certificates-2025.7.15-haa95532_0
libffi               pkgs/main/win-64::libffi-3.4.4-hd77b12b_1
openssl              pkgs/main/win-64::openssl-3.0.17-h35632f6_0
pip                  pkgs/main/noarch::pip-25.1-pyhc872135_2
python               pkgs/main/win-64::python-3.11.8-he1021f5_0
setuptools            pkgs/main/win-64::setuptools-78.1.1-py311haa95532_0
sqlite                pkgs/main/win-64::sqlite-3.50.2-hda9a48d_1
tk                    pkgs/main/win-64::tk-8.6.15-hf199647_0
tzdata                pkgs/main/noarch::tzdata-2025b-h04d1e81_0
ucrt                  pkgs/main/win-64::ucrt-10.0.22621.0-haa95532_0
vc                    pkgs/main/win-64::vc-14.3-h2df5915_10
vc14_runtime          pkgs/main/win-64::vc14_runtime-14.44.35208-h4927774_10
vs2015_runtime        pkgs/main/win-64::vs2015_runtime-14.44.35208-ha6b5a95_10
wheel                 pkgs/main/win-64::wheel-0.45.1-py311haa95532_0
xz                    pkgs/main/win-64::xz-5.6.4-h4754444_1
zlib                  pkgs/main/win-64::zlib-1.2.13-h8cc25b3_1

Proceed ([y]/n)? y
```

等到成功建立 Python 虛擬環境後，就會在最後顯示啟動 F5326 虛擬環境的相關指令說明，如下圖所示：



```
Anaconda Prompt x + v

#
# To activate this environment, use
#
#     $ conda activate F5326
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) C:\Users\User>
```

啟動與關閉 Python 虛擬環境

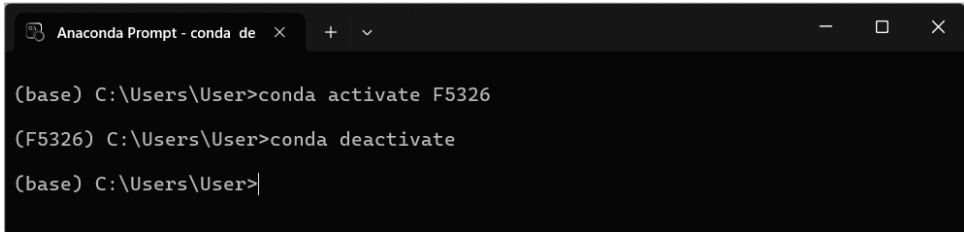
啟動 Python 虛擬環境 F5326 的命令列指令，如下所示：


```
> conda activate F5326 
```

在成功執行上述指令後，可以看到在前方的提示文字從 (base) 改成 (F5326)，表示目前是在 Python 虛擬環境 F5326。

關閉啟動的 Python 開發環境，請在啟動 Python 虛擬環境 F5326 後，使用下列命令列指令關閉 Python 虛擬環境，可以看到從 (F5326) 改成 (base)，如下所示：

```
> conda deactivate 
```



```
Anaconda Prompt - conda de x + v
(base) C:\Users\User>conda activate F5326
(F5326) C:\Users\User>conda deactivate
(base) C:\Users\User>|
```

A-2

安裝本書客製化的 WinPython 可攜式套件

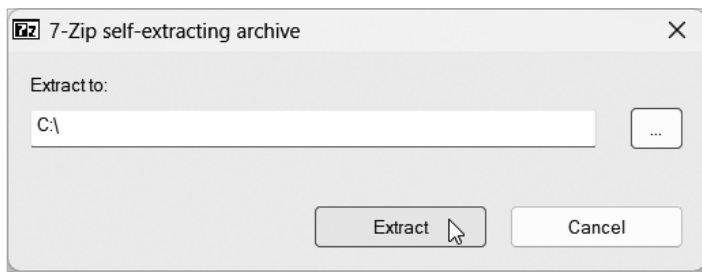
為了避免 Python 套件的更新腳步不一，造成新版本套件彼此之間不相容的問題，本書提供一套使用 WinPython 建立的客製化 Python 套件。此套件已經安裝好 Node.js + n8n，以及本書各章節指定版本的 Python 套件和 Thonny Python IDE，只需解壓縮，就可以馬上啟動 Thonny 來執行本書的 Python 程式。

首先，請參閱書附範例檔的說明來取得客製化 WinPython 套件，這是一個 7-Zip 自解壓縮檔，檔名是：**fChartEasy_n8n.exe**，其安裝和使用的基本步驟，如下所示：

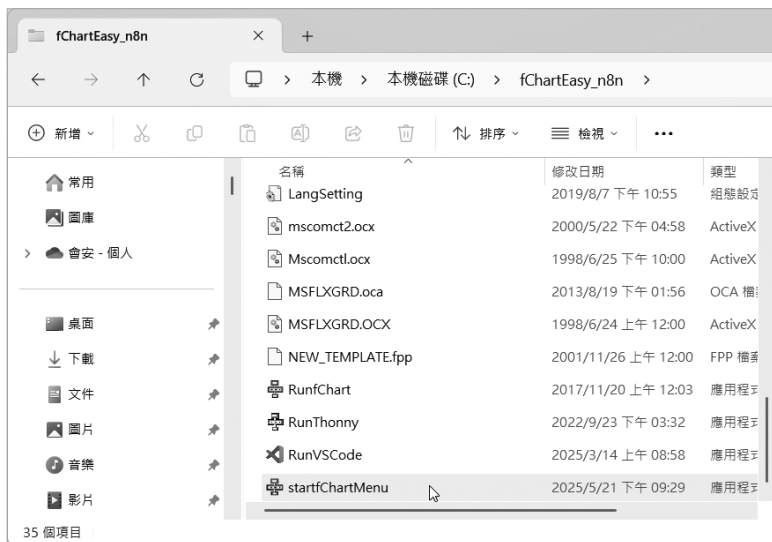
Step 1 請雙擊執行 **fChartEasy_n8n.exe**，可以看到「7-Zip self-extracting archive」對話方塊。

Tips 雙擊 **fChartEasy_n8n.exe** 後，如果出現「Windows 已保護您的電腦」視窗，請先點擊**其他資訊**，再按**仍要執行**鈕即可。

Step 2 在 **Extract to:** 欄位輸入解壓縮的硬碟根路徑，例如：「C:\」或「D:\」（請解壓縮至根目錄，以避免路徑過深和中文目錄等問題），再按 **Extract** 鈕解壓縮檔案來安裝客製化 WinPython 套件，如下圖所示：



Step 3 成功解壓縮後，預設建立名為「fChartEasy_n8n」目錄，請開啟此目錄且捲動至最後，雙擊 **startfChartMenu.exe** 來執行 fChart 主選單，如下圖所示：



Step

4

可以看到訊息視窗顯示「已經成功在 Windows 工作列啟動主選單」，請按**確定**鈕。



Step

5

在右下方 Windows 工作列可以看到 fChart 十字形圖示，點選該圖示，可以開啟主選單來啟動 Python 相關工具。請執行 **Thonny Python IDE** 命令啟動 Thonny 開發工具（**Python 命令提示字元 (CLI)** 命令則是啟動命令提示字元視窗來安裝 Python 套件），如下圖所示：



A-3

使用 Thonny 的 Python IDE

成功安裝客製化 WinPython 套件後，我們就可以啟動 Thonny 來撰寫第 1 個 Python 程式，或在互動環境輸入和執行 Python 程式碼。

補充說明

在 VS Code 只需指定 WinPython 套件的 Python 直譯器路徑，就可以使用本書客製化的 WinPython 套件，如下所示：

```
\\fChartEasy_n8n\WinPython\python-3.11.8.amd64\python.exe
```

A-3-1

建立第一個 Python 程式

現在，我們準備從啟動 Thonny 開始，一步一步建立你的第 1 個 Python 程式，其步驟如下所示：

Step

1

請在 fChart 主選單執行 **Thonny Python IDE** 命令啟動 Thonny 開發環境，可以看到簡潔的開發介面。



上述開發介面的上方是功能表，在功能表下方是工具列。工具列下方分成三部分；右邊是「協助功能」視窗，顯示協助說明（執行「檢視 / 協助功能」命令可切換顯示）。左邊分成上 / 下兩部分，上方是程式碼編輯器的標籤頁；下方是「互動環境 (Shell)」視窗，可以看到「Python 版本 3.11.8」。若要結束 Thonny 請執行「檔案 / 結束」命令。

Step

2

請在編輯器的**未命名**標籤輸入第 1 個 Python 程式的程式碼（如果沒有看到此標籤，請執行「檔案 / 開新檔案」命令新增 Python 程式檔案）。我們準備建立的 Python 程式只有 1 列程式碼，如下所示：

```
print("第1個Python程式")
```



請注意！如果輸入中文字串內容的 Python 程式碼，當輸入完中文字後，若無法成功輸入「"」符號，請記得從中文模式切換成英數模式，即可成功輸入「"」符號。

Step

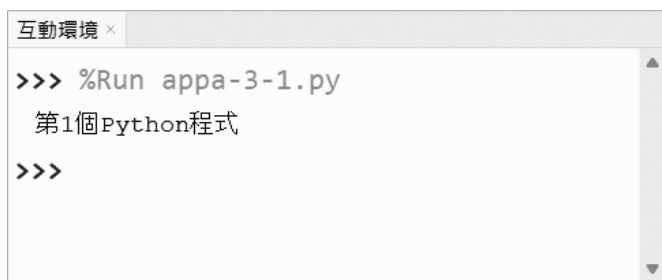
3

請執行「檔案 / 儲存檔案」命令或按工具列的**儲存檔案**鈕，然後在「另存新檔」對話方塊切換至「/F5326/appa」目錄，在下方輸入檔名 **appa-3-1**，按**存檔**鈕儲存成 appa-3-1.py 程式。

Step

4

可以看到標籤名稱已經改成檔案名稱。請執行「執行 / 執行目前腳本」命令，或按工具列綠色箭頭圖示的**執行目前腳本**鈕（也可按 **F5** 鍵）來執行此 Python 程式，就可以在下方「互動環境 (Shell)」視窗看到 Python 程式的執行結果。

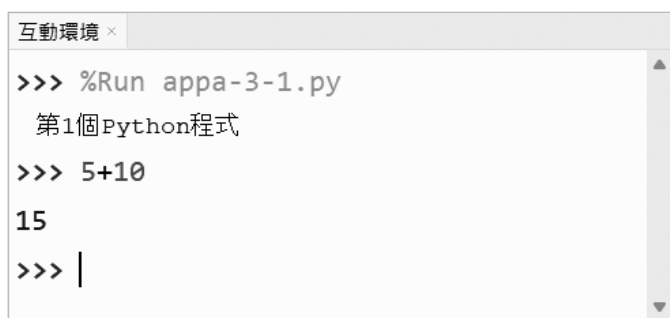


```
>>> %Run appa-3-1.py
第1個Python程式
>>>
```

對於本書的 Python 程式範例，請執行「檔案 / 開啟舊檔」命令開啟 Python 程式檔案，就能立即測試執行 Python 程式。

A-3-2 使用 Python 互動環境

在 Thonny 開發介面下方的「互動環境 (Shell)」視窗就是 REPL 交談模式。REPL (Read-Eval-Print Loop) 是循環「讀取 - 評估 - 輸出」的互動程式開發環境，可以直接在「>>>」提示文字後輸入 Python 程式碼並立即執行。例如：輸入 `5+10`，按 `Enter` 鍵，可立即看到執行結果 `15`，如下圖所示：



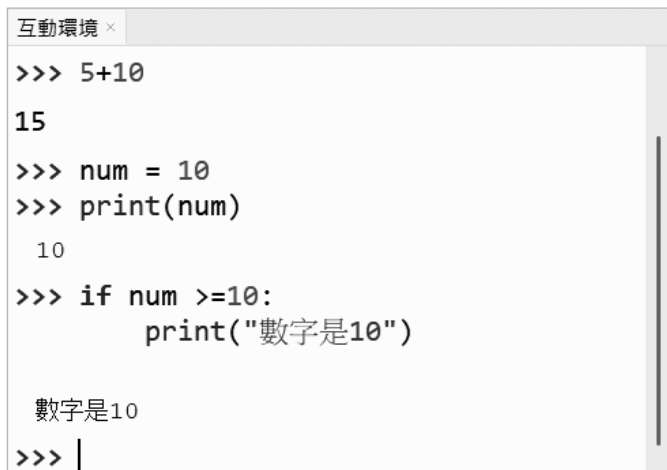
```
>>> %Run appa-3-1.py
第1個Python程式
>>> 5+10
15
>>> |
```

同樣地，我們可以定義變數 `num = 10` 後，輸入 `print()` 函式來顯示變數 `num` 的值，如下圖所示：



```
>>> 5+10
15
>>> num = 10
>>> print(num)
10
>>> |
```

如果輸入的是程式區塊，例如：if 條件敘述，請在輸入 if num >= 10: 後（最後輸入「:」冒號），按 **Enter** 鍵，就會換行且自動縮排 4 個空白字元。此時，我們需要按 2 次 **Enter** 鍵來執程式碼，才可看到執行結果，如下圖所示：



```
>>> 5+10
15
>>> num = 10
>>> print(num)
10
>>> if num >= 10:
    print("數字是10")
    數字是10
>>> |
```

A-3-3

Thonny 的 AI 輔助學習功能表

在 Thonny 的 Python IDE 有提供客製化功能表，這些新增的選單可以依據 fChart 流程圖符號來插入對應的 Python 程式碼。而最後的「AI 輔助學習」功能表則可自動產生用於詢問生成式 AI 的提示詞，如下圖所示：



上述命令能夠依據開啟的 Python 程式碼來自動產生 AI 提示詞並複製至剪貼簿，換句話說，只需按 **Ctrl** + **V** 鍵貼上至生成式 AI 的訊息欄位。例如：目前是開啟 `appa-3-1.py` 程式，執行「AI 輔助學習 / 解釋程式」命令，就會產生解釋此程式的 AI 提示詞並自動複製至剪貼簿，接著就可以開啟 ChatGPT 網頁，在欄位貼上此提示詞來詢問 ChatGPT。

在「AI 輔助學習」功能表的 AI 提示詞分成兩大類，第 1 大類的 3 種提示詞是幫助我們看懂程式碼，如下所示：

- **解釋程式**：讓生成式 AI 一一地詳細解說目前開啟的程式碼；如果有選取部分程式碼，則只會解釋選取的部分程式碼，例如：選取一列 Python 程式碼的方法，就可以讓生成式 AI 幫你解釋此方法。
- **更多範例**：讓生成式 AI 依據目前開啟的程式作為範本來寫出相似功能的 2 個程式。
- **語法測驗**：讓生成式 AI 用目前開啟的程式作為題目，出 2 題此語法範圍的選擇題。

而在第 2 大類的 3 種提示詞則是幫助我們除錯程式碼，如下所示：

- **用錯誤訊息除錯程式：**讓生成式 AI 自動使用程式執行產生的錯誤訊息來建立除錯的提示詞。
- **用解釋程式碼除錯程式：**如果沒有錯誤訊息，但覺得程式有問題，可以讓生成式 AI 在詳細解釋程式碼的同時，一一找出程式碼的所有可能錯誤，並且同時依據註解文字的說明來找出可能的語意錯誤。
- **修訂並更正程式問題：**讓生成式 AI 將你寫出的半成品程式碼，修訂更正成可執行的程式。



ChatGPT 在使用「AI 輔助學習」功能表的提示詞時，因為 AI 提示詞仍然會受到交談過程的上下文影響，當 ChatGPT 的回應已經明顯受到影響時，請開啟一個新交談或重新開啟 ChatGPT 來進行交談。