第八章、檔案與例外處理

檔案依照功能可以分為「程式檔」(programfile)和「資料檔」(datafile)兩大類。所謂的「程式檔」就是一群程式碼指令的集合體,經過編譯後所產生的(*.exe)檔案,它可以被用來解決問題或處理資料。至於程式要處理的「資料」,若資料量小者可放在程式碼的串列或集合中成為程式內部資料,但資料量很龐大時,就不適合以內部資料來處理。這時就必須將資料獨立於程式碼之外的外部資料,外部資料就稱為「資料檔」。「資料檔」的內容必須透過專屬程式檔的處理,才能呈現有用的資訊。

例如學校成績系統是程式檔,可以處理各科系、各班級的成績,而學生的各種成績資料會獨立成資料檔。當成績資料有異動時,只要修改資料檔案的內容即可,不用修改程式檔案的內容。而同一個程式系統(程式檔),可以處理同一規格的不同資料檔。例如不同班級的成績資料,是用不同的資料檔案分別存放。

檔案也可分成「文字檔」與「非文字檔」兩種。文字檔是可以直接閱讀,即一般人可以看得懂的,如:程式的原始碼(*.py)、網頁程式碼(*.htm)。而非文字檔是一般人看不懂的字碼,當然是無法直接閱讀的,如:音樂檔、圖形檔、編譯後的類別檔。

isdir()、isfile()、exists()

這三個函式是 os 套件中 path 物件下的函式, 因此須特別留意引用時的語法應為

```
import os
f = os.path.isfile(file name)
```

isdir()與 isfile()函式,分別用來檢查指定的資料夾路徑名稱與檔案路徑名稱是否存在；exists()函式是用來檢查指定的檔案或資料夾路徑名稱是否存在,可同時取代 isdir()及 isfile()函式。

檔案的開啟與關閉

當一個資料檔的資料要進行處理時,必須先用開檔函式將檔案打開,才能進行資料檔內容的讀取、處理、修改、存放,最後再用關檔函式將檔案關閉。Python 的內建函式 open()可用來開啟指定的資料檔, 已開啟的資料檔若不再使用,則要用 close()函式來關閉。資料檔內容若有經過寫入、修改,有部分的資料串流是暫時放在電腦的主記憶體緩衝區內,如果沒有用 close()函式來關檔而是直接結束程式執行,會造成暫放在緩衝區中的資料串流沒有回存資料檔內而遺漏語法如下:

```
物件變數 = open(檔案路徑名稱, '模式')
物件變數.close()
```

1. 物件變數：若開檔成功,此時一個檔案是一組字元資料串流,而這個串流是存放在主記憶體準備進一步的運用操作。所以這個串流已經是一個「物件」了,故用物件變數代表該開啟的檔

案資料串流。

2. 檔案路徑名稱：是必須被使用參數,不能省略。它的內容是用來存取的資料檔案名稱,包含檔案所在的路徑名稱,屬字串型別。如果內容只有檔案名稱而沒有路徑名稱,則系統會以目前系統程式執行檔所在的資料夾做為檔案的資料夾所在。至於路徑名稱所在的資料夾必須事先存在,若不存在會出現錯誤,系統不會主動建立。
3. 模式參數用來設定資料檔的開啟模式,省略時預設為讀取模式,屬字串型別, 如下:

模式	說明
r	讀取模式(預設值)。若資料檔不存在，會出現錯誤。(不可寫入)
w	寫入模式。若資料檔不存在，會建立該名稱的資料檔；若資料檔已存在，則原資料檔的內容會先被刪除，再寫入新的資料。(不可讀取)
a	新增模式。若資料檔不存在，會建立該名稱的資料檔；若資料檔已存在，則新寫入的資料會新增至原資料檔內容的尾端。(不可讀取)
r+	讀寫模式。讀取時，使用方式同 r 模式。寫入時，則寫入的資料會覆蓋原檔案相同位置的內容，若檔案不存在，會出現錯誤。(可讀寫)
w+	讀寫模式。寫入時，使用方式同 w 模式。讀取時，需先用 seek() 函式指定讀取指標位址，才能讀取所需資料。seek(0)為檔案開頭。(可讀寫)
a+	新增讀寫模式。寫入新增時，使用方式同 a 模式。讀取時，需先用 seek() 函式指定讀取指標位址，才能讀取所需資料。seek(0)為檔案開頭。(可讀寫)

with…as…

使用 open()函式開啟的檔案,經處理後,必須使用 close()函式來將檔案關閉。若使用 with…as…語法來開啟檔案,則檔案處理完後,不需要使用 close()函式便會自動關閉檔案。語法如下:

with open(檔案路徑名稱, “模式”) as 物件變數:

文檔的讀取與寫入

用程式處理檔案資料的方式,有從資料檔讀取資料再進一步操作運算呈現需要的訊息,有從鍵盤輸入或從讀卡機掃入的資料再放入資料檔儲存。本節會說明如何將資料寫入檔案,以及如何從檔案中將資料讀取出來。Python 常用的檔案處理函式如下表所示:

函式	說明
flush()	清理緩衝區釋放記憶空間，若緩衝區內還有資料，會全部寫入檔案中。
write(字串)	將字串寫入資料串流。先暫存於緩衝區，再由緩衝區寫入檔案中。
read([size])	從檔案中讀取指定 size 的字元。若未指定 size，則讀取指標位置後面所有字元。
readline([size])	從檔案中讀取所在行指定 size 的字元。若未指定 size，則讀取一整行。
readlines()	從檔案讀取所有行的資料，並回傳一個串列，一個元素放置一個行的內容。
seek()	移動檔案文件讀取指標的位置，如: seek(0)為檔案的開頭。

例外處理

所謂「例外」(Exception),就是當程式碼在編譯期間沒有出現錯誤訊息,而在程式執行時發生錯誤,這種錯誤稱為執行時期錯誤。進行例外處理是不希望程式中斷。而是希望程式能捕捉錯誤,進行錯誤補救,並繼續執行程式。若錯誤是使用者輸入不正確資料所造成的,可以要求使用者再輸入正確資料後才繼續執行。

Python 使用 try…except…finally 敘述來解決例外處理,它的方式是將被監視的敘述區段寫在:的程式區塊,當程式執行到 try:內的敘述有發生錯誤時,會逐一檢查該錯誤所屬的例外類別,以便執行該 except:內的敘述。最後不管是否有符合 except,都會執行最後的 finally:敘述區段。例外處理的語法如下:

```
try:
    受監視的敘述區段
except 例外類別 1 [as e] :
    處理錯誤的敘述區段 1
except 例外類別 2 [as e] :
    處理錯誤的敘述區段 2
except Exception [as e] :
    處理其它錯誤的敘述區段
finally:
    最後會執行敘述區段
```

例外類別	說明
ZeroDivisionError	除數為 0 的算術運算錯誤。
ValueError	數值錯誤。如使用內建函式時，參數型別與傳入值不符。
NameError	變數名稱未定義，而直接運算產生的錯誤。
IndexError	串引註標(索引)超出宣告範圍
IOError	I/O 異常處理所產生錯誤。
FileNotFoundError	檔案或資料夾找不到時所產生的錯誤。
Exception	程式執行時，所有內建、非系統引發所產生的異常錯誤。

1. 使用 try:敘述時,至少要有一個檢查 except(捕捉)或 finally:敘述區段配合。
2. 多個檢查 except(捕捉)敘述區段,由上至下 except 逐一檢查,若遇到符合條件的例外類別,則執行該對應敘述區段,則以下的 except 就不再處理(但若有 finally 會執行 finally 區塊)。
3. 常用到的例外類別如下表: