

Chapter 5

Files

Access

5-1以文字表達檔案的存放路徑

- 以文字表示桌面路徑（位置）
- 上面：Windows；下面：Mac

語法

C:\Users\使用者帳號\Desktop

語法

/Users/使用者帳號/Desktop

5-2 使用者介面的概念

- GUI 和 CUI 分別是下列 2 個名詞的簡稱
- GUI **G**raphical **U**ser **I**nterface（圖形化使用者介面）
- CUI **C**haracter **U**ser **I**nterface（字符使用者介面）

▼ CUI 表達的位置等同於 GUI 上看到的位置



5-3 以CUI操作電腦的方法

- Windows的狀況：
- CD指令:Change Directory (改到不同目錄)



DIR 指令：顯示所在目錄的內容



```
C:\Users\Flag>dir ←  
磁碟區 C 中的磁碟沒有標籤。  
磁碟區序號： XXXX-XXXX
```

C:\Users\Flag 的目錄

2015/12/01	18:02	<DIR>	.
2015/12/01	18:02	<DIR>	..
2015/10/09	19:48	<DIR>	.gimp-2.8
2014/03/07	12:05	<DIR>	.gradle
2015/11/17	20:50	<DIR>	.idlerc
...省略			
2015/12/01	18:27	<DIR>	.vagrant.d
2015/12/01	18:29	<DIR>	.VirtualBox
2016/01/08	16:12	<DIR>	Desktop

顯示目前所在目
錄內容的清單

改到Desktop目錄



```
C:\Users\Flag>cd Desktop ↵
```

```
C:\Users\Flag\Desktop>
```

mkdir: 建立目錄名稱

語法

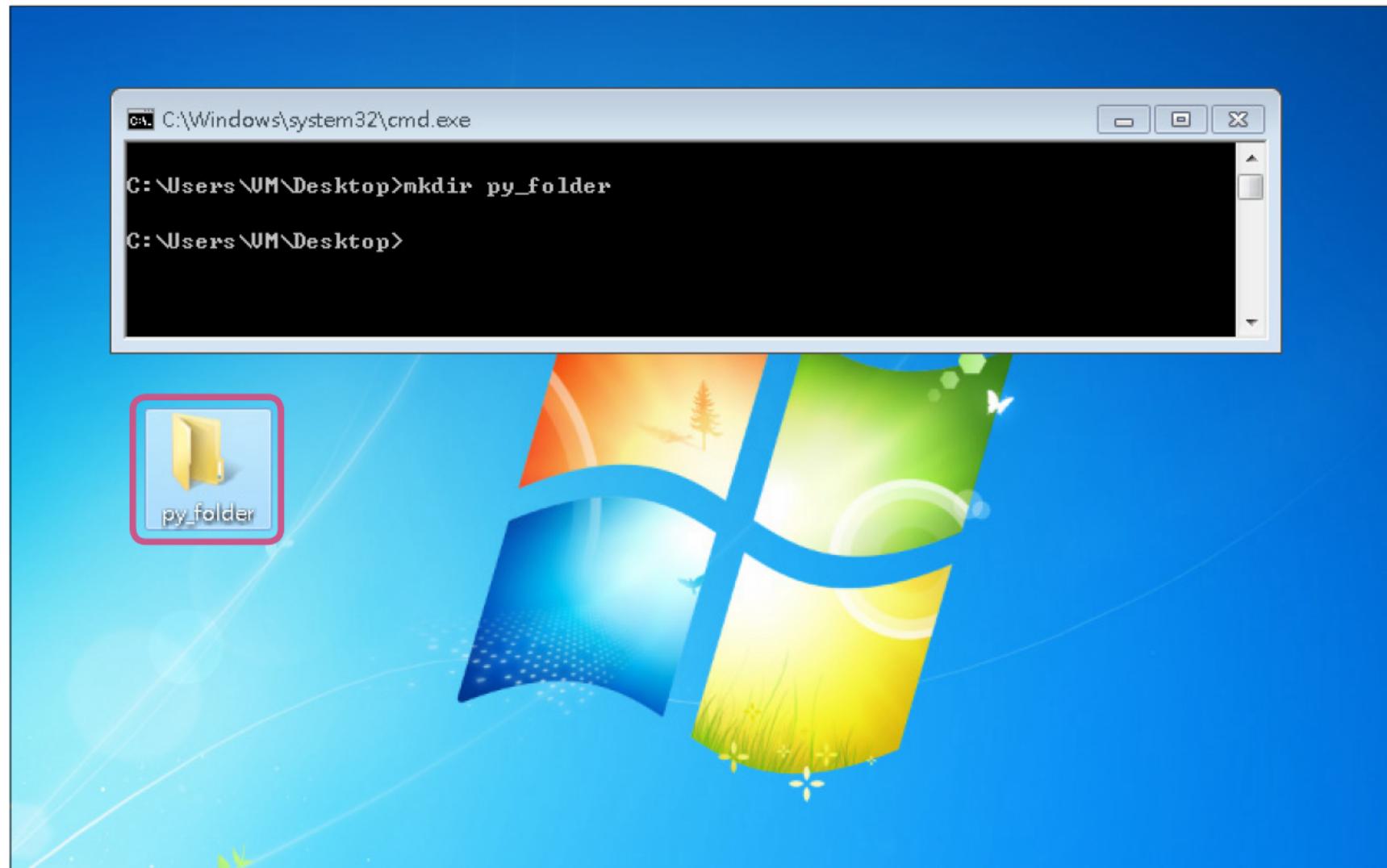
mkdir 目錄名稱



```
C:\Users\Flag\Desktop>mkdir py_folder ↵
```

```
C:\Users\Flag\Desktop>
```

▼ 在桌面新增了資料夾！



Windows指令一覽表

● 指令的記憶方式

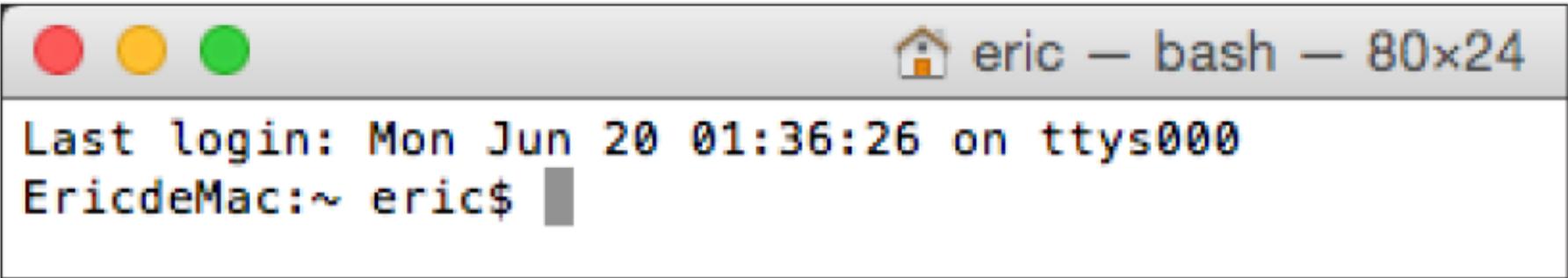
▼ 目前為止所學過的指令一覽表

指令	指令來源	用途
cd	current directory change directory	顯示目前所在位置 從現在位置移動到指定的新位置
dir	directory	顯示目前位置下的目錄和檔案清單
mkdir	make directory	在目前位置, 新增特定名稱的目錄

Mac 的狀況

- 一開始請先啟動終端機吧！可以在 Finder 的應用程式中，找到工具程式的資料夾，而終端機程式就位於其中。

▼ Mac 的終端機



A screenshot of a Mac OS X terminal window. The window title bar shows three colored window control buttons (red, yellow, green) on the left and the text "eric - bash - 80x24" on the right. The main window area contains the following text:
Last login: Mon Jun 20 01:36:26 on ttys000
EricdeMac:~ eric\$

pwd : 顯示路徑; ls : 顯示路徑內容



```
$ pwd ↵  
/Users/flag
```



```
$ ls ↵  
Applications      Public      Library  
Desktop           Movies     Documents  
Music            Downloads   Pictures
```

cd : 變更路徑



```
$ cd Desktop/ ↵  
$ pwd ↵  
/Users/flag/Desktop  
$ ls ↵  
$
```

mkdir: 建立目錄

語法

```
mkdir 目錄名稱
```



```
$ mkdir python_folder ↵  
$ ls ↵  
python_folder
```

MAC指令一覽表

● 指令的記憶方式

▼ 目前為止所學過的 Mac 指令一覽表

指令	指令來源	用途
pwd	print working directory	顯示目前所在位置
ls	List	顯示目前位置下的目錄和檔案清單
cd	change directory	從現在位置移動到指定的新位置
mkdir	make directory	在目前位置, 新增特定名稱的目錄

Try ...以程式操作檔案前的準備

● Windows的狀況



cd Desktop ↵	←	移動至桌面
mkdir py_folder ↵	←	建立新資料夾
cd py_folder ↵	←	移動至新資料夾中
python ↵	←	啟動互動式介面

Try...以程式操作檔案前的準備

● Mac 的狀況

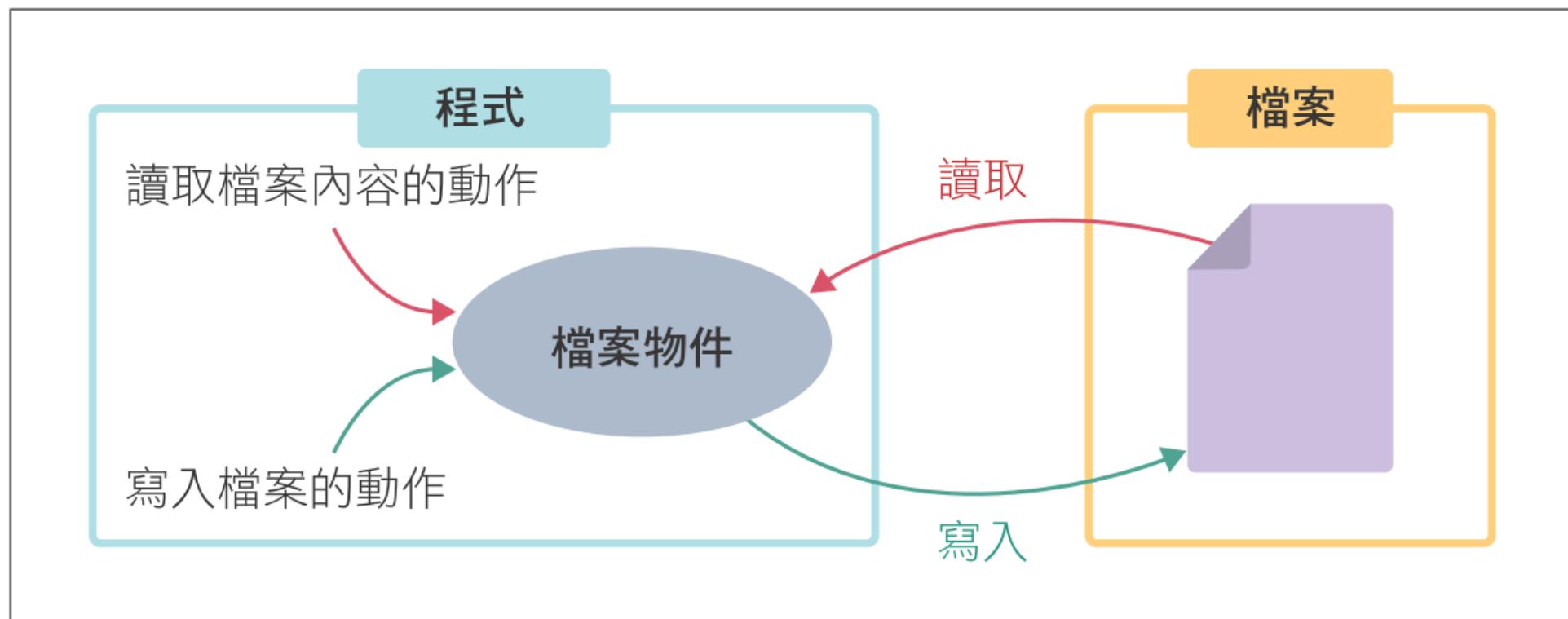


```
bash-3.2$ cd Desktop/ ← • 移動至桌面  
bash-3.2$ mkdir py_folder ← • 建立新資料夾  
bash-3.2$ cd py_folder/ ← • 移動至新資料夾中  
bash-3.2$ python3 ← 啟動互動式介面
```

5-4 檔案物件：協助存取實際檔案的介面

● 什麼是檔案物件？

▼ 檔案物件的示意圖



建立檔案物件

語法

```
open('檔案名稱', '模式')
```

▼ 模式的指定方式

指定模式	意義
r	用來讀取檔案內容
w	清空檔案再寫入資料
a	將資料附加至原本內容最後面

File not found ... Try



```
>>> open('null.txt', 'r') ↵
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'null.txt'
```

使用寫入模式的步驟

- 1 以寫入模式建立檔案物件。
- 2 透過檔案物件將資料寫入檔案。
- 3 關閉檔案物件。

Open ... Try ...

- 檔案不存在則建立它，若存在則清空內容準備寫入



```
01 >>> file_object = open('python.txt', 'w') ←  
02 >>> file_object.write('this is sample of python.') ←  
25 ────────── 顯示寫入的文字數量  
03 >>> file_object.close() ← ────────── 關閉檔案物件 確認是否已經關閉  
04 >>> file_object.write('this is sample of python.') ← ──  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
ValueError: I/O operation on closed file.
```

因為檔案物件已經關閉所以發生錯誤

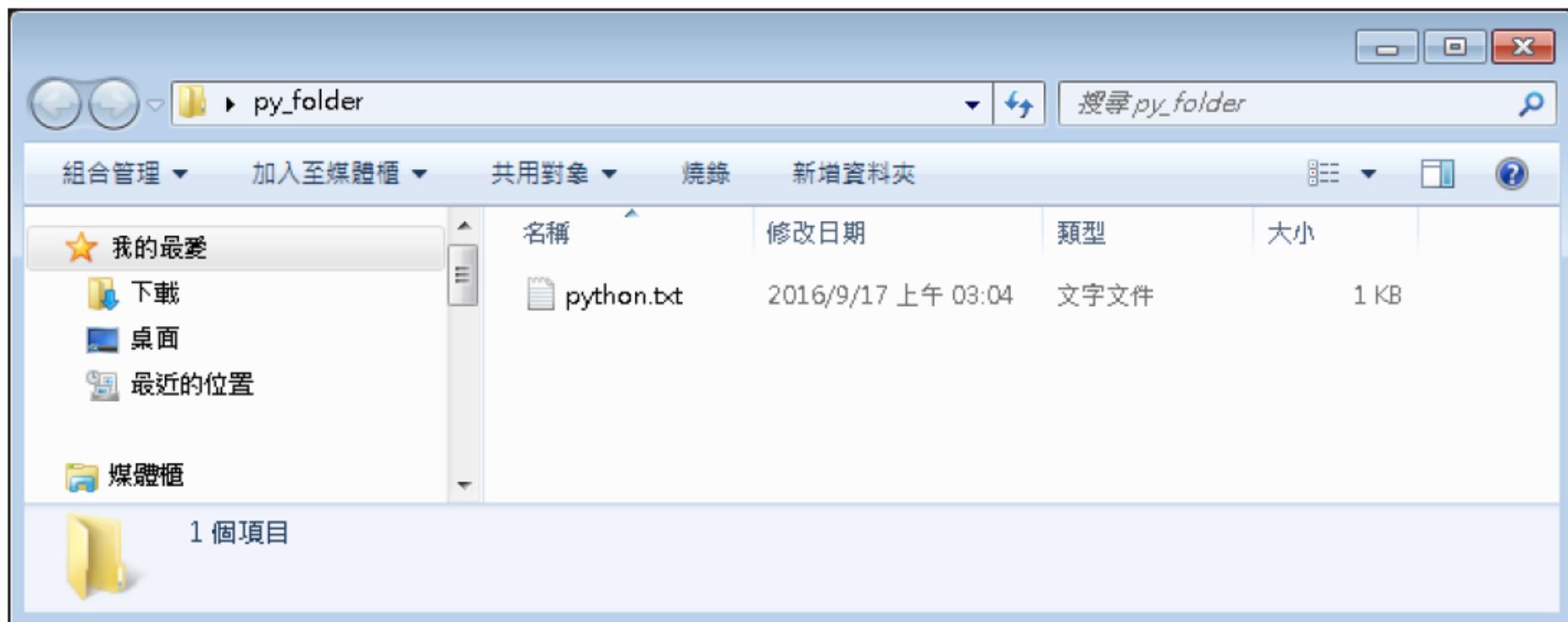
file_object.flush() ...



```
>>> file_object = open('python.txt', 'w') ← 最後的「\n」  
>>> file_object.write('this is sample of python.\n') ← 為換行符號  
25  
>>> file_object.flush() ← 要求立即完成寫入的動作
```

確認新建立檔案的位置

▼ GUI 中顯示檔案的畫面



使用讀取模式的步驟... try

- 1 以讀取模式建立檔案物件。
- 2 透過檔案物件以 `read` 方法讀取資料。
- 3 關閉檔案物件。



```
>>> file_object = open('python.txt', 'r')
>>> file_object.read()
'this is sample of python.'
```

指定檔案名稱與位置



Windows 的操作方式

```
>>> file_object = open('C:\\\\Users\\\\使用者帳號\\\\Desktop\\\\py_folder\\\\python.txt','r') ↵  
>>> file_object.read() ↵  
'this is sample of python.'
```

※ 在 Windows 作業系統中, 以 1 串文字指定檔案位置時, 各層資料夾必須以「\\」隔開。



Mac 的操作方式

```
>>> file_object = open('/Users/使用者帳號/Desktop/py_folder/python.txt','r') ↵  
>>> file_object.read() ↵  
'this is sample of python.'
```

附加模式：在既有檔案最後加入新資料

- 1 以附加模式建立檔案物件。
- 2 以檔案物件的 `write` 方法寫入資料。
- 3 關閉檔案物件。



```
>>> file_object = open('python.txt', 'a') ↵
>>> file_object.write('Add data from program!!') ↵
>>> file_object.close() ↵
```

對檔案執行讀取+寫入動作：'r+'



```
01 >>> file_object = open('python.txt', 'r+') ↵
02 >>> file_object.read() ↵
03 'this is sample of python.\n'
04 >>> file_object.write('Add data from program!!') ↵
05 23
```



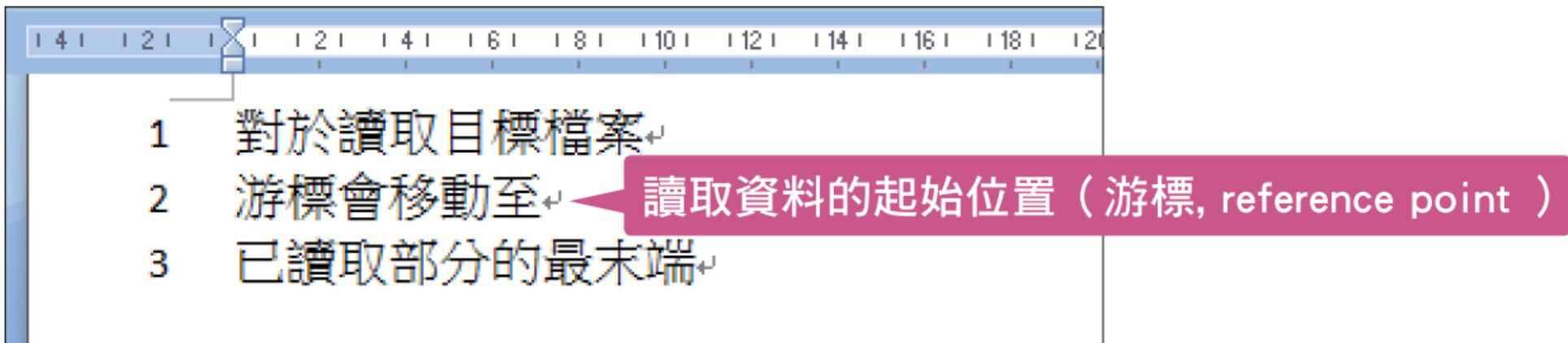
```
>>> file_object.read() ↵
''
```

若沒有關檔則讀不到資料



```
>>> file_object.read() ↵  
''  
>>> file_object.close() ↵
```

▼ read 方法的「讀取資料的起始位置」示意圖



Try...



```
01 >>> file_object = open('python.txt', 'r+') ↵
02 >>> file_object.read() ↵
03 'this is sample of python.\n' ↵
04 >>> file_object.read() ↵
05 ''
```



```
>>> file_object.write('Add data from program!!') ↵
23
```

將讀取資料的起始位置移至檔案開頭



```
>>> file_object.seek(0) ←  
0
```



```
>>> file_object.read() ←  
'this is sample of python.\n Add data from program!!'  
>>> file_object.close() ←
```

使用with語句的檔案寫入方式

- 當程式脫離with範圍檔案物件自動關閉

語法

```
with open('檔案名稱', '模式') as 檔案物件名稱:  
    tab 對檔案執行的操作
```



```
>>> with open('with.txt', 'w') as file_object: ↵  
...     tab file_object.write('using with!') ↵
```

```
... ↵
```

```
11
```

} with 的區塊

如果模式錯誤會如何？



```
>>> obj = open('python.txt', 'r') ↵
>>> obj.write('can I take it?') ↵
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    io.UnsupportedOperation: not writable
```

縮排建議採用 4 個空白

▼ Python 2.x 讓 tab 和空白共存的縮排方式

```
>>> for i in range(2):
... [tab] print('tab') •————— 1 個 tab
...           print('space') •————— 8 個空白
```

End