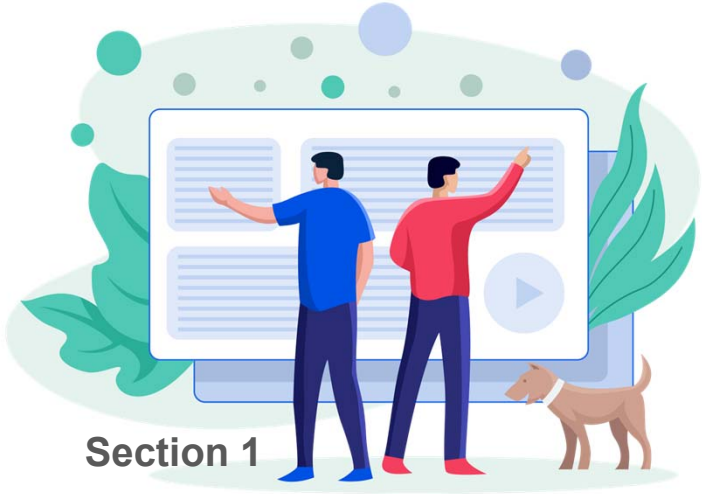
基本輸入與輸出

學習目標

- 使用輸入內建函數
- 使用輸出函數
- 解釋各種文字格式化的優缺點與用法

輸出與輸入

- 介紹 print() 用法
- 介紹 input() 與 eval() 用法



顯示到螢幕

- Python 使用 print() 函數，可以把資料顯示到標準輸出(standard output)，預設為螢幕
- print() 函數會把收到的運算轉成文字，再把結果送到螢幕上
- print() 語法：
  - print(*objects, end='\n')
  - 可以傳入零或多個參數，用逗點分隔
  - 預設 print() 會換行，可用 “end” 參數來改變
- 範例
  - print ('Python is simple,', 'enjoy! ')

從鍵盤輸入資料

- Python 使用 input() 函數從鍵盤讀資料
- input 語法：
  - input([prompt])
  - 如果有提供 “prompt” 參數，則螢幕會出現 “prompt” 訊息，然後游標停在訊息後面等待輸入資料
- eval() 函數會將收到的參數做運算後傳回結果
- eval 語法：
  - eval(expression)
  - expression 會被視為 Python 運算式來做運算

輸出與輸入(Demo)

- 使用 Python 互動式環境使用 print()、input()、eval()

格式化輸出

- 介紹 Python 三種格式化輸出
  - f-字串(f-Strings)
  - 文字格式化方法(str.format())
  - 文字格式化運算子(% Operator)

Section 2

文字格式化

- Python提供強大的內建文字格式化功能，不需要額外的程式庫
- 三種字串格式化方式
  - f-Strings
    - 最新版格式化字串風格
    - 從 python 3.6 版開始支援
  - str.format()
    - 較新風格
    - 豐富的客製化功能
  - 文字格式化運算子
    - 舊風格
    - 類似C語言的 sprintf 風格

f-strings -1

- 又稱為 “格式化字串文字 (formatted string literals)”
- f-strings是以 “f|F” 開頭的字串，可以使用大括號包含運算式
- Syntax: f'[{expr}...]'
- 簡單範例

```
>>> name = 'III'
>>> est = 1979
>>> f'{name} was established in {est}.'
'III was established in 1979.'
```

f-strings -2

- 可包含任意的運算式
  - >>> f'{2 * 37}'
    '74'
- 支援多列 f-strings
  - >>> name = 'Benjamin'
    >>> prof = 'scientist'
    >>> lang = 'Python'
    >>> message = (
    ... f'Hi {name}. '
    ... f'You are a {prof}. '
    ... f'You like {lang}.'
    ...)
  - >>> message
    'Hi Benjamin. You are a scientist. You like python.'

f-strings -3

- **數字格式** 語法：{資料變數:格式}
  - >>> data=12.34567
  - >>> f'Value={data:4.0f}'
    'Value= 12'
  - >>> f'Value={data:5.2f}'
    'Value=12.35'
  - >>> f'Value={data:06.2f}'
    'Value=012.35'
- **日期時間格式**
  - >>> from datetime import datetime
  - >>> f'{datetime(2001, 2, 3, 4, 5):%Y/%m/%d}'
    '2001/02/03'
  - >>> f'{datetime(2001, 2, 3, 4, 5):%Y-%m-%d %H:%M:%S}'
    '2001-02-03 04:05:00'

f-strings -4

- 可以直接使用方法(method)
  - >>> name = 'PYTHON'
    >>> f'{name.lower()} is funny.'
    'python is funny.'
- 字典(Dictionaries) 風格
  - >>> data = {'name': 'III', 'est': 40}
    >>> f"The name is {data['name']}, aged {data['est']}."
    The name is III, aged 40.
- f-strings的 ‘f’ 也是代表快速(fast) 的意思
  - 效能比另外兩種文格式化方式好，f-String在大數據操作時，處理效率較高
  - 可以在f-String直接使用方法及字典風格

*args 和 **kwargs

- str.format(*args, **kwargs)
- *args 代表可以收多個引數，所收集的資料是位置引數
- **kwargs 代表可以收多個關鍵字引數，所收集的資料是關鍵字引數，使用dict來取得關鍵字名稱(關鍵字名稱即為dict的key)
- args 是 argument的簡寫，是引數的意思，透過 * 收集的引數會被放到一個 tuple 中，所以我們可以使用 for 來對它進行拆解。

```
print('{2}, {1}, {0}'.format(*'abc'))
# 'c, b, a'
```

- kwargs 則是 Keyword Argument 的簡寫，透過 ** 可以拆解dict或是將資料收集至dict

```
data = {'name': ‘LabVIEW360’, 'est': 20}
print('The name is {name}, aged {est}.'.format(**data))
# The name is LabVIEW360, aged 20
```

str.format()方法 -1

- 文字格式化是由字串內建的方法完成
- str.format(*args, **kwargs)
  - *args 代表可以收多個引數
  - **kwargs 代表可以收多個關鍵字引數
- 格式化文字內可以包含多個以 {} 表達的置換欄位，每一個置換欄位可以是一個索引，代表參數位置，或是關鍵字參數的名稱
- 每一個置換欄位都會換成相對應的參數內容值

str.format()方法 -2

- 簡單格式
  - >>> name='LabVIEW360'
    >>> est=2000
    >>> '{} is established in {}'.format(name, est)
    'LabVIEW360 is established in 2000'
  - >>> '{0} is established in {1}'.format(name, est)
    'LabVIEW360 is established in 2000'
  - >>> '{1} is the established year of {0}'.format(name, est)
    '2000 is the established year of LabVIEW360'

str.format()方法 -3

- 數字格式 語法：{:格式}
  - >>> data=12.34567
  - >>> 'Value={:4.0f}'.format(data)
    'Value= 12'
  - >>> 'Value={:5.2f}'.format(data)
    'Value=12.35'
  - >>> 'Value={:06.2f}'.format(data)
    'Value=012.35'
- 日期時間格式
  - >>> from datetime import datetime
  - >>> '{:%Y/%m/%d}'.format(datetime(2001,2,3,4,5))
    '2001/02/03'
  - >>> '{:%Y-%m-%d %H:%M:%S}'.format(datetime(2001,2,3,4,5))
    '2001-02-03 04:05:00'
  - 日期格式：YYYY-mm-dd HH:MM:SS 的格式相容性高，很多資料庫都支援

str.format()方法 -4

- 字典(Dictionaries) 風格
  - >>> data = {'name': 'LabVIEW360', 'est': 20}
    >>>'The name is {name}, aged {est}.'.format(**data) # unpack
    The name is LabVIEW360, aged 20.
- 其他
  - >>> '{2}, {1}, {0}'.format(*'abc') # unpack
    'c, b, a'
    >>> '{0}{1}{0}'.format('abra', 'cad')
    'abracadabra'

文字格式化運算子 -1

- 使用文字格式化運算子：%
- 語法：format % values
  - format 是一個字串
  - format 字串內可以使用 ‘%’ 轉換規格來放入某些值
  - value 代表參數值，如果 format 只需要一個參數，value 可以是單一資料。否則 values 必須是一個元組(tuple)，內含必要的參數值，或是一個對應物件，例如字典(dictionary)
- 目前較少使用，常見於Python 2的程式碼中。
- 未來盡量減少使用這一種文字格式化運算子。

文字格式化運算子 -2

- 簡單格式
  - >>> name='LabVIEW360'
    >>> est=2000
    >>> '%s is established in %d' % (name, est)
    'LabVIEW360 is established in 2000'
  - >>> '%d is the established year of %s' % (est,name)
    '2000 is the established year of LabVIEW360'
- %s 對應文字
- %d 對應整數
- %.nf n是小數點位數、f表示浮點數
- %.ne n是小數點位數、e表示科學記號

文字格式化運算子 -3

- 數字格式
  - >>> data=12.34567
  - >>> 'Value=%4.0f' % data
    'Value= 12'
  - >>> 'Value=%5.2f' % data
    'Value=12.35'
  - >>> 'Value=%06.2f' % data
    'Value=012.35'
- %f 對應浮點數

格式化輸出(Demo)

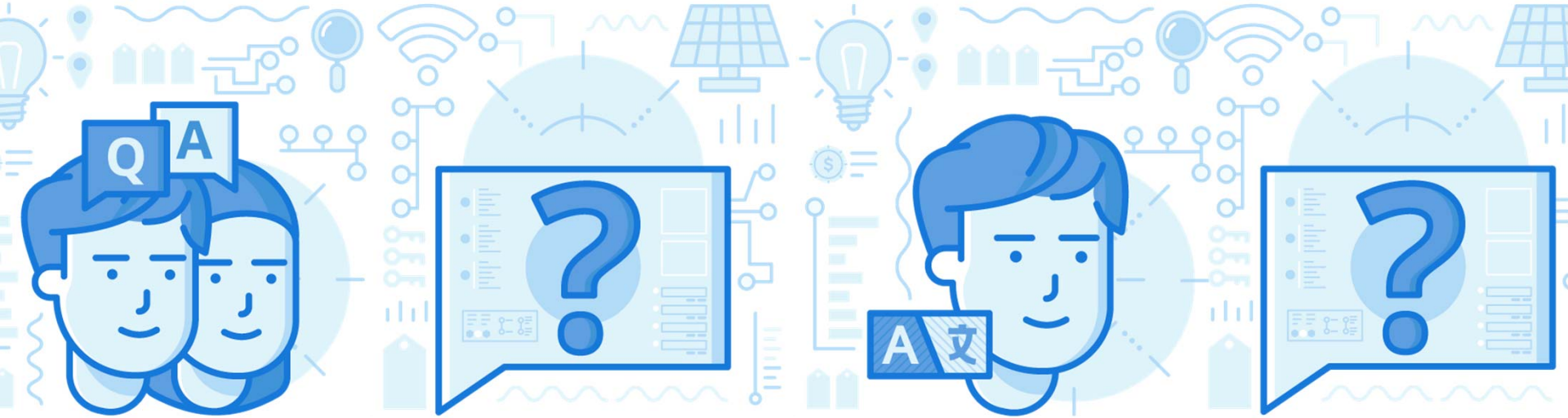
- 使用 Python 互動式環境做各種不同的格式化輸出方式

本章重點精華回顧

- print() 的用法
- 善用 input() 與 eval()
- 各種格式化輸出的方式

Lab: 基本輸入與輸出

- Lab01: 使用互動式環境做 print()、input() 與 eval() 練習
- Lab02: 使用互動式環境做格式化輸出練習



Lab01: print()、input() 與 eval() 練習

- 啟動Python互動式執行環境，做以下練習
- >>> name = 'John'
- >>> age = 33
- >>> print(name + ' is ' + str(age) + ' years old.')
- >>> print(name, 'is', age, 'years old.’)

- >>> data = input('Enter a number : ')
- >>> data
- >>> data = int(input('Enter a number : '))
- >>> data
- >>> data = eval(input('Enter a number : '))
- >>> data
- >>> data = eval(input('Enter expression : '))
- >>> data

Lab02: 使用互動式環境做格式化輸出練習 -1

```
>>> lang = 'Python'
>>> ver = 3
>>> f'{lang} {ver} is future'
>>> '{} {} is future'.format(lang, ver)
>>> '{0} {1} is future'.format(lang, ver)
>>> '%s %d is future' % (lang, ver)

>>> r = 4
>>> f’  Radius is {r}, area is {3.14 * r * r}'
>>> 'Radius is {}, area is {}'.format(r, 3.14 * r * r)
>>> 'Radius is %d, area is %f' % (r, 3.14 * r * r)
```

Lab02: 使用互動式環境做格式化輸出練習 -2

- >>> data = eval(input('Enter expression : '))
- >>> f'Result : {data}'
- >>> f'Result : {data:6.1f}'
- >>> f'Result : {data:06.1f}'
- >>> 'Result : {}'.format(data)
- >>> 'Result : {:6.1f}'.format(data)
- >>> 'Result : {:06.1f}'.format(data)
- >>> 'Result : %f' % data
- >>> 'Result : %6.1f' % data
- >>> 'Result : %06.1f' % data

Lab02: 使用互動式環境做格式化輸出練習 -3

- >>> from datetime import datetime
- >>> mydate=datetime.today()
- >>> f'Today is {mydate}'
- >>> f'Today is {mydate:%Y-%m-%d %H:%M:%S}'
- >>> 'Today is {0}'.format(mydate)
- >>> 'Today is {0:%Y-%m-%d %H:%M:%S}'.format(mydate)