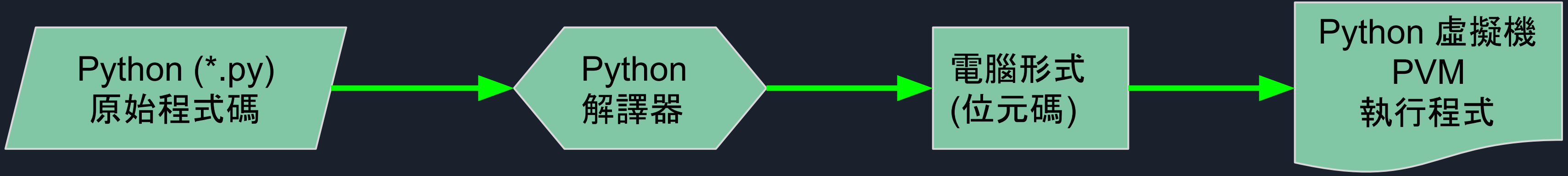
蟒蛇

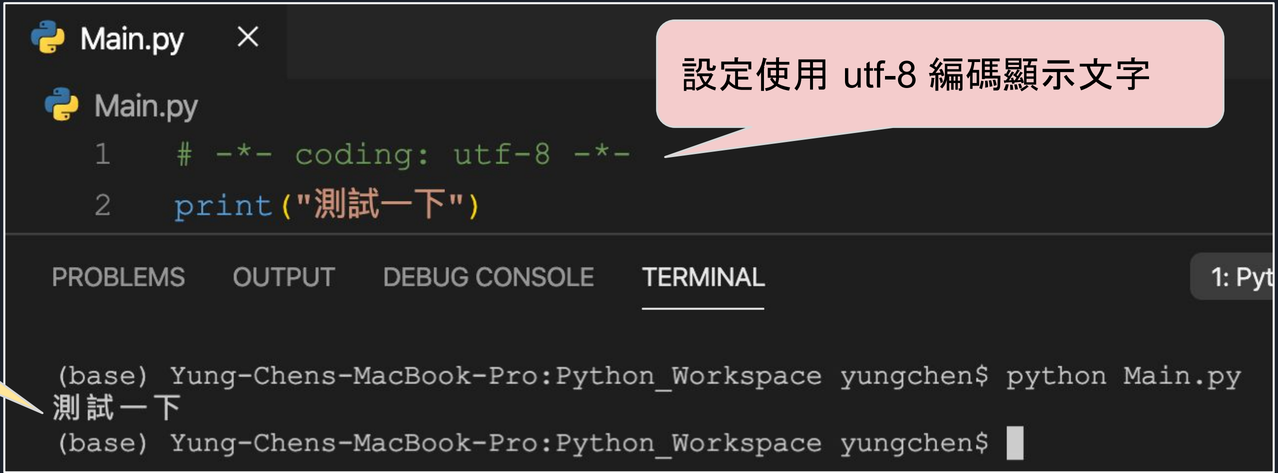
Python 3.8
Basic

基礎程式設計
Fundamental Programming
-- Python 3.8 Basic

Yung-Chen Chou Ph.D.
iSchool, Feng Chia University
Aug. 18, 2021

Python 執行流程





```python
# -*- coding: utf-8 -*-
print("測試一下")
```

設定使用 utf-8 編碼顯示文字

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
測試一下
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ 
```

執行結果

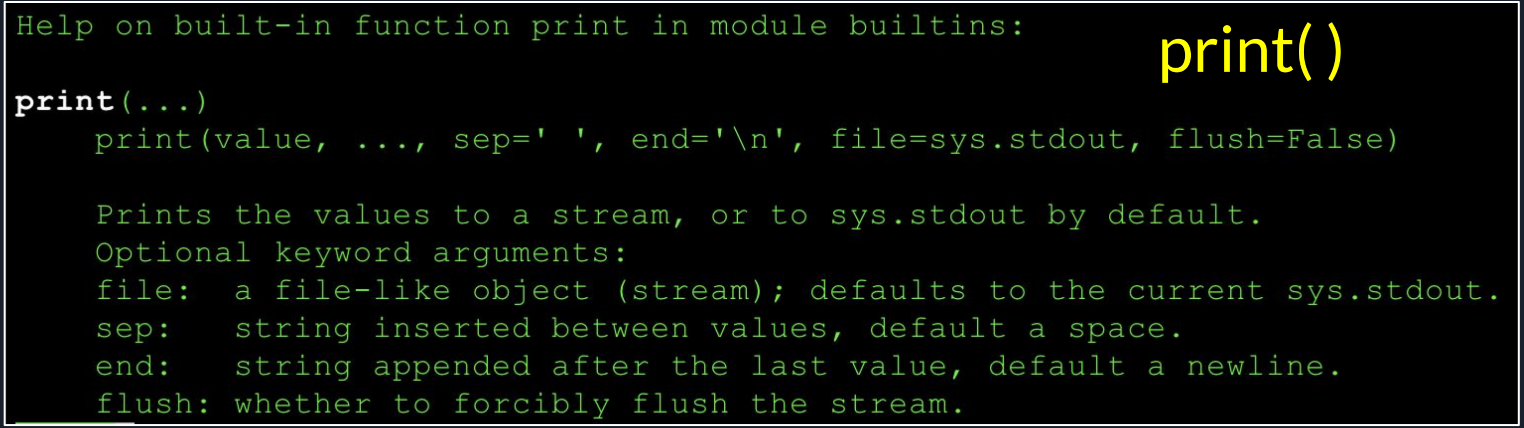
Python 介紹

- Python是支援程序導向、物件導向的動態語言
  - 動態語言不需事先宣告變數的資料型別，變數的資料型別可以在執行時再指定
- Python使用直譯器執行程式
  - 直譯器從頭到尾一行接著一行執行程式碼，又稱作腳本語言(scripting language)，不需要編譯就可以執行
- Python提供許多標準函式庫，並有許多第3方模組(third-party module)可以使用
- Python使用縮排方式表達程式區塊，語法直覺而簡單

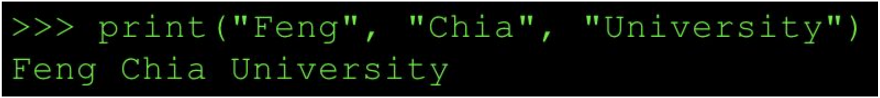
Python 介紹

- Python具備垃圾回收(garbage collection) 功能, 會自動管理記憶體, 回收沒有使用的記憶體
- Python中變數、數字、字串、函式、模組都是物件, 完全支援物件導向的程式設計
- Python能夠結合C與C++語言所撰寫的擴充程式
    - 使用Python將C與C++語言所撰寫的程式整合起來, 因此Python又稱為膠水語言(glue language)

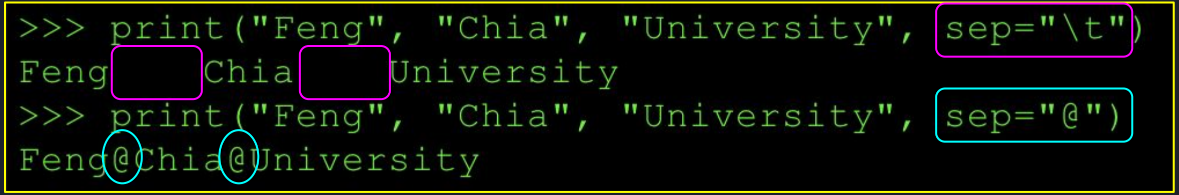
print( )



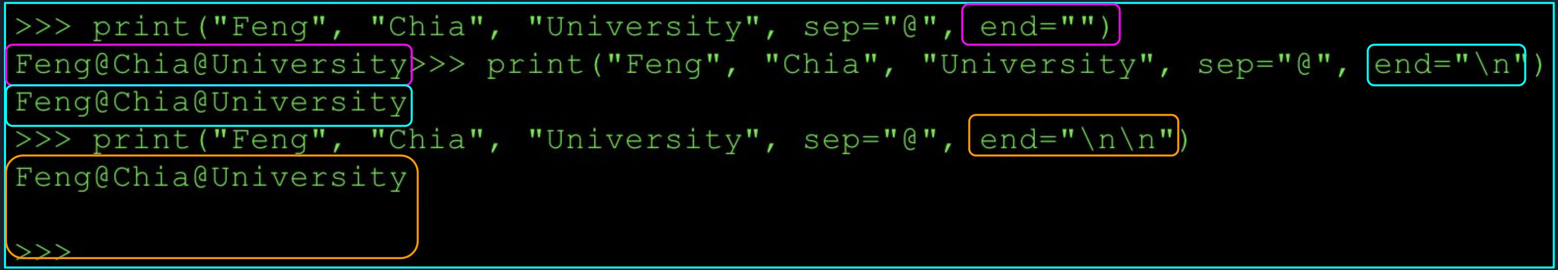
```
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```



```
>>> print("Feng", "Chia", "University")
Feng Chia University
```

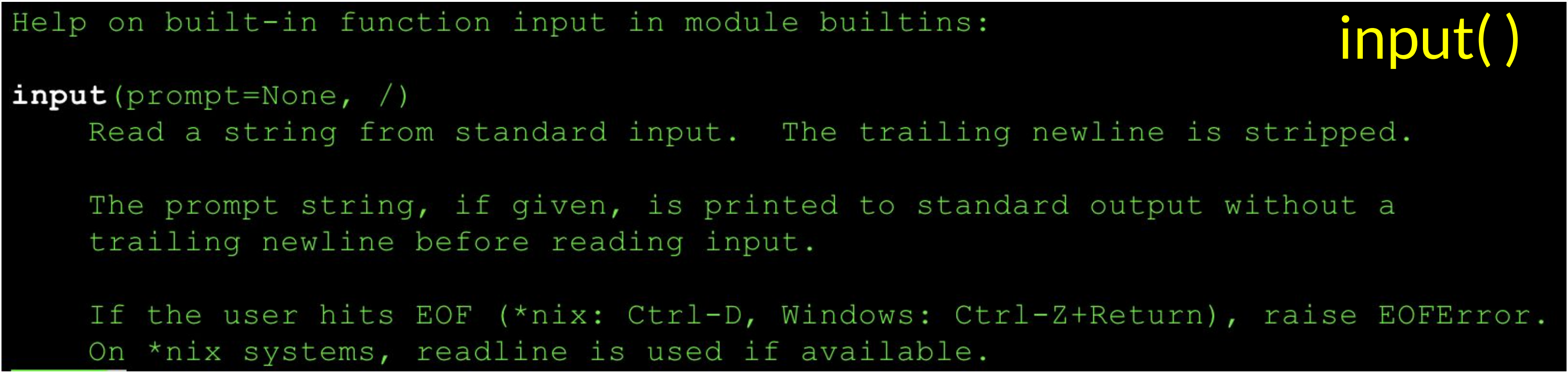


```
>>> print("Feng", "Chia", "University", sep="\t")
Feng    Chia    University
>>> print("Feng", "Chia", "University", sep="@")
Feng@Chia@University
```

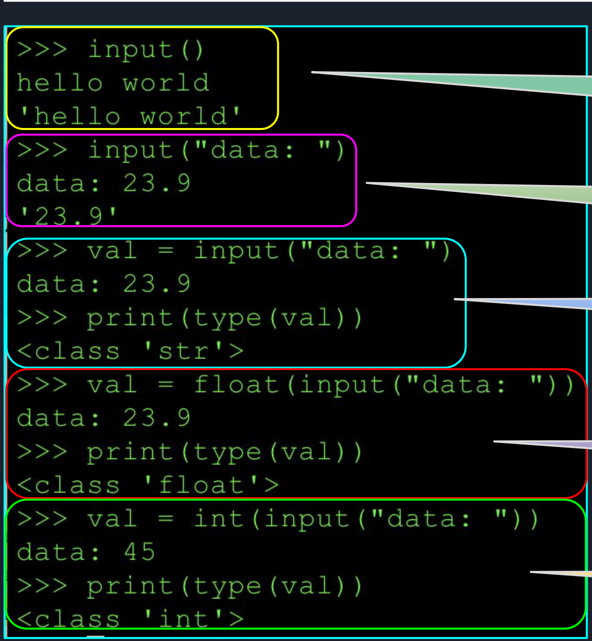


```
>>> print("Feng", "Chia", "University", sep="@", end="")
Feng@Chia@University>>> print("Feng", "Chia", "University", sep="@", end="\n")
Feng@Chia@University
>>> print("Feng", "Chia", "University", sep="@", end="\n\n")
Feng@Chia@University


>>>
```

input( )



```
Help on built-in function input in module builtins:

input(prompt=None, /)
    Read a string from standard input.  The trailing newline is stripped.

    The prompt string, if given, is printed to standard output without a
    trailing newline before reading input.

    If the user hits EOF (*nix: Ctrl-D, Windows: Ctrl-Z+Return), raise EOFError.
    On *nix systems, readline is used if available.
```



```
>>> input()
hello world
'hello world'
>>> input("data: ")
data: 23.9
'23.9'
>>> val = input("data: ")
data: 23.9
>>> print(type(val))
<class 'str'>
>>> val = float(input("data: "))
data: 23.9
>>> print(type(val))
<class 'float'>
>>> val = int(input("data: "))
data: 45
>>> print(type(val))
<class 'int'>
```

無提示訊息

有提示訊息，無資料型態轉換

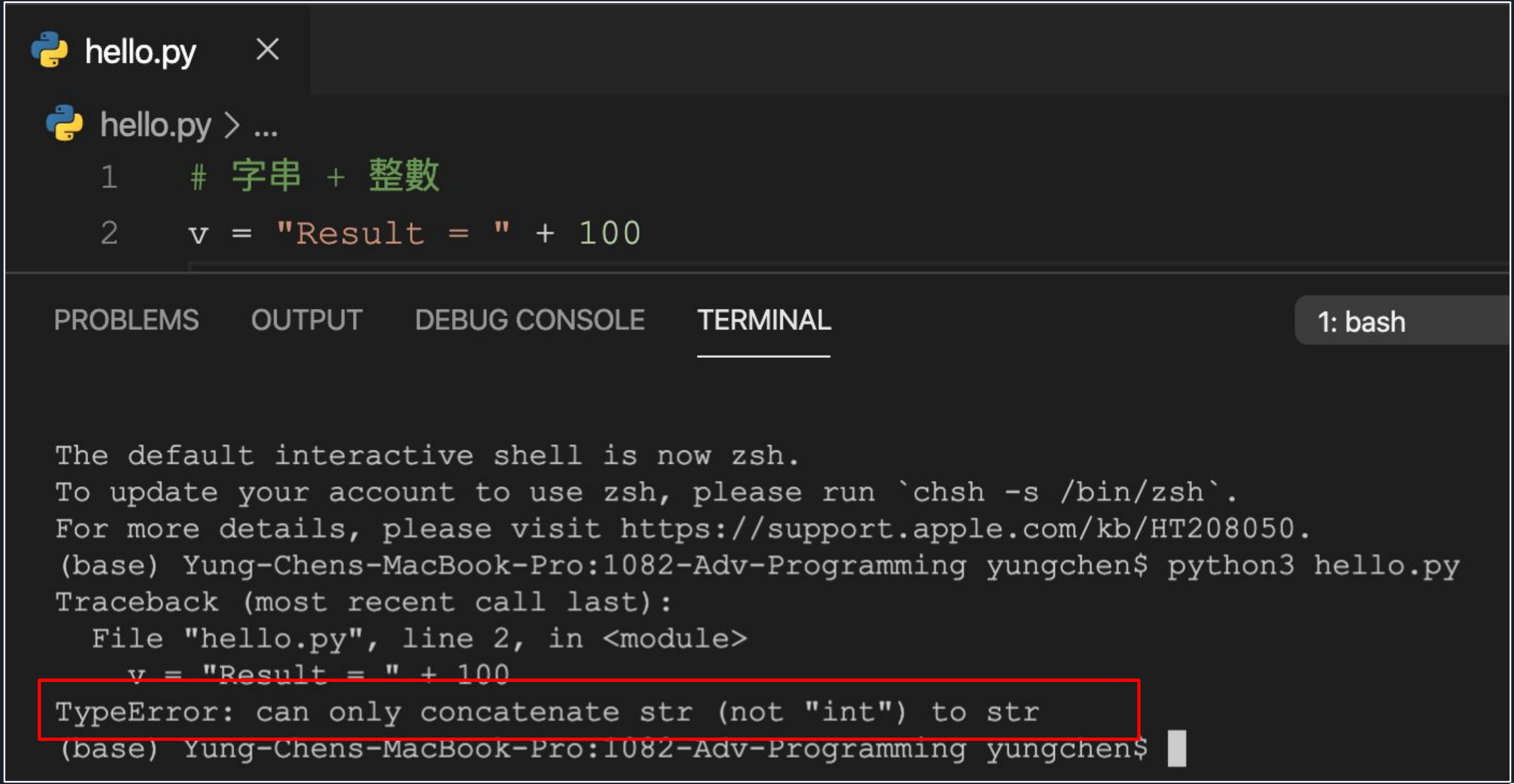
有提示訊息，接收後資料型態預設為 str

有提示訊息，接收後 str 轉成 float

有提示訊息，接收後 str 轉成 int

介紹

- Python 是一種動態型態 (Dynamic type) 語言
- 變數使用不需要先宣告資料型態
- Python直譯器會自動判斷變數應有 資料型態
- Python是一種強型態程式語言
    - Python裡使用的變數不會自動轉換變數的資料型態
        - v = “result = “ + 100 ⇐ 不能執行
        - v = “result=” + str(100) ⇐將整數轉成 String 即可



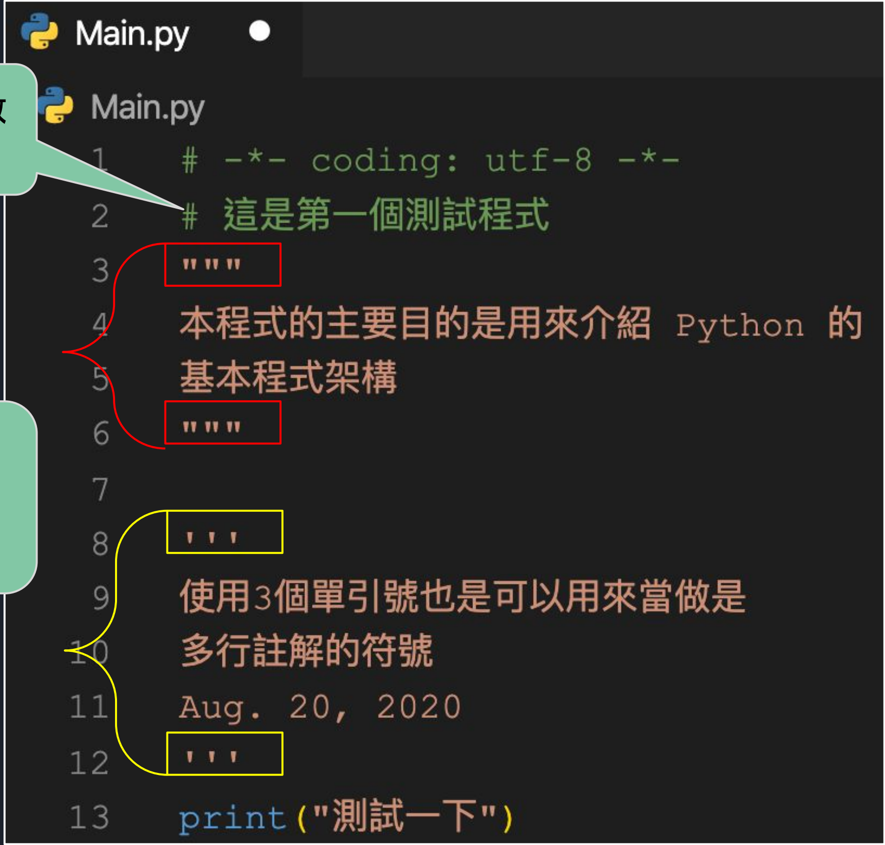
```python
# 字串 + 整數
v = "Result = " + 100
```

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Yung-Chens-MacBook-Pro:1082-Adv-Programming yungchen$ python3 hello.py
Traceback (most recent call last):
  File "hello.py", line 2, in <module>
    v = "Result = " + 100
TypeError: can only concatenate str (not "int") to str
(base) Yung-Chens-MacBook-Pro:1082-Adv-Programming yungchen$
```

程式註解

單一個 # 用來做單行註解之用

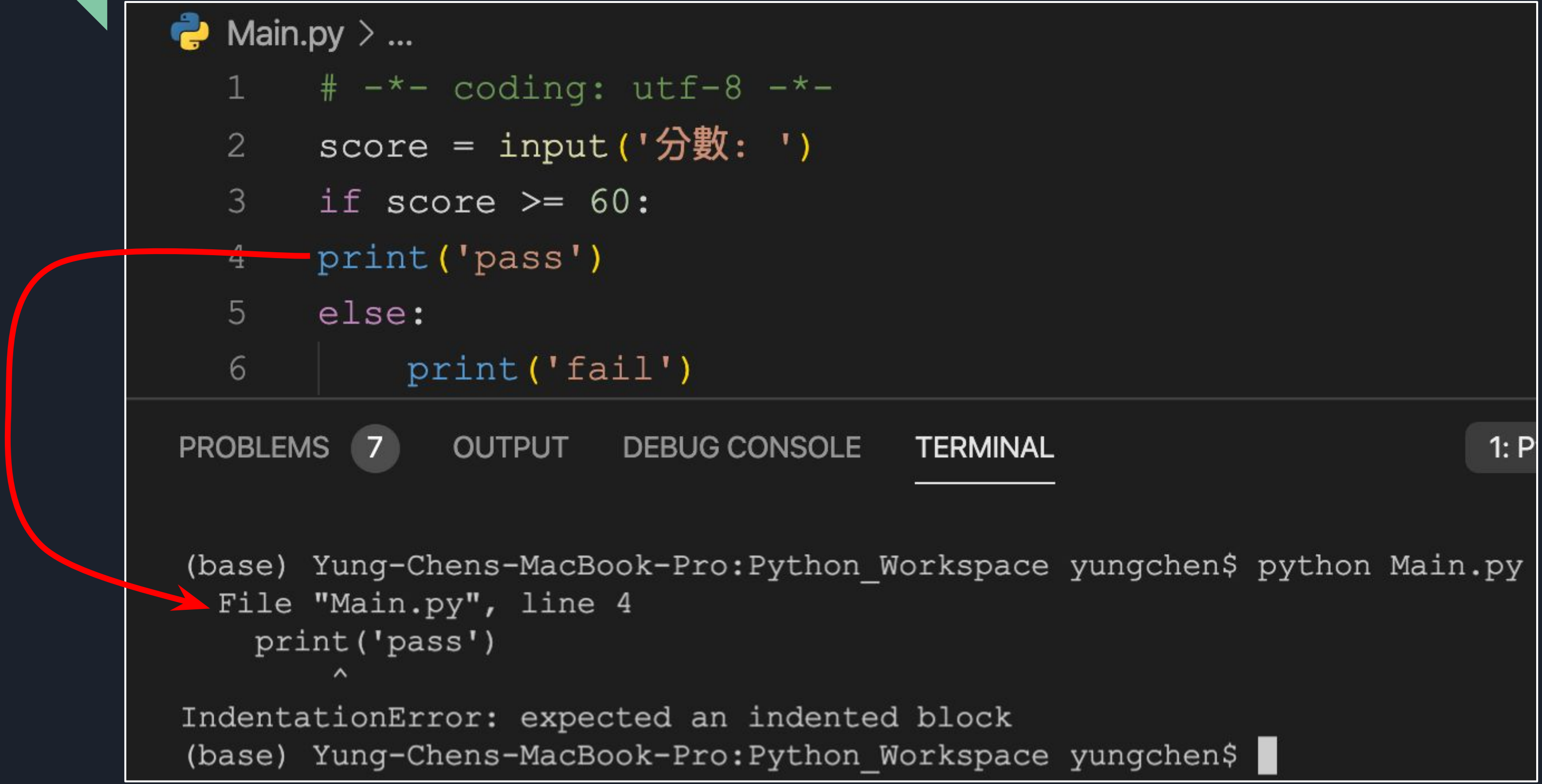
使用 連續3個 “ 雙引號或 3個單引號 ‘ 可以達到 多行註解的作用



```
# -*- coding: utf-8 -*-
# 這是第一個測試程式
"""
本程式的主要目的是用來介紹 Python 的
基本程式架構
"""

'''
使用3個單引號也是可以用來當做是
多行註解的符號
Aug. 20, 2020
'''
print("測試一下")
```

程式碼縮排

- Suite: Python 是用縮排來表示程式區塊 (Code Block)

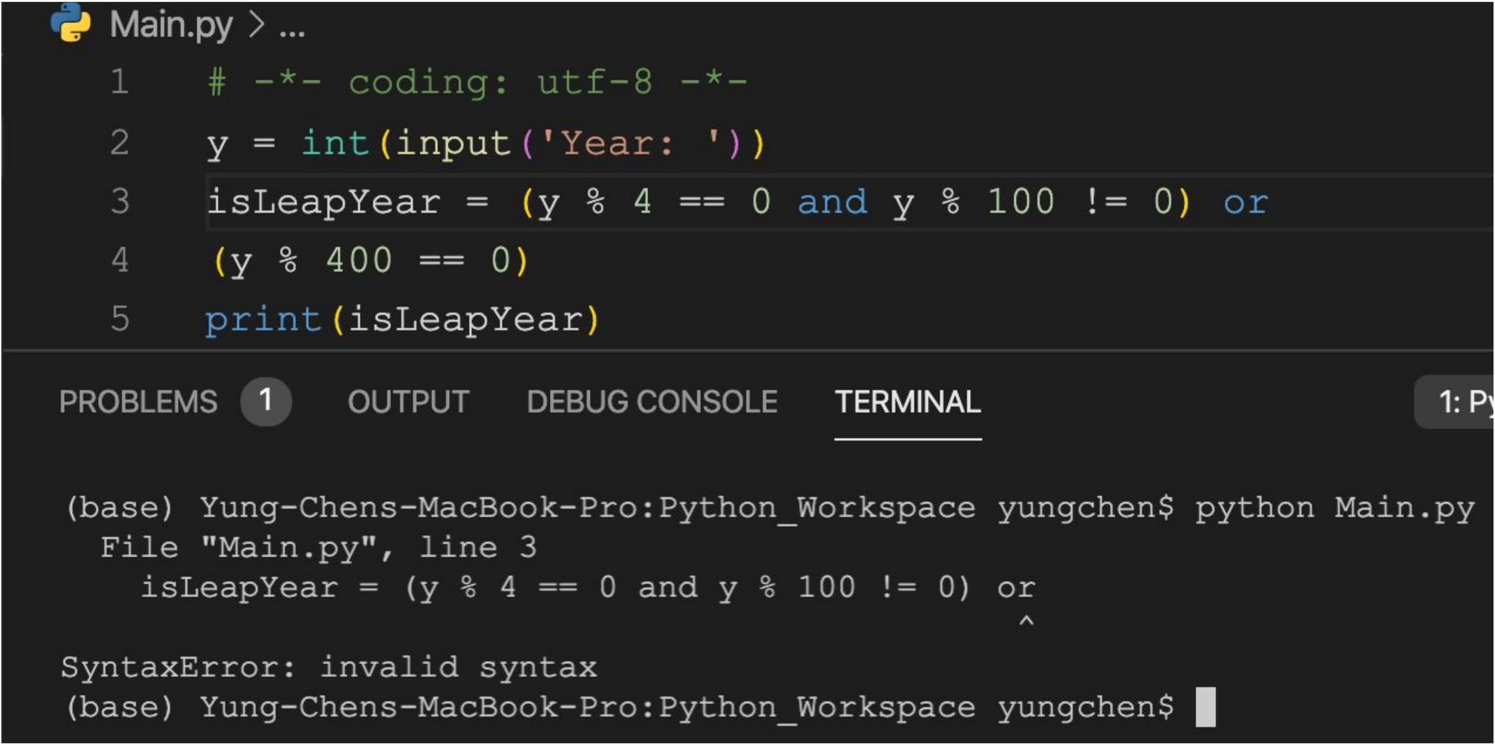


```
1   # -*- coding: utf-8 -*-
2   score = input('分數: ')
3   if score >= 60:
4   print('pass')
5   else:
6       print('fail')
```

沒縮排

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
  File "Main.py", line 4
    print('pass')
        ^
IndentationError: expected an indented block
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ █
```

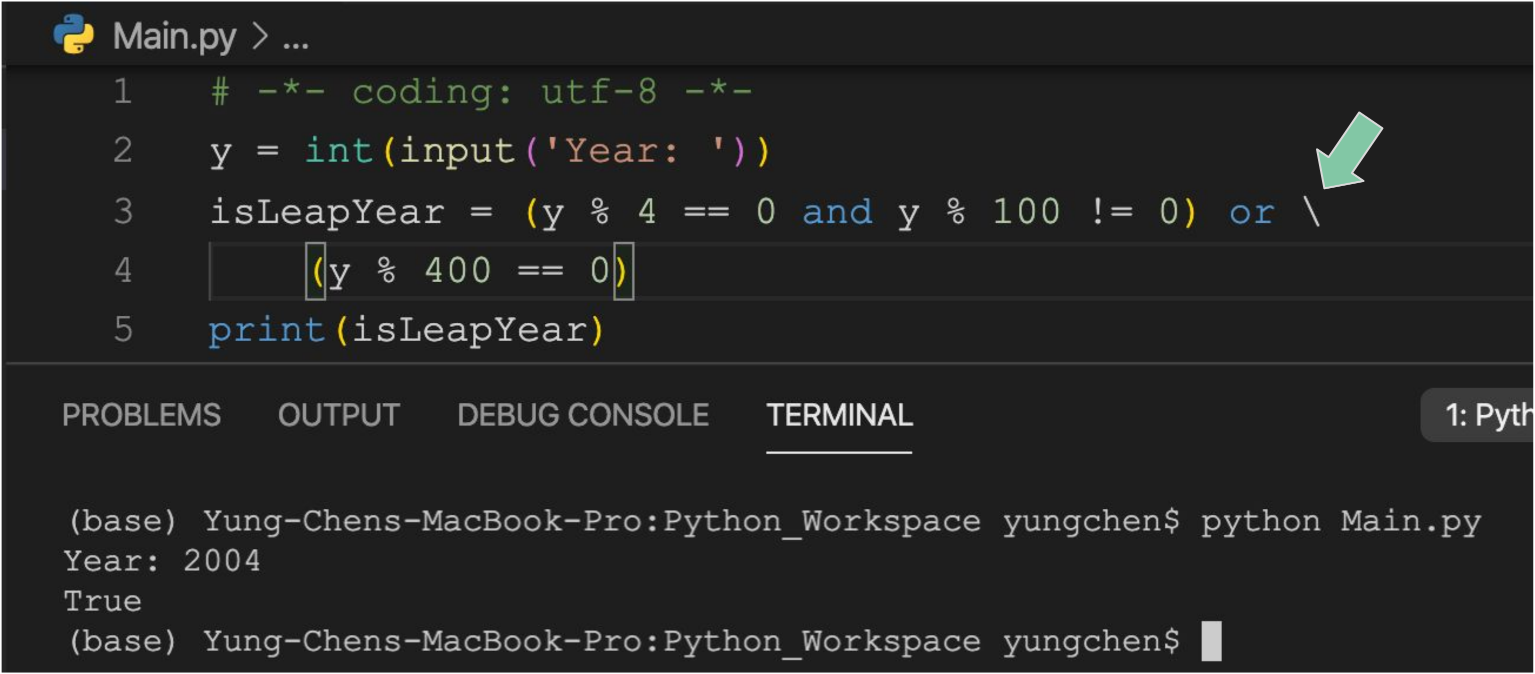
指令列的分行與合併



```python
1   # -*- coding: utf-8 -*-
2   y = int(input('Year: '))
3   isLeapYear = (y % 4 == 0 and y % 100 != 0) or
4   (y % 400 == 0)
5   print(isLeapYear)
```

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
  File "Main.py", line 3
    isLeapYear = (y % 4 == 0 and y % 100 != 0) or
                                                ^
SyntaxError: invalid syntax
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$
```

指令列的分行與合併

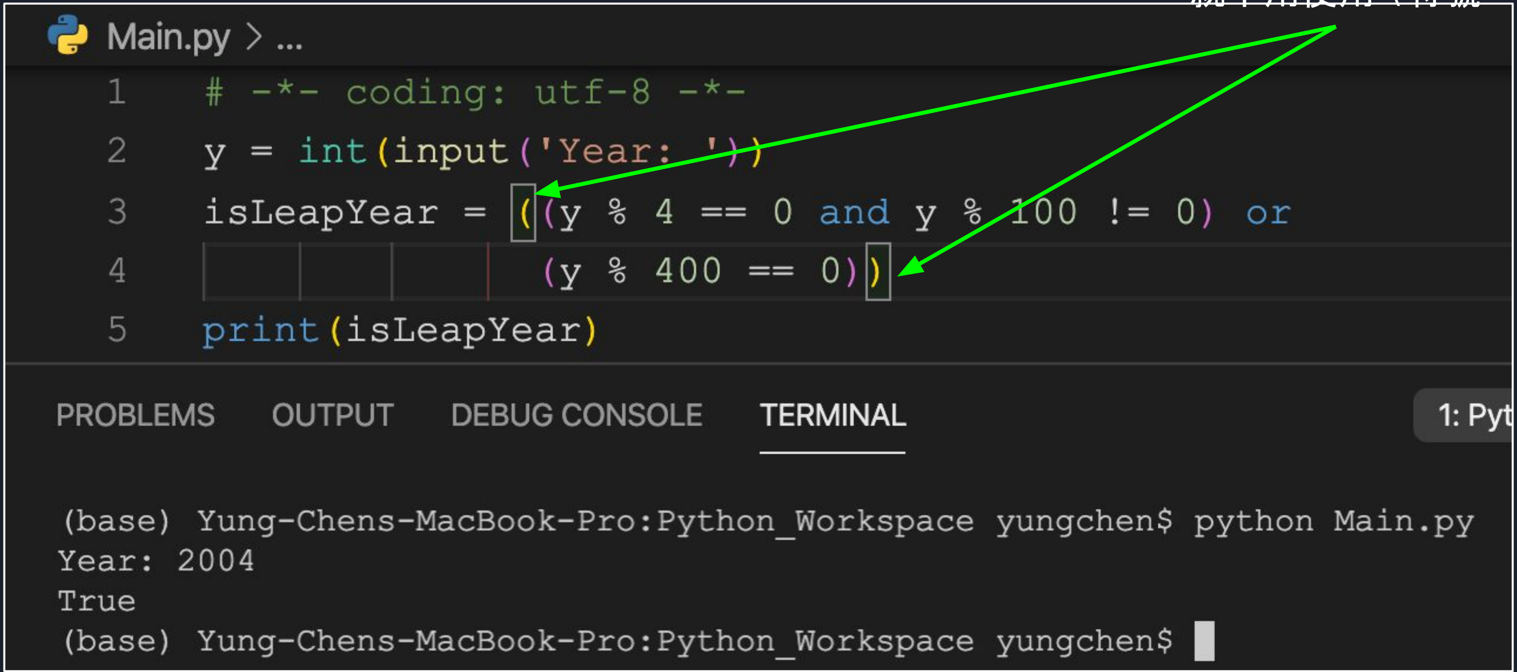


```python
# -*- coding: utf-8 -*-
y = int(input('Year: '))
isLeapYear = (y % 4 == 0 and y % 100 != 0) or \
    (y % 400 == 0)
print(isLeapYear)
```

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
Year: 2004
True
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ 
```

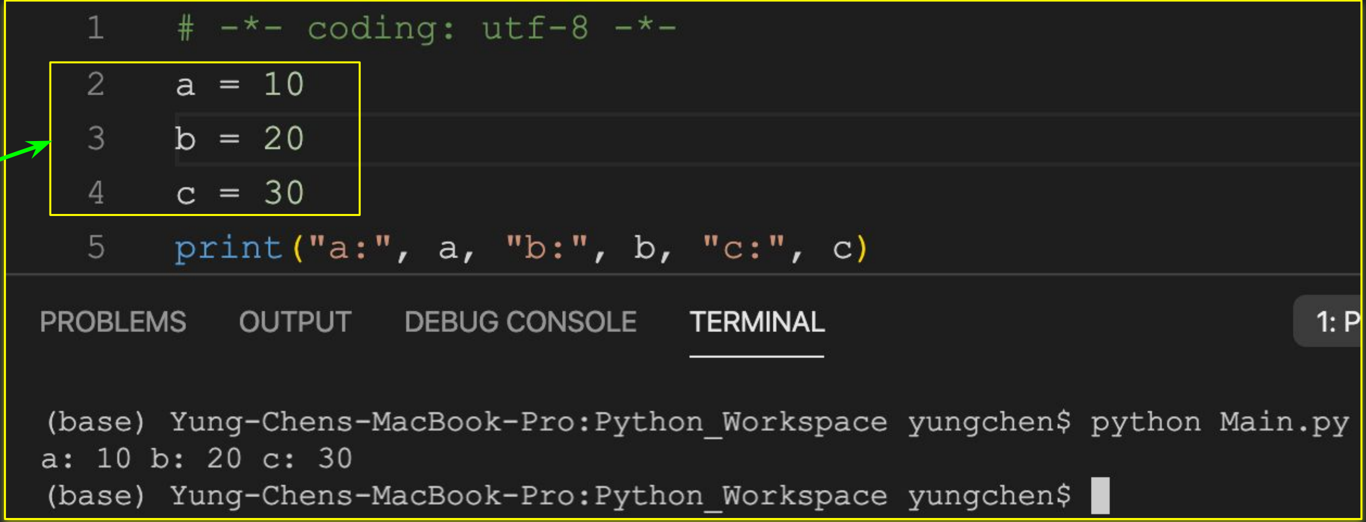
指令列的分行與合併

用括號把多行包起來
就不用使用 \ 符號



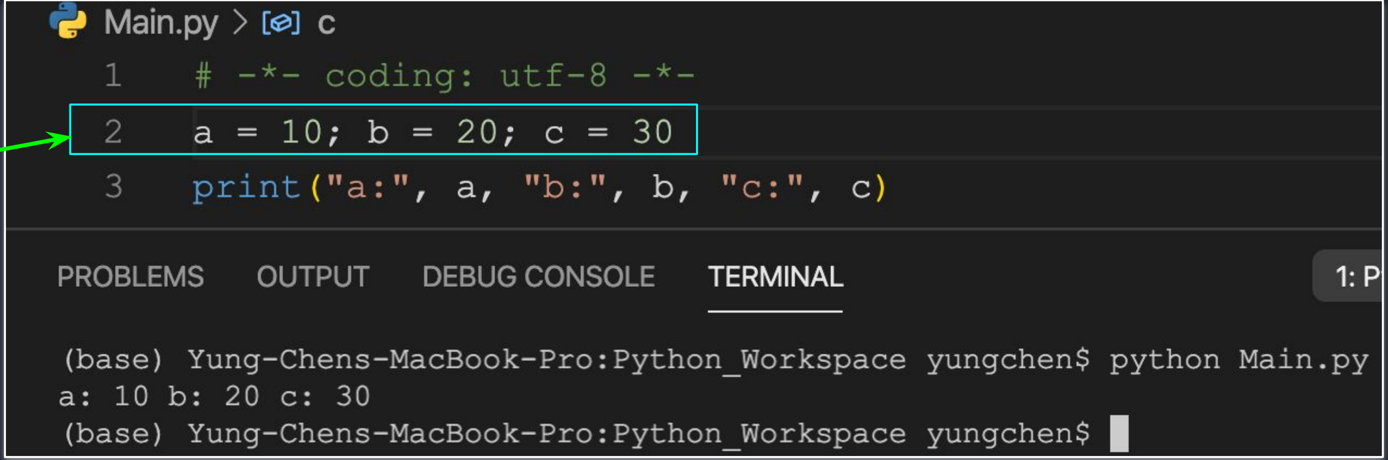
```python
# -*- coding: utf-8 -*-
y = int(input('Year: '))
isLeapYear = ((y % 4 == 0 and y % 100 != 0) or
              (y % 400 == 0))
print(isLeapYear)
```

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
Year: 2004
True
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ 
```



```python
# -*- coding: utf-8 -*-
a = 10
b = 20
c = 30
print("a:", a, "b:", b, "c:", c)
```

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
a: 10 b: 20 c: 30
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$
```



```python
# -*- coding: utf-8 -*-
a = 10; b = 20; c = 30
print("a:", a, "b:", b, "c:", c)
```

```
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$ python Main.py
a: 10 b: 20 c: 30
(base) Yung-Chens-MacBook-Pro:Python_Workspace yungchen$
```