



巨匠直播教學

APCS Python語法基礎班

# 資料儲存與型態

[www.pcschoolonline.com.tw](http://www.pcschoolonline.com.tw)

# 本堂教學重點

1. 變數資料型態
2. 數值型態
3. 字串型態
4. 集合型態

# 課程內容

## 1. 變數資料型態

1-1. 型態與記憶體位址

1-2. 數值型態

## 2. 字串型態

2-1. 建立字串

2-2. 索引操作

## 3. 集合型態

3-1. List & Tuple

3-2. Set

3-3. Dict

# 課程內容

## 1. 變數資料型態

1-1. 型態與記憶體位址

1-2. 數值型態

## 2. 字串型態

2-1. 建立字串

2-2. 索引操作

## 3. 集合型態

3-1. List & Tuple

3-2. Set

3-3. Dict

# 變數資料型態

## ◆ 變數資料型態

- ◆ Python 是動態型別，變數不必宣告固定的資料型態。
- ◆ 已使用的變數，可以改變來存放不同型態的資料

## ◆ 常用的變數型態

- ◆ `var1=123`      `#var1` 是整數，`int` 型態。
- ◆ `var2=3.14`      `#var2` 是浮點數，`float` 型態。
- ◆ `var3=True`      `#var3` 是布林值 `bool`，值為 `True` 或者 `False`。
- ◆ `var4='Hello'`      `#var4` 是字串 `str`，可用單引號或雙引號包起來。

# Python內建型態

## ◆ Python 常用內建資料型態

- ◆ 數字型態：整數(int)、布林值(bool)、浮點數(float) 及複數(complex)

- ◆ 文字型態：字串(str)、字元(char)

- ◆ 集合型態：tuple、list、set、dict

## ◆ Python中，所有的資料都是物件

# 變數資料型態

- ◆ 查詢變數的資料型態

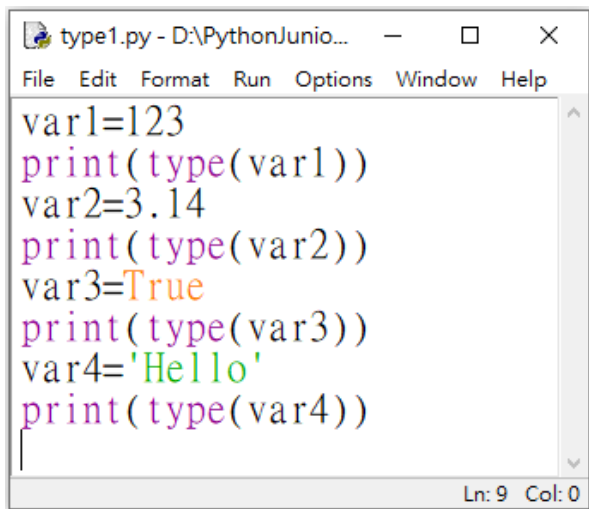
- ◆ 使用 `type()` 函式顯示。

- ◆ Python 變數中存放的是記憶體位址

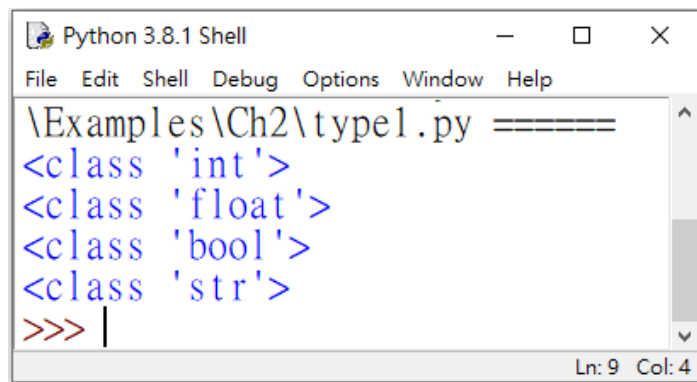
- ◆ 使用 `id()` 指令顯示記憶體位址

- ◆ 數值變數內容變更後會儲存於不同的記憶體位址。

# 程式範例



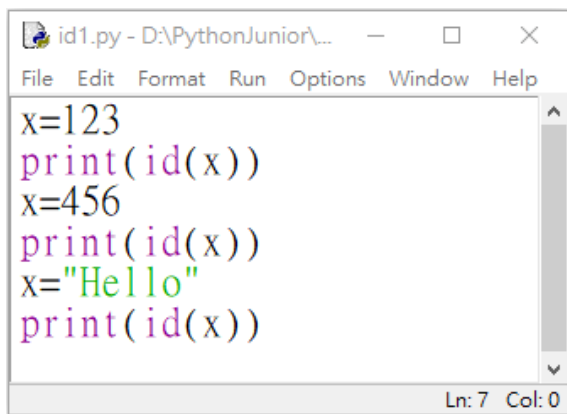
```
type1.py - D:\PythonJunio...  
File Edit Format Run Options Window Help  
var1=123  
print(type(var1))  
var2=3.14  
print(type(var2))  
var3=True  
print(type(var3))  
var4='Hello'  
print(type(var4))  
Ln: 9 Col: 0
```



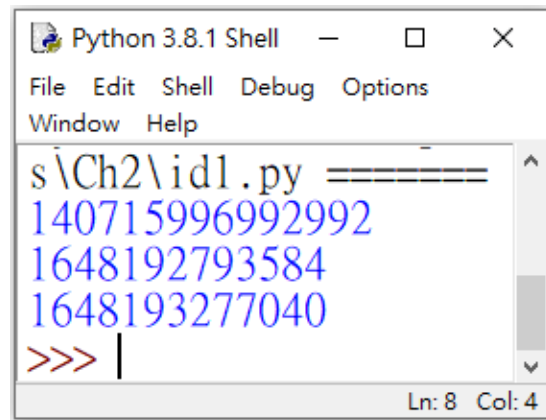
```
Python 3.8.1 Shell  
File Edit Shell Debug Options Window Help  
\\Examples\\Ch2\\type1.py =====  
<class 'int'>  
<class 'float'>  
<class 'bool'>  
<class 'str'>  
>>> |  
Ln: 9 Col: 4
```



# 程式範例



```
id1.py - D:\PythonJunior\...  
File Edit Format Run Options Window Help  
x=123  
print(id(x))  
x=456  
print(id(x))  
x="Hello"  
print(id(x))  
Ln: 7 Col: 0
```



```
Python 3.8.1 Shell  
File Edit Shell Debug Options  
Window Help  
s\Ch2\id1.py ==  
140715996992992  
1648192793584  
1648193277040  
>>> |  
Ln: 8 Col: 4
```

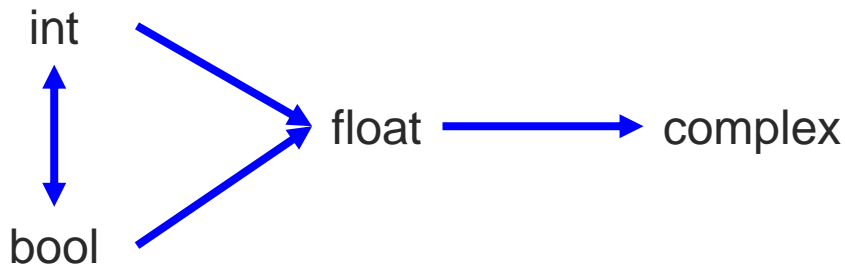
# Python數字型態

類型	類別	範例	說明
整數	int	i = 123 i = 0b1111011 i = 0o173 i = 0x7B	預設為10進位表達，可帶正負號(+, -) 2進位表達 8進位表達 16進位表達
浮點數	float	pi = 3.14 na = 6.02e23	小數點 科學符號 $6.02e23 = 6.02 \times 10^{23}$
布林值	bool	b1 = True b2 = False	是非值，可與其他數值轉換 非 0 為 True，0 為 False
複數	complex	c = 1.2+3.4j	real實部為1.2，imaginary虛部為3.4

# 數值運算自動轉型

## ◆ 數值運算支援自動轉型

- ◆ 二元運算時兩個運算元型態不同
- ◆ 型態較小的資料會自動晉升為較大的資料型態後再運算

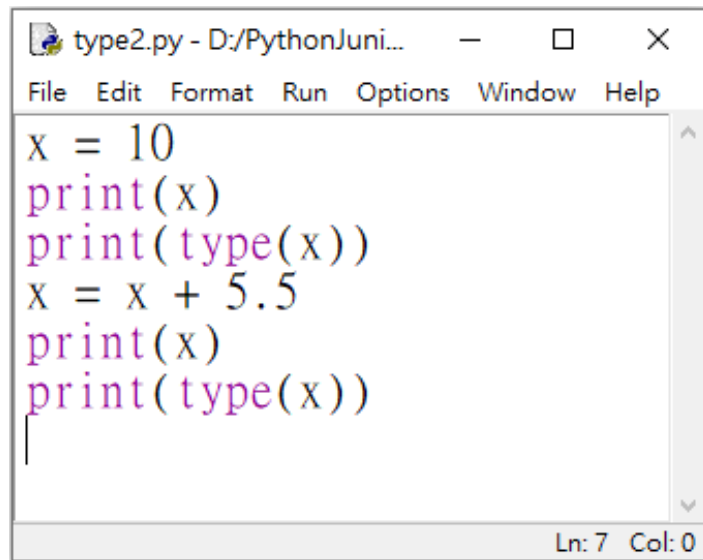


# 數值運算手動轉型

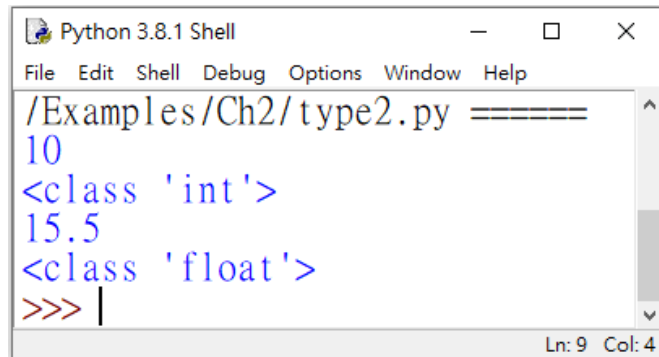
## ◆ 數值運算手動轉型

- ◆ 將型態較大的資料轉換為較小的資料型態
- ◆ `int( )` 函式將數值轉換為整數
- ◆ `float( )` 函式將數值轉換為浮點數
- ◆ `bool( )` 函式將數值轉換為布林值

# 程式範例

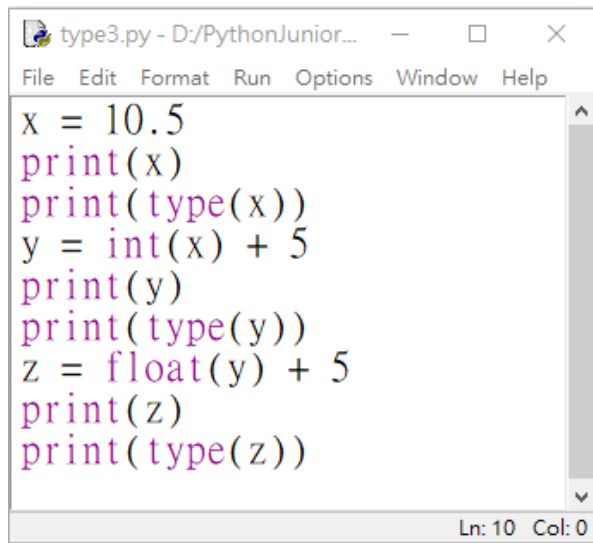


```
type2.py - D:/PythonJuni...  
File Edit Format Run Options Window Help  
x = 10  
print(x)  
print(type(x))  
x = x + 5.5  
print(x)  
print(type(x))  
Ln: 7 Col: 0
```

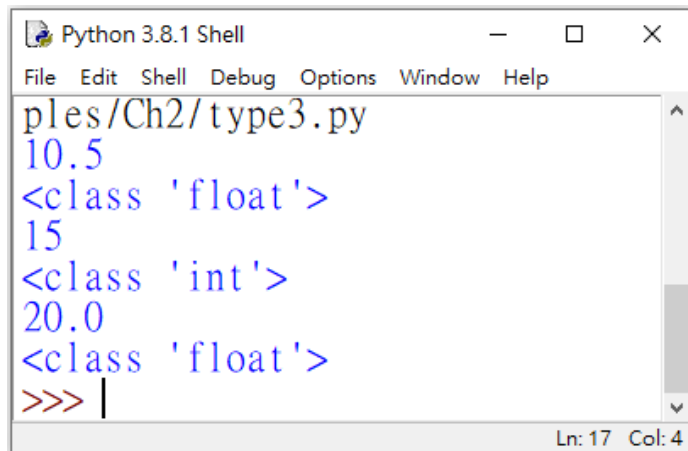


```
Python 3.8.1 Shell  
File Edit Shell Debug Options Window Help  
/Examples/Ch2/type2.py =====  
10  
<class 'int'>  
15.5  
<class 'float'>  
>>> |  
Ln: 9 Col: 4
```

# 程式範例

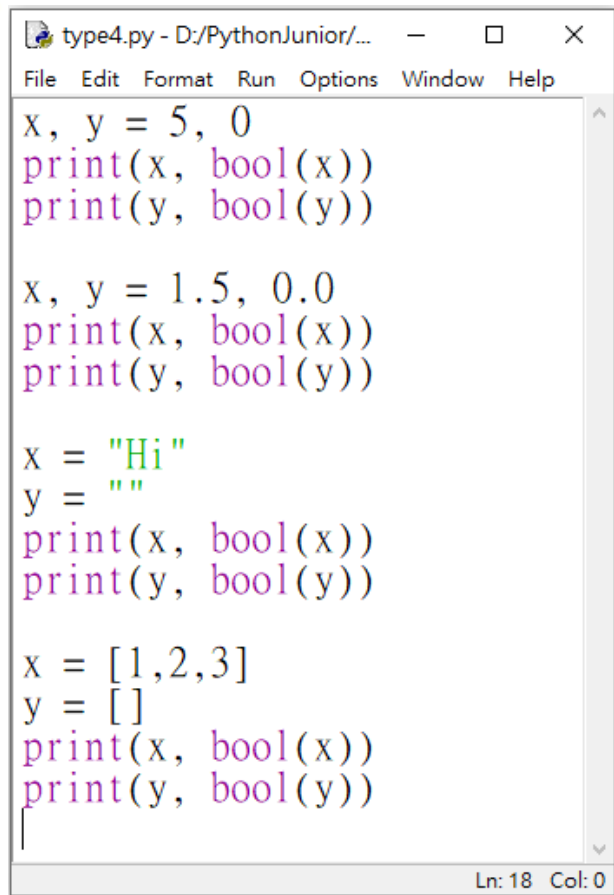


```
type3.py - D:/PythonJunior...  
File Edit Format Run Options Window Help  
x = 10.5  
print(x)  
print(type(x))  
y = int(x) + 5  
print(y)  
print(type(y))  
z = float(y) + 5  
print(z)  
print(type(z))  
Ln: 10 Col: 0
```

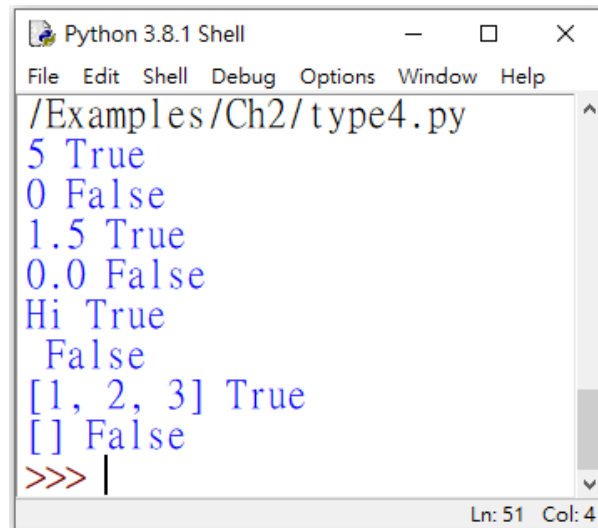


```
Python 3.8.1 Shell  
File Edit Shell Debug Options Window Help  
ples/Ch2/type3.py  
10.5  
<class 'float'>  
15  
<class 'int'>  
20.0  
<class 'float'>  
>>> |  
Ln: 17 Col: 4
```

# 程式範例



```
type4.py - D:/PythonJunior/...  
File Edit Format Run Options Window Help  
  
x, y = 5, 0  
print(x, bool(x))  
print(y, bool(y))  
  
x, y = 1.5, 0.0  
print(x, bool(x))  
print(y, bool(y))  
  
x = "Hi"  
y = ""  
print(x, bool(x))  
print(y, bool(y))  
  
x = [1,2,3]  
y = []  
print(x, bool(x))  
print(y, bool(y))  
  
Ln: 18 Col: 0
```



```
Python 3.8.1 Shell  
File Edit Shell Debug Options Window Help  
  
/Examples/Ch2/type4.py  
5 True  
0 False  
1.5 True  
0.0 False  
Hi True  
False  
[1, 2, 3] True  
[] False  
>>> |  
  
Ln: 51 Col: 4
```

Q：下列哪個函式可以取得變數的型態？

- a) `int()`
- b) `type()`
- c) `id()`
- d) `str()`



**Q : `print(type(3.14))` 顯示結果為何？**

- a) 3.140000
- b) 3.14
- c) 3
- d) `<class 'float'>`

# 課程內容

## 1. 變數資料型態

1-1. 型態與記憶體位址

1-2. 數值型態

## 2. 字串型態

2-1. 建立字串

2-2. 索引操作

## 3. 集合型態

3-1. List & Tuple

3-2. Set

3-3. Dict

# 字串宣告

## ◆ 字串用單引號或雙引號夾起來

### ◆ 引號需成對使用，不可混用

- 內容如包含單引號，可用雙引號夾起來，反之亦然

### ◆ 內容中包含Python中有意義的符號，使用跳脫字元(\)

### ◆ 分行輸入：行結尾使用 \

### ◆ 跨行字串：使用成對的三個單引號或雙引號

# 常見跳脫字元

跳脫字元	意義
\\	表示 \ 字元。
\'	表示 ' 字元。
\"	表示 " 字元。
\n	換行 ( newline ) 。
\t	水平跳格，相當於按鍵盤的Tab鍵。
\b	退一格 ( backspace ) 。
\r	返回 ( carriage return ) ，游標移至行首。

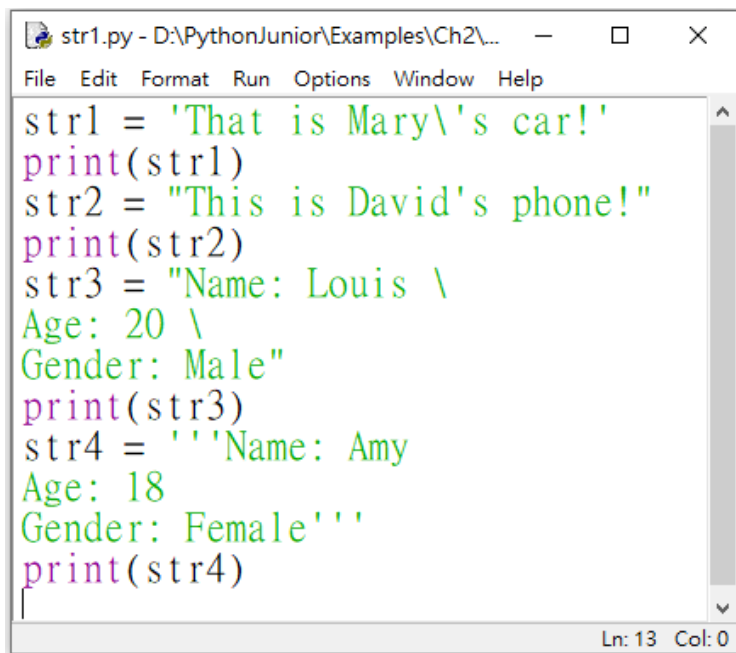
# 原始字串 ( Raw String )

## ◆ 原始字串 ( Raw String )

- ◆ 字串符號前加上R 或r
- ◆ 字串中「\」不再表示跳脫字元
  - 可以用來方便地表示Windows 系統下的路徑

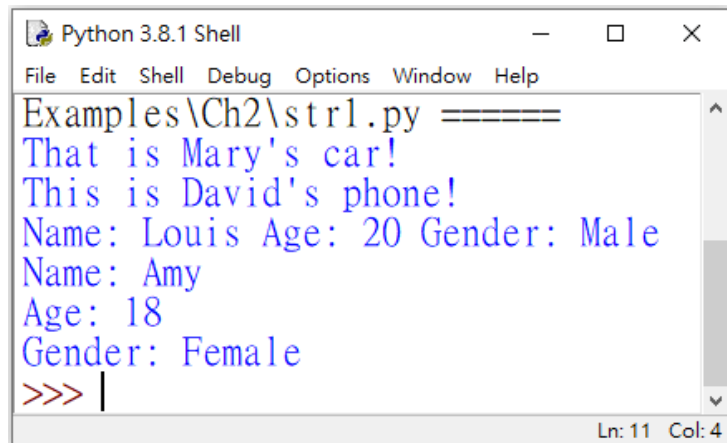
path='c:\\python36\\Scripts' ➡ path=r'C:\python36\Scripts'

# 程式範例



```
str1 = 'That is Mary\'s car!'
print(str1)
str2 = "This is David's phone!"
print(str2)
str3 = "Name: Louis \
Age: 20 \
Gender: Male"
print(str3)
str4 = '''Name: Amy
Age: 18
Gender: Female'''
print(str4)
```

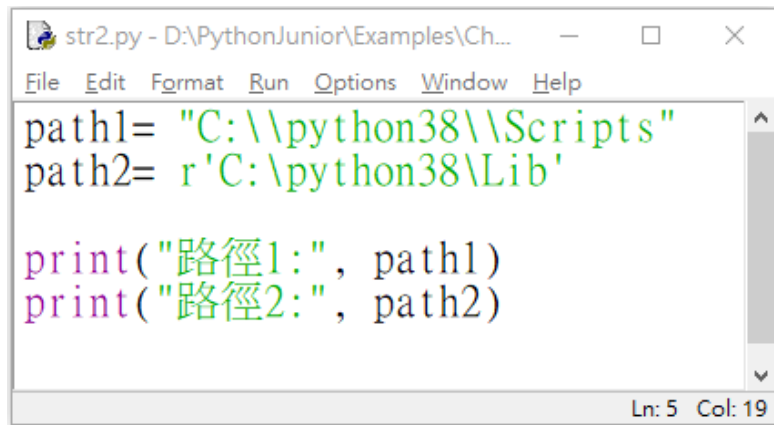
Ln: 13 Col: 0



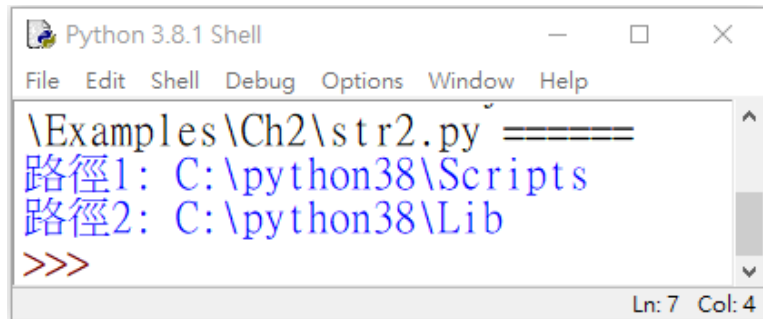
```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
Examples\Ch2\str1.py =====
That is Mary's car!
This is David's phone!
Name: Louis Age: 20 Gender: Male
Name: Amy
Age: 18
Gender: Female
>>> |
```

Ln: 11 Col: 4

# 程式範例



```
str2.py - D:\PythonJunior\Examples\Ch...  
File Edit Format Run Options Window Help  
path1= "C:\\python38\\Scripts"  
path2= r'C:\python38\Lib'  
  
print("路徑1:", path1)  
print("路徑2:", path2)  
  
Ln: 5 Col: 19
```

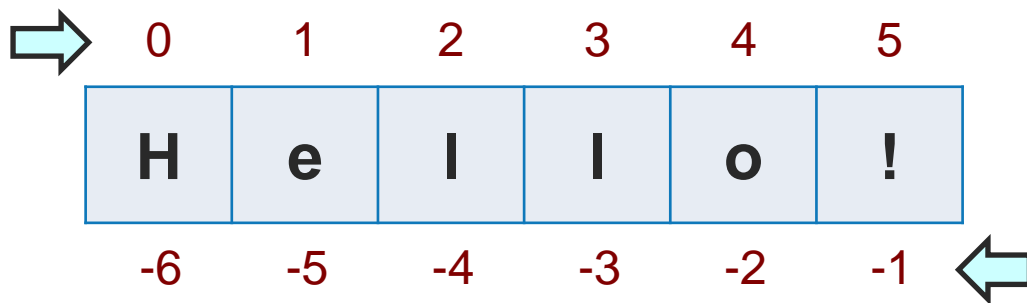


```
Python 3.8.1 Shell  
File Edit Shell Debug Options Window Help  
\\Examples\\Ch2\\str2.py =====  
路徑1: C:\python38\Scripts  
路徑2: C:\python38\Lib  
>>>  
  
Ln: 7 Col: 4
```

# 字串索引

## ◆ 字元依照順序排序：

- ◆ 第一個字索引編號為 0，第二個字索引編號為 1
- ◆ 最後一個字索引編號為 -1，倒數第二個字索引編號為 -2





# 字串索引取值

## ◆ str[n]

- ◆  $n$  為整數，取得字串中索引欄位  $n$  的字元。
- ◆  $n > 0$ ，由左向右，編號由 0 開始遞增。
- ◆  $n < 0$ ，由右向左，編號由 -1 開始遞減。

0	1	2	3	4	5
H	e	l	l	o	!
-6	-5	-4	-3	-2	-1

# 字串部分取值 Slice

## ◆ str[n : m]

◆ n 與 m 均為整數，取得字串中索引欄位 n 到索引欄位 m-1範圍內的子字串

- n 代表起始位置索引值，m 代表結束位置索引值(不包含這個位置)

◆ n 與 m 可省略其一

- n 省略表示起始位置開始
- m 省略取到最後一個字元

0	1	2	3	4	5
H	e	l	l	o	!
-6	-5	-4	-3	-2	-1

# 字串反轉

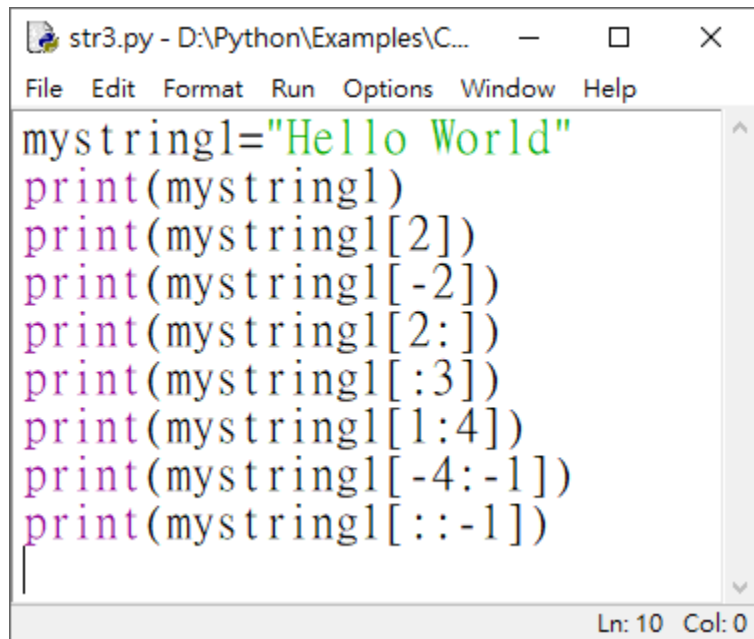
◆ `str[::-1]`

◆ Python提供一個便捷的字串反轉語法

H	e	l	l	o	!
---	---	---	---	---	---

!	o	l	l	e	H
---	---	---	---	---	---

# 程式範例



```
str3.py - D:\Python\Examples\C...  -  □  ×  
File Edit Format Run Options Window Help  
mystring1="Hello World"  
print(mystring1)  
print(mystring1[2])  
print(mystring1[-2])  
print(mystring1[2:])  
print(mystring1[:3])  
print(mystring1[1:4])  
print(mystring1[-4:-1])  
print(mystring1[::-1])  
Ln: 10 Col: 0
```

# 字串索引取值

```
mystring1="Hello World"
```

```
print(mystring1[2])
```

```
print(mystring1[-2])
```

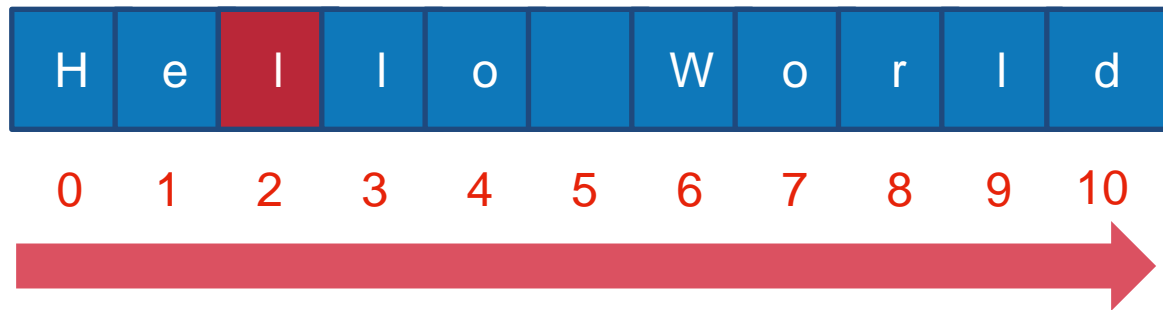
```
print(mystring1[2:])
```

```
print(mystring1[:3])
```

```
print(mystring1[1:4])
```

```
print(mystring1[-4:-1])
```

```
print(mystring1[::-1])
```



# 字串索引取值

```
mystring1="Hello World"
```

```
print(mystring1[2])
```

```
print(mystring1[-2])
```

```
print(mystring1[2:])
```

```
print(mystring1[:3])
```

```
print(mystring1[1:4])
```

```
print(mystring1[-4:-1])
```

```
print(mystring1[::-1])
```



# 字串部分取值

```
mystring1="Hello World"
```

```
print(mystring1[2])
```

```
print(mystring1[-2])
```

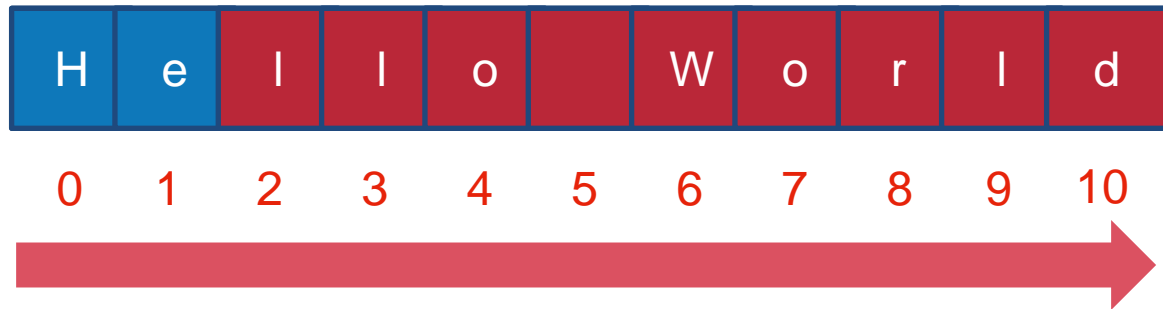
```
print(mystring1[2:])
```

```
print(mystring1[:3])
```

```
print(mystring1[1:4])
```

```
print(mystring1[-4:-1])
```

```
print(mystring1[::-1])
```



# 字串部分取值

```
mystring1="Hello World"
```

```
print(mystring1[2])
```

```
print(mystring1[-2])
```

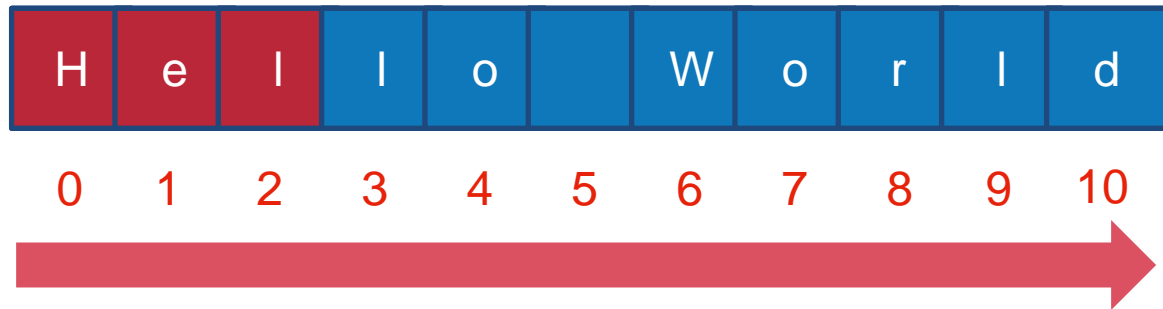
```
print(mystring1[2:])
```

```
print(mystring1[:3])
```

```
print(mystring1[1:4])
```

```
print(mystring1[-4:-1])
```

```
print(mystring1[::-1])
```





# 字串部分取值

```
mystring1="Hello World"
```

```
print(mystring1[2])
```

```
print(mystring1[-2])
```

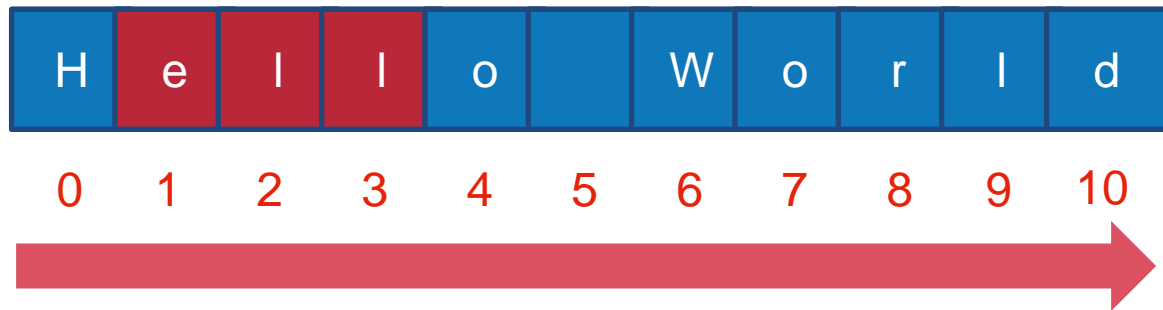
```
print(mystring1[2:])
```

```
print(mystring1[:3])
```

```
print(mystring1[1:4])
```

```
print(mystring1[-4:-1])
```

```
print(mystring1[::-1])
```



# 字串部分取值

```
mystring1="Hello World"  
print(mystring1[2])  
print(mystring1[-2])  
print(mystring1[2:])  
print(mystring1[:3])  
print(mystring1[1:4])  
print(mystring1[-4:-1])  
print(mystring1[::-1])
```



# 字串反轉

```
mystring1="Hello World"  
print(mystring1[2])  
print(mystring1[-2])  
print(mystring1[2:])  
print(mystring1[:3])  
print(mystring1[1:4])  
print(mystring1[-4:-1])  
print(mystring1[::-1])
```

H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10
d	l	r	o	W		o	l	l	e	H
0	1	2	3	4	5	6	7	8	9	10

**Q：** 下列哪個跳脫字元可跳至下一行輸出？

a) `\n`

b) `\t`

c) `\\`

d) `\b`

Q : `print("C:\\temp")` 輸出結果為何？

- a) `"C:\\temp"`
- b) `C:\\temp`
- c) `C:\temp`
- d) `C: emp`

## 練習：字串練習

- ◆ 某字串由前面讀取與從後面讀取結果相同稱為迴文 (Palindrome)
- ◆ 請撰寫一個程式，檢查使用者輸入的字串是否為迴文

```
if (...):
```

```
...
```

```
else:
```

```
...
```

# 練習：字串練習

- ◆ 台灣身份證字號共十碼，包括一個大寫英文字母與九個阿拉伯數字
  - ◇ 首碼英文代表出生登記的戶籍地，如下表
  - ◇ 首數字表示性別，男性為1、女性為2
- ◆ 撰寫一個程式
  - ◇ 輸入身分證字號
  - ◇ 判斷性別

代碼	出生地	代碼	出生地	代碼	出生地	代碼	出生地
A	臺北市	G	宜蘭縣	M	南投縣	T	屏東縣
B	臺中市	H	桃園市	N	彰化縣	U	花蓮縣
C	基隆市	I	嘉義市	O	新竹市	V	臺東縣
D	臺南市	J	新竹縣	P	雲林縣	W	金門縣
E	高雄市	K	苗栗縣	Q	嘉義縣	X	澎湖縣
F	新北市	L	臺中縣	R	臺南縣	Y	陽明山
				S	高雄縣	Z	連江縣

# 課程內容

## 1. 變數資料型態

1-1. 型態與記憶體位址

1-2. 數值型態

## 2. 字串型態

2-1. 建立字串

2-2. 索引操作

## 3. 集合型態

3-1. List & Tuple

3-2. Set

3-3. Dict



# Python集合資料

## ◆ 用一個變數儲存多筆資料

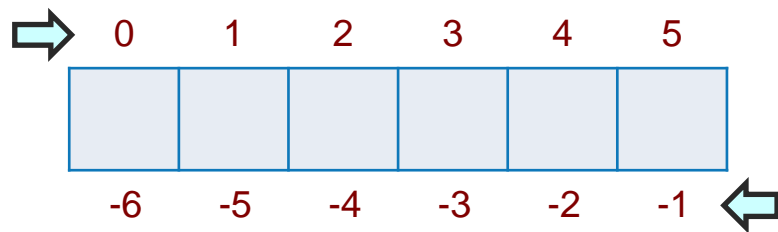
資料型態	說明
list	變動長度的有序集合
tuple	元素不可變更的List
set	元素不可重複的無序集合
dict	鍵值對(Key Value Pair)資料集合

# 內建序列 list / tuple

## ◆ list / tuple

### ◇ Python 內建序列集合

- 元素一個接著一個儲存
- 使用 [索引] 讀取元素內容
  - ▶ 索引值為正數，編號由 0 開始遞增，資料由左至右讀取
  - ▶ 索引值為負數，編號由 -1 開始遞減，資料由右至左讀取



# tuple

## ◆ tuple

- ◆ 使用()小括號將元素包起來
- ◆ 元素之間以，隔開
- ◆ 元素可以不同類型的值或集合
- ◆ 資料不可變更
  - 只能對其使用索引或部分取值操作。
  - 不能使用類似List的函數操作

```
t = (1, 2, 3)
```

```
print(t[0]) ➡ 1
```

```
print(t[-2]) ➡ 2
```

```
t[1]  4
```

# list

## ◆ list

◆ 使用[]中括號將元素包起來

◆ 元素之間以 , 隔開

◆ 元素可以不同類型的值或集合

◆ 資料可以變更

```
l = [1, 'Python', (2, 3)]
```

```
print(l[2]) ➡ (2, 3)
```

```
print(l[-2]) ➡ 'Python'
```

```
l[1] = 4 😊
```

```
print(l[1]) ➡ 4
```

# list / tuple 部分取值(slice)

◆ `list [ n : m ]` : 傳回原序列之部分子集合

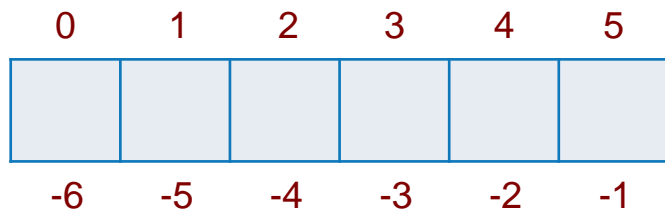
◆ `n` 與 `m` 是整數，可以省略。

◆ `n` 為起始位置索引

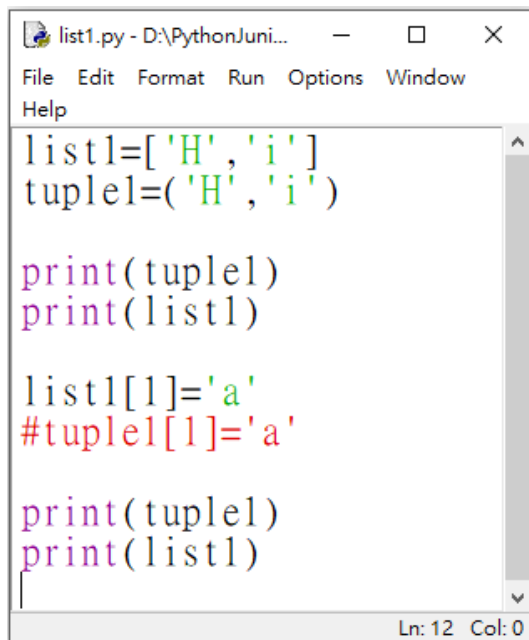
- 省略時表示起始位置為0

◆ `m` 為結束位置索引值

- 取值時不包含這個位置的值
- 省略時表示取至最後一個值



# 程式範例



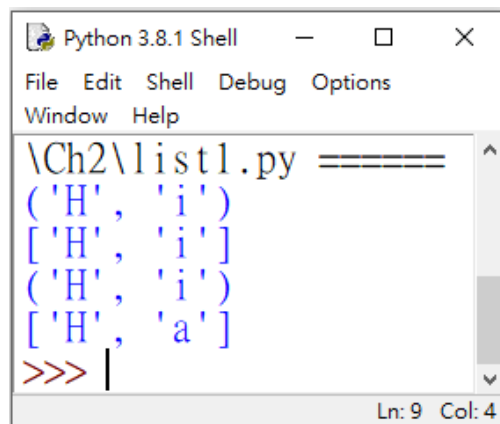
```
list1=['H','i']
tuple1=('H','i')

print(tuple1)
print(list1)

list1[1]='a'
#tuple1[1]='a'

print(tuple1)
print(list1)
```

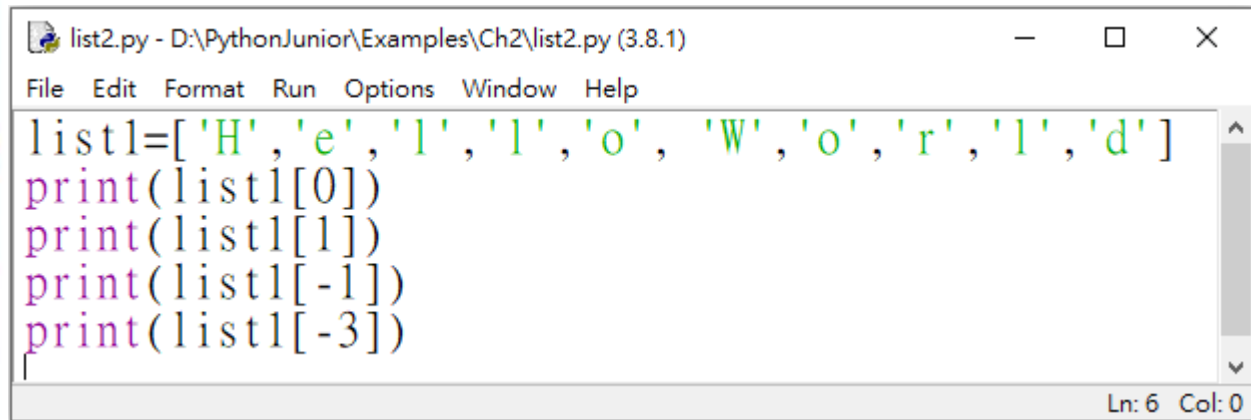
Ln: 12 Col: 0



```
\Ch2\list1.py =====
('H', 'i')
['H', 'i']
('H', 'i')
['H', 'a']
>>> |
```

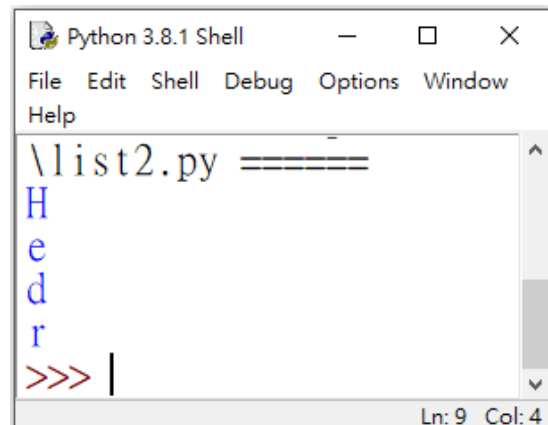
Ln: 9 Col: 4

# 程式範例



```
list1=['H','e','l','l','o',' ','W','o','r','l','d']
print(list1[0])
print(list1[1])
print(list1[-1])
print(list1[-3])
```

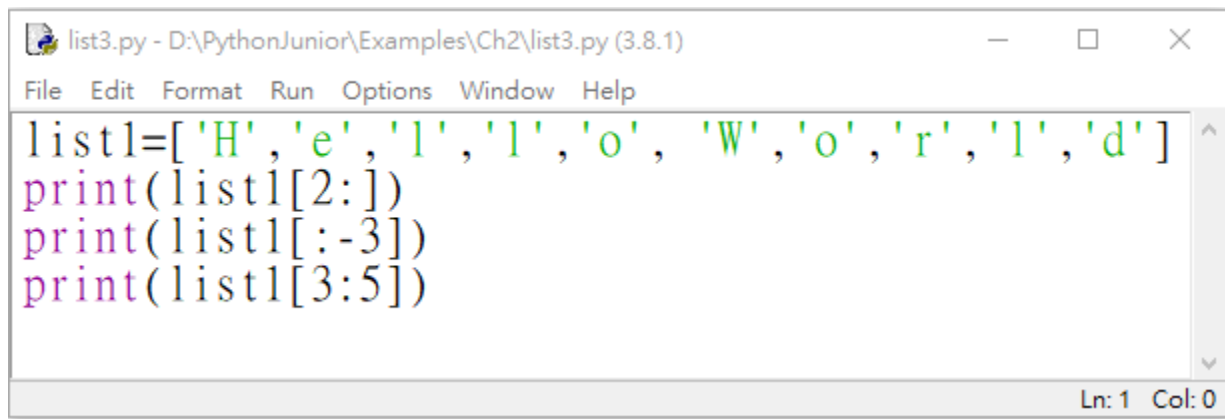
Ln: 6 Col: 0



```
\list2.py =====
H
e
d
r
>>> |
```

Ln: 9 Col: 4

# 程式範例



The screenshot shows a Python IDE window titled "list3.py - D:\PythonJunior\Examples\Ch2\list3.py (3.8.1)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code editor contains the following Python code:

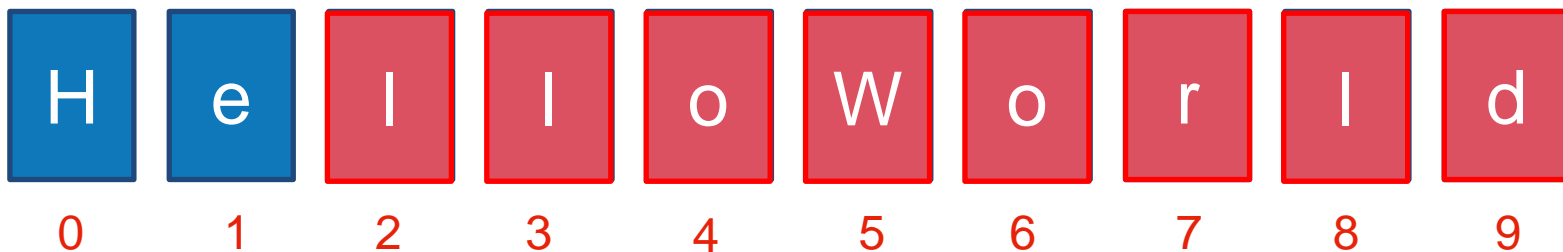
```
list1=['H','e','l','l','o', 'W','o','r','l','d']  
print(list1[2:])  
print(list1[:-3])  
print(list1[3:5])
```

The status bar at the bottom right indicates "Ln: 1 Col: 0".



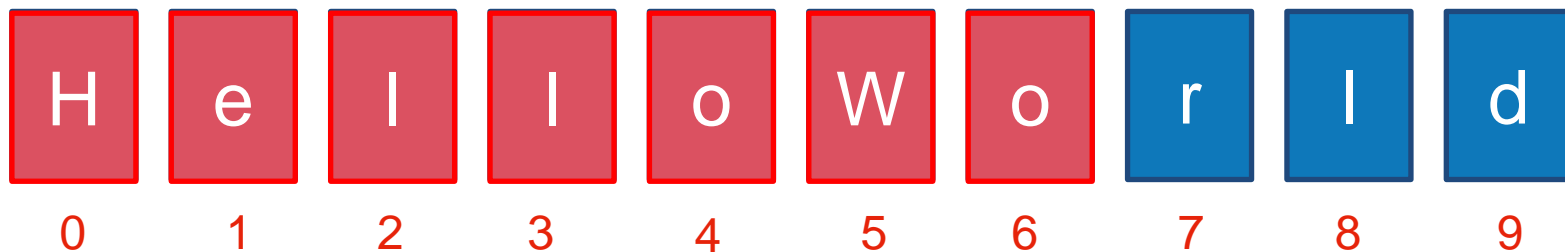
## 部分取值

```
list1=['H','e','l','l','o', 'W','o','r','l','d']  
print(list1[2:])
```



## 部分取值

```
list1=['H','e','l','l','o','W','o','r','l','d']  
print(list1[:-3])
```



## 部分取值

```
list1=['H','e','l','l','o','W','o','r','l','d']  
print(list1[3:5])
```



# list修改元素內容

## ◆ 修改元素內容

◆ 以索引值指定 list 中的某筆元素為新的資料

◆ `list1[2]=123`

# list 刪除元素內容

- ◆ 使用 del 關鍵字，刪除 list 中指定位置元素

- ◆ del list[n]

- ◆ 使用del 刪除 list 中部份元素：

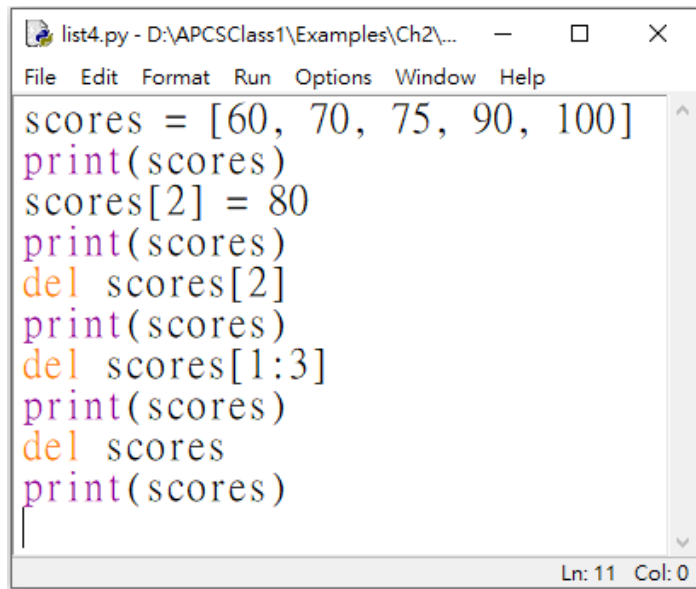
- ◆ del list[n:m]

- n省略表示刪除起始位置為0
    - m省略表示刪除至最後一個值

- ◆ 使用 del 刪除整個 list：

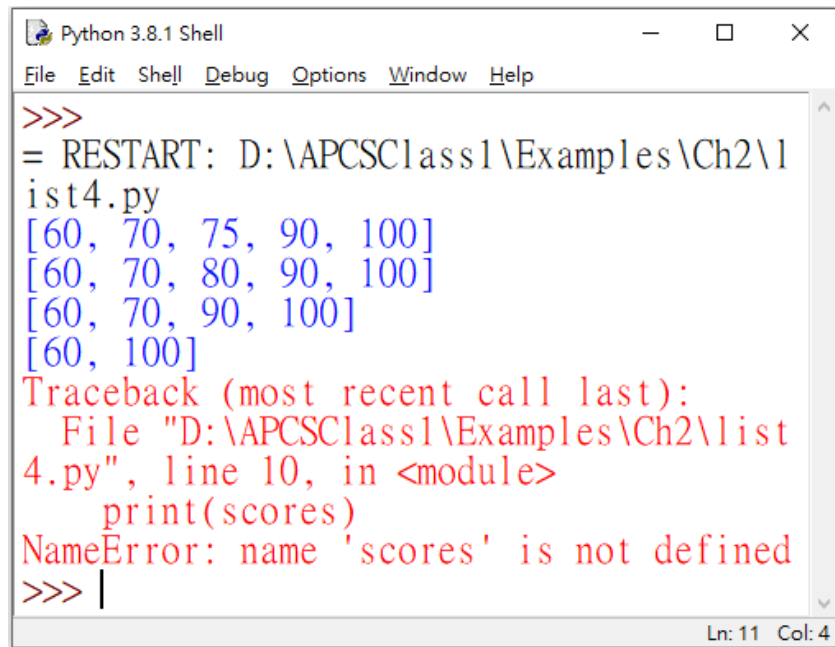
- ◆ del list

# 程式範例



```
list4.py - D:\APCSCClass1\Examples\Ch2\...
File Edit Format Run Options Window Help
scores = [60, 70, 75, 90, 100]
print(scores)
scores[2] = 80
print(scores)
del scores[2]
print(scores)
del scores[1:3]
print(scores)
del scores
print(scores)
```

Ln: 11 Col: 0



```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
>>>
= RESTART: D:\APCSCClass1\Examples\Ch2\l
ist4.py
[60, 70, 75, 90, 100]
[60, 70, 80, 90, 100]
[60, 70, 90, 100]
[60, 100]
Traceback (most recent call last):
  File "D:\APCSCClass1\Examples\Ch2\list
4.py", line 10, in <module>
    print(scores)
NameError: name 'scores' is not defined
>>> |
```

Ln: 11 Col: 4

# Set 集合

## ◆ Set 集合

- ◆ 資料不可重複的資料集合
- ◆ 使用大括號包起來
- ◆ 不可以使用索引取值
- ◆ 使用 in 驗證資料是否存在於Set之中
- ◆ 使用 for – in 將資料逐一取出

```
s = {1, 2, 2, 3, 1}
```

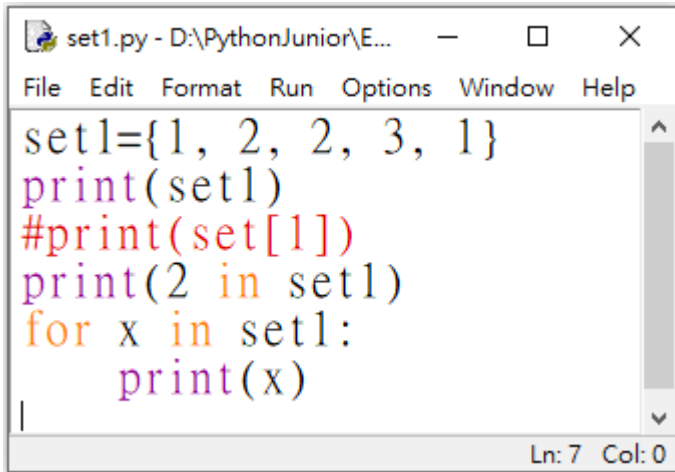
```
print(s)           ⇒ {1, 2, 3}
```

```
print(s[0])
```

```
print(2 in s)      ⇒ True
```

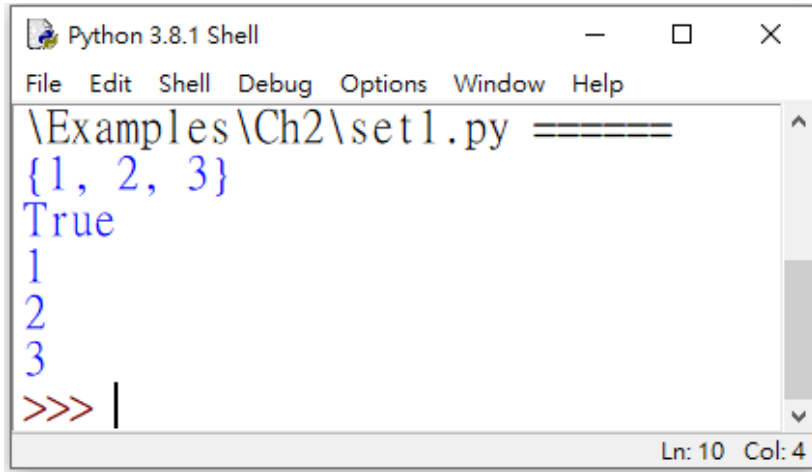
```
for x in s:        ⇒ 1  
                    2  
                    3  
    print(x)
```

# 程式範例



```
set1={1, 2, 2, 3, 1}
print(set1)
#print(set[1])
print(2 in set1)
for x in set1:
    print(x)
```

Ln: 7 Col: 0



```
\Examples\Ch2\set1.py =====
{1, 2, 3}
True
1
2
3
>>> |
```

Ln: 10 Col: 4



# dict 字典

## ◆ dict 字典

### ◇ Python內建鍵值對(Key Value Pair)物件

- 資料集合使用大括號包起來
- 資料元素由 **key** : **value** 組成，多組資料用逗號隔開
- **key** 不能夠重複，重複時造成**value**被取代

### ◇ 字典Dictionary，單字(Key)與翻譯(Value)之間有著對應的關係

- 輸入單字可找到對應的翻譯

# dict 字典基本操作

## ◆ dict 字典基本操作

- ◆ {key:value} 建立字典
- ◆ 使用dict[Key]來取得Value
- ◆ 使用dict[Key]=Value來新增或修改值
- ◆ 使用 del dict[Key] 刪除鍵值對
- ◆ 取值時key不存在拋出 KeyError

```
dict1={'a':100, 'b':200, 'c':300}
```

```
print(dict1['a']) ➡ 100
```

```
dict1['d']=400
```

```
print(dict1['d']) ➡ 400
```

```
dict1['a']=400
```

```
print(dict1['a']) ➡ 400
```

```
del dict1['a']
```

```
print(dict1['a']) ➡ KeyError
```

# 程式範例

```
*dict1.py - D:\APCSClass1\Examples\Ch2\dict1.py (3.8.1)*
File Edit Format Run Options Window Help
scores={'Ch':95,'En':80,'Ma':70,'Ch':100}
print(scores)
scores['Ma']=75
print(scores)
scores['Sc']=75
print(scores)
print(scores['Ch'])
print(scores['En'])
print(scores['Ma'])
print(scores['Sc'])
print(scores['So']) #不存在
|
```

Ln: 12 Col: 0

```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
>>>
= RESTART: D:\APCSClass1\Examples\Ch2\dict1.py
{'Ch': 100, 'En': 80, 'Ma': 70}
{'Ch': 100, 'En': 80, 'Ma': 75}
{'Ch': 100, 'En': 80, 'Ma': 75, 'Sc': 75}
100
80
75
75
Traceback (most recent call last):
  File "D:\APCSClass1\Examples\Ch2\dict1.py",
    line 11, in <module>
      print(scores['So']) #不存在
KeyError: 'So'
>>> |
```

Ln: 14 Col: 4

## Q：下列關於集合型態敘述何者正確？

- a) 串列 **list** 是由相同資料型態的元素所組成的
- b) 串列 **list** 的第一個元素索引值為 1
- c) 元組 **tuple** 的元素值不能改變，但數量可以變更
- d) 使用元組 **tuple** 儲存資料，可以避免疏忽造成資料變更

## Q：下列關於集合型態敘述何者正確？

- a) 集合 set 是有序的資料結構
- b) 集合 set 可以用索引取得集合內容
- c) 集合 set 中每個元素是唯一的
- d) 集合 set 中元素'Devid' 字拼錯了，可以使用集合提供的方法來修正為'David'

## Q：下列關於集合型態敘述何者錯誤？

- a) 字典 dict 的元素適用 key:value 配對方式儲存
- b) 字典 dict 的key 限定為數值或字串
- c) 字典 dict 是無序的資料結構
- d) 串列 list 的元素可以是字典 dict，字典 dict 的值也可以是串列 list

# 練習：身份證字號練習

- ◆ 台灣身份證字號共十碼，包括一個大寫英文字母與九個阿拉伯數字
  - ◇ 首碼英文代表出生登記的戶籍地，如下表
  - ◇ 首數字表示性別，男性為1、女性為2
- ◆ 撰寫一個程式
  - ◇ 輸入身分證字號
  - ◇ 判斷出生地

代碼	出生地	代碼	出生地	代碼	出生地	代碼	出生地
A	臺北市	G	宜蘭縣	M	南投縣	T	屏東縣
B	臺中市	H	桃園市	N	彰化縣	U	花蓮縣
C	基隆市	I	嘉義市	O	新竹市	V	臺東縣
D	臺南市	J	新竹縣	P	雲林縣	W	金門縣
E	高雄市	K	苗栗縣	Q	嘉義縣	X	澎湖縣
F	新北市	L	臺中縣	R	臺南縣	Y	陽明山
				S	高雄縣	Z	連江縣

## 練習：尾牙摸彩

- ◆ 公司舉辦尾牙摸彩，獎品編號如右表
- ◆ 寫一個程式輸入抽到的號碼後顯示獲得的獎品
  - ◇ 摸彩券號碼為1-20號
  - ◇ 11-20號為"銘謝惠顧，明年還有機會"

編號	品名
1	汽車一輛
2	獎金十萬
3	家庭劇院一組
4	筆記型電腦一台
5	iPhone 手機一支
6	Switch 遊樂器一台
7	飯店住宿券一張
8	飯店住宿券一張
9	下午茶券兩張
10	下午茶券兩張



# 練習：資料儲存

- ◆ 寫一個程式接收使用者輸入擁有的汽車廠牌，使用不同的集合儲存資料，滿足下列需求
  - ◆ 取得第N個輸入的汽車廠牌
  - ◆ 輸出所有出現的品牌
  - ◆ 統計下列品牌出現的次數 Audi / Benz / BMW



巨匠直播教學

[www.pcschoolonline.com.tw](http://www.pcschoolonline.com.tw)