

Python Programming

2023 Spring; week 11

Instructor: Cheng-Chun Chang (張正春)
Department of Electrical Engineering

Textbook: Python程式設計:從入門到進階應用(第三版) 2020

課程助教

協助**dubug**, 禁止同學看**code**照抄

第一排:

第二排:

第三排:

第四排:

第五排:

第六排:

打游擊:

Programming: it's all about format ...analogous to 唐詩宋詞



□ 低頭吃便當....

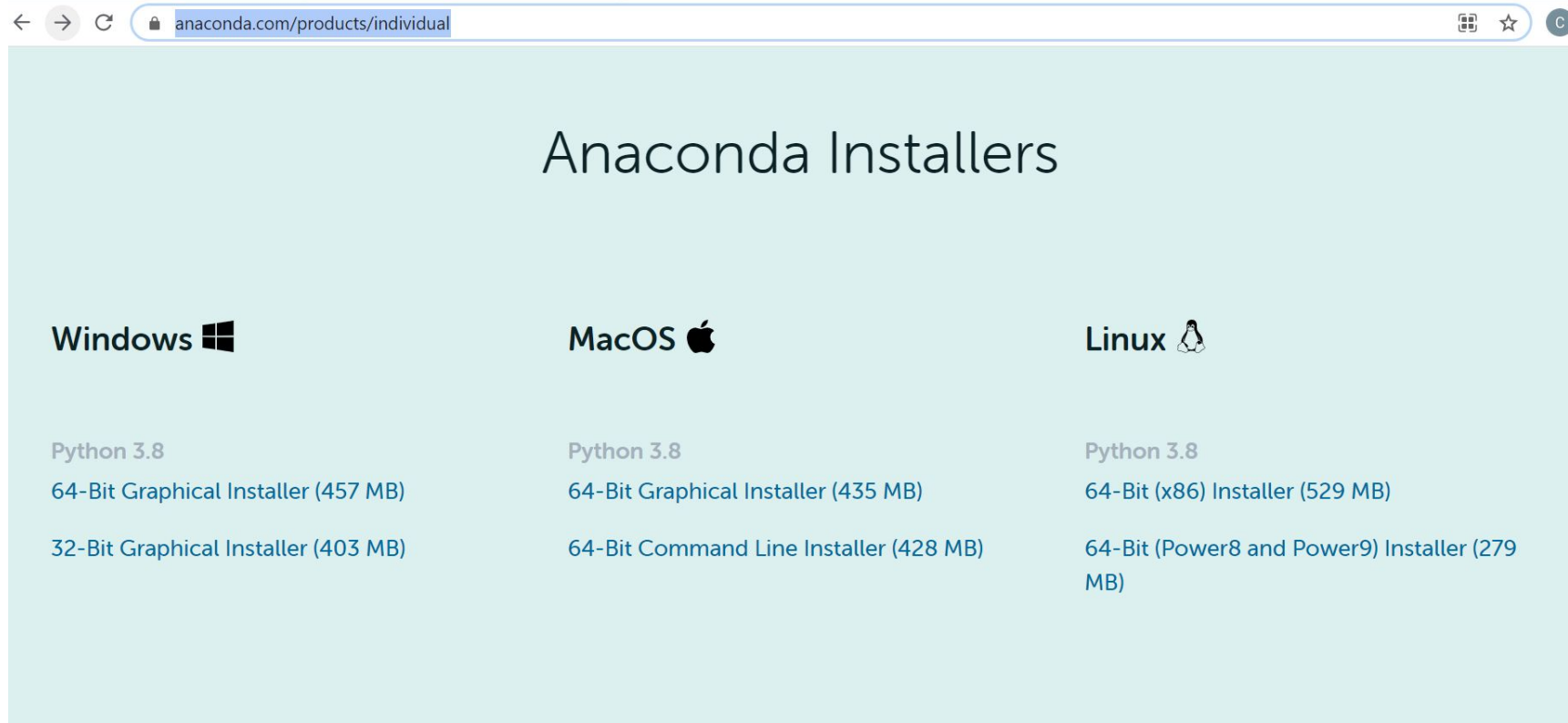


□ 處處蚊子咬....

KEY: 1) 用法 2) 用法 3)用法, and then you can modify it.

目前學校計中提供的IDE

.....註: **IDE** 不是唯一

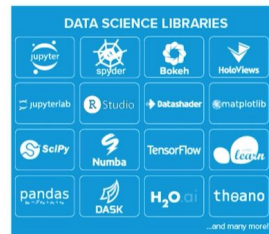


□ <https://www.anaconda.com/products/individual>

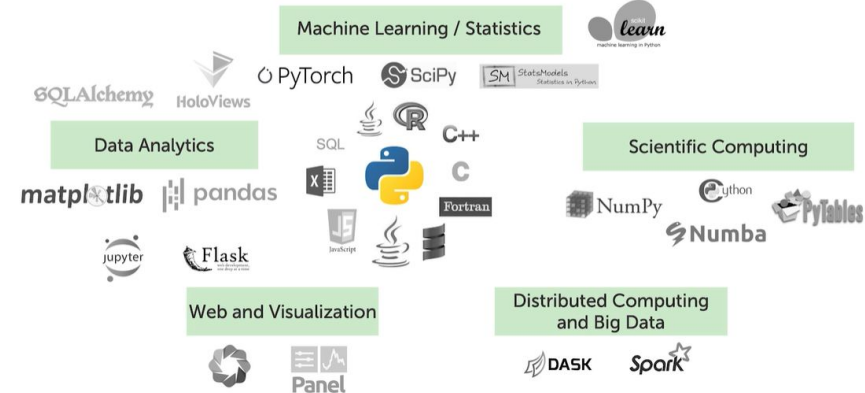
Getting Started with Anaconda Individual Edition

Anaconda Distribution

- Thousands of curated packages
 - Analysis
 - Visualization
 - Modeling
- Mac OS, Linux, and Windows
 - 200+ packages pre-installed
 - It “just works”



Why is Data Science so complicated?



https://anaconda.cloud/tutorials/getting-started-with-anaconda-individual-edition?source=individual_tutorial

Spyder: python IDE



Spyder

4.1.5

Scientific PYTHON Development
EnviRONment. Powerful Python IDE with
advanced editing, interactive testing,
debugging and introspection features

Launch

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our [tutorial](#)

Variable explorer Help Plots Files

Console 1/A

```
In [4]: runfile('C:/Users/cccha/.spyder-py3/temp.py', wdir='C:/Users/cccha/.spyder-py3')
```

your name ?

IPython console History

LSP Python: ready Kite: Installing conda: base (Python 3.8.5) Line 1, Col 30 UTF-8 CRLF RW Mem 63%

Ch1

Python 簡介與程式編輯器介紹

1-1 Python 簡介

- 1989 年的聖誕節, Guido van Rossum在阿姆斯特丹為了消磨假期, 開發新的直譯式語言, 命名為Python
- 2000年10月16日發布Python 2.0
- 2008年12月 日發布Python 3.0
 - 此版不完全相容於Python 2.0
 - Python2.0只到Python 2.7
- 所有新功能都加入到Python 3.0以後的版本

1-1 Python 簡介

- 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

1-1 Python 簡介

- 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常用於字串處理、數學運算、科學計算、系統管理、網頁框架、大數據分析與網頁分析等

1-1 Python 簡介

- 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)

1-2 Python 開發環境

- IDLE為官方的程式開發環境，提供基本功能
 - IDLE安裝與操作步驟請參閱本書P1-3到P1-9
- PyCharm為編輯、除錯與執行的整合開發環境，會自動找出可能的錯誤，社群版本可以免費下載使用
 - PyCharm安裝與操作步驟請參閱本書P1-10到P1-22

-
- Anaconda整合許多軟體與內建許多第3方模組，可以直接使用
 - Anaconda安裝與操作步驟請參閱本書P1-23到P1-31

1-3 Python 的輸入與輸出

- Python3 語言中最常用輸入與輸出的函式為input與print

函式 print	說明	範例與執行結果
print(*objects)	將 objects 顯示在螢幕上，* 表示可以顯示一個以上的 object。若沒有指定間隔字元與結束字元，預設間隔字元 (sep) 使用一個空白鍵，一行的結束字元 (end) 為「\n」表示換到下一行。	<pre>print('學習', 'Python', '真有趣')</pre> <p>說明 預設 sep 使用一個空白鍵，所以每一句話之間都有一個空白鍵。</p> <p>執行結果</p> <pre>學習 Python 真有趣</pre>

函式 print	說明	範例與執行結果
print(*objects, sep=' ', end='\n')	將 objects 顯示在螢幕上，* 表示可以顯示一個以上的 object。預設間隔字元 (sep) 為一個空白鍵，一行的結束字元 (end) 為「\n」表示換到下一行，sep 與 end 都可以依照使用者需求而重新定義。	<pre>print('學習', 'Python', '真有趣', sep='\t')</pre> <p>說明 修改 sep 使用「\t」，「\t」表示 tab 鍵，所以每一句話之間使用一個 tab 鍵，間隔就改成 tab 鍵。</p> <p>執行結果</p> <pre>學習 Python 真有趣</pre>

表 1-2 函式 input 使用方式

函式 input	說明	範例與執行結果
input([prompt])	顯示 prompt 在螢幕上，等待使用者輸入資料，資料會以字串回傳。	<pre>name = input(' 請問貴姓大名? ') print(' 你好, ', name)</pre> <p>說明 因為函式 input 將輸入的資料使用字串回傳，姓名是字串型別所以不用修改，將結果指定給物件 name，使用函式 print 顯示「你好」與物件 name 在螢幕上。物件 name 與運算子「=」的概念將於下一章介紹。</p> <p>執行結果</p> <pre>請問貴姓大名? John 你好, John</pre>

Programming codes I



1-4 第一個Python 程式

▣ 範例1-1 輸入與輸出範例 — 基本資料調查

▣ ch1\1-4- 基本資料調查.py

▣ 問題敘述

- ▣ 寫一個程式，螢幕輸出「請問貴姓大名？」，等待使用者輸入姓名，顯示輸入的姓名在螢幕上。
- ▣ 螢幕輸出「請問年紀？」，等待使用者輸入年紀，顯示輸入的年紀在螢幕上。螢幕輸出「請問體重？」，等待使用者輸入體重，顯示輸入的體重在螢幕上。

ch1\1-4- 基本資料調查.py

□ 解題想法

- 這個程式需要使用input 與print 兩個函式, 函式input 用於輸入資料, 函式print 用於顯示資料到螢幕



WordPad
Document

```
1 name = input(' 請問貴姓大名? ')
2 print(' 你好, ', name)
3 y = int(input(' 請問年紀? '))
4 print(' 原來你 ', y, ' 歲 ')
5 w = float(input(' 請問體重? '))
6 print(' 體重為 ', w)
```

1-4 第一個Python 程式

□ 程式解說

- 第1 行:在螢幕上顯示「請問貴姓大名？」, 等待使用者輸入, 輸入的資料指定給物件name, 物件name 與運算子「=」的概念將於下一章介紹
- 第2 行:輸出「你好, 」與物件name在螢幕上
- 第3 行: 在螢幕上顯示「請問年紀?」, 等待使用者輸入, 輸入的資料經由函式int將字串轉換成整數指定給物件y

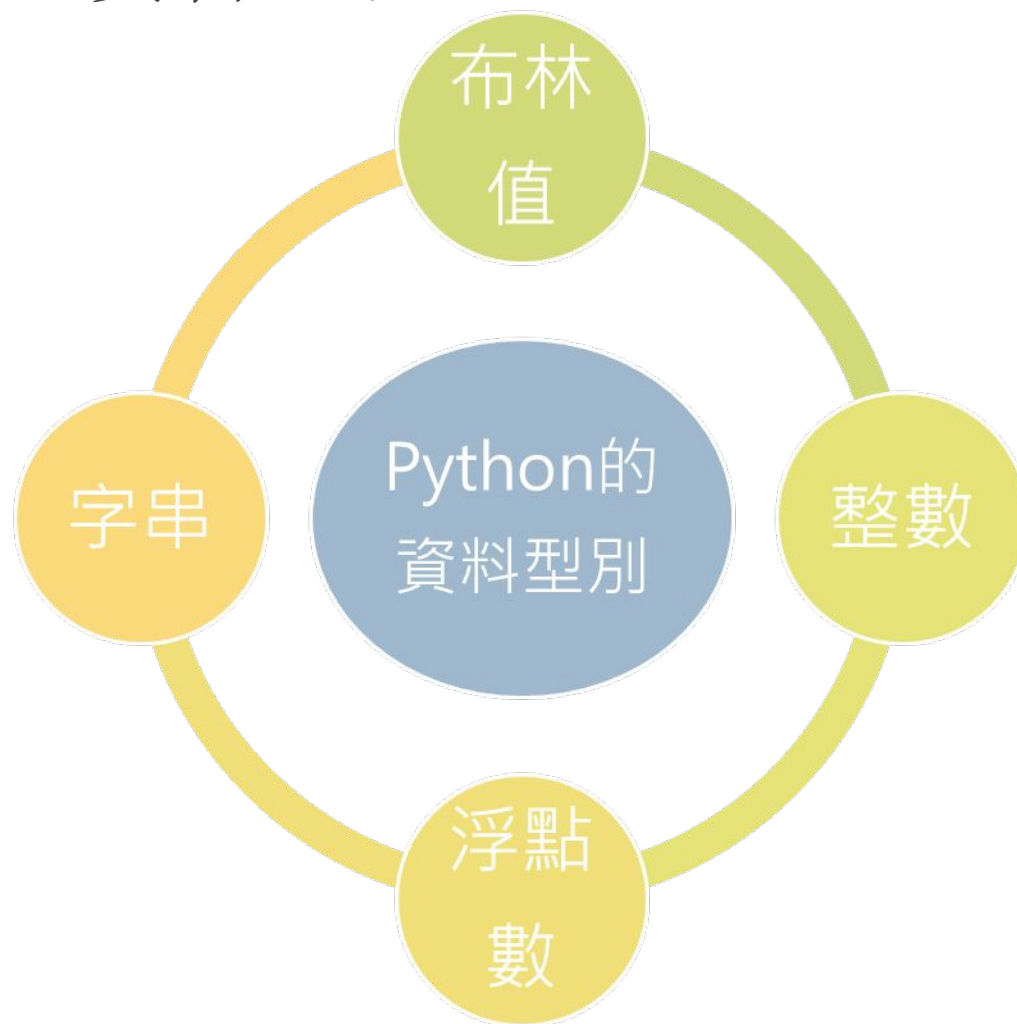
1-4 第一個Python 程式

- 第4 行: 輸出「原來你」、物件y 與「歲」在螢幕上
- 第5 行: 在螢幕上顯示「請問體重?」, 等待使用者輸入, 輸入的資料經由函式float 將字串轉換成浮點數指定給物件w
- 第6 行: 輸出「體重為」與物件w在螢幕上

```
請問貴姓大名?小明  
你好 小明  
請問年紀?55  
原來你 55 歲  
請問體重?99  
體重為 99.0
```

Ch2資料型別、變數與 運算子

2-1 Python 資料型別



布林值


表 2-1 Python 視為 False 的物件

符號	說明
False	布林值 False
0	整數 0
0.0	浮點數 0.0
None	None
()	空 tuple
[]	空串列
{}	空字典
"	空字串
註：tuple、串列與字典將於下一章介紹。	

布林值

表 2-2 函式 bool 的介紹

函式	功能
bool()	根據輸入的資料決定結果是 True 或 False，例如：bool(1) 會輸出 True，bool(0) 會輸出 False。

程式 ( : ch2\2-1-bool1.py)	執行結果
<pre>print(bool(1)) print(bool(0)) print(bool(()))</pre>	<pre>True False False</pre>

整數

表 2-3 函式 int 的介紹

函式	功能
int()	經由函式 int 可以將任何整數、浮點數與整數字串當作輸入，轉換成整數，其中只有整數字串的輸入可以指定整數字串的基底。


程式 ( : ch2\2-1-int1.py)	執行結果
<pre>print(int(100)) print(int(3.14)) print(int('100',2))</pre>	<pre>100 3 4</pre> <p>說明：int('100',2) 表示 100 是二進位表示，轉換成十進位變成 4。</p>

浮點數

- 浮點數表示任何帶有小數點的數值

表 2-4 函式 float 的介紹


函式	功能
float()	可以將任何整數、浮點數與浮點數字串轉換成浮點數。

程式 ( : ch2\2-1-float1.py)	執行結果
<pre>print(float(1)) print(float(3.14)) print(float('3.1415'))</pre>	<pre>1.0 3.14 3.1415</pre>

字串

表 2-5 函式 str 的介紹

函式	功能
str()	將輸入的物件轉成字串。

程式 ( : ch2\2-1-str1.py)	執行結果
<pre>s= str('Python') s[0]='Q'</pre>	<p>s 是字串，所以不能使用 s[0] 來修改字串 s 的第一個元素，會發生 TypeError 錯誤，錯誤訊息如下。</p> <p>Traceback (most recent call last): File "H:/teach/python/ch2-vars-operator/str.py", line 3, in <module> s[0]='Q' TypeError: 'str' object does not support item assignment</p>

Programming codes 2



範例2-1: ch2\2-1-type.py

使用函式 `type`，可以印出變數參考到物件的所屬類別，變數概念將在下一節說明。

```
1 b = True
2 print(b,type(b))
3 num = 20
4 print(num,type(num))
5 pi = 3.14
6 print(pi,type(pi))
7 s = "Hello"
8 print(s,type(s))
```



範例2-1: ch2\2-1-type.py

□ 執行結果

```
True <class 'bool'>
```

```
20 <class 'int'>
```

```
3.14 <class 'float'>
```

```
Hello <class 'str'>
```

2-2 變數

▣ 2-2-1 何謂變數

- ▣ 在Python中任何整數、浮點數、字串、變數與函式都是物件，Python的變數可以想像為標籤，可以使用「=」將標籤貼到物件上，例如：「x=2」，相當於有一個數字2 的物件，將數字2 物件上貼上一個x 的標籤，讀取x 就可取出數字2，也就是變數x 參考到數字2 物件。

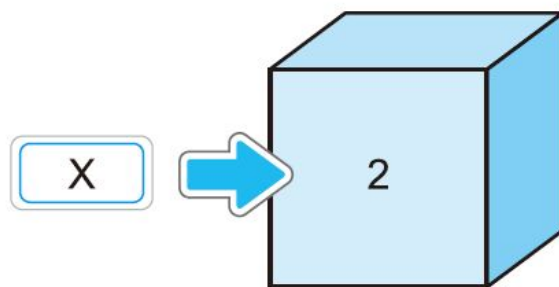


圖 2-1 將數字 2 的物件貼上一個 x 的標籤

2-2 變數

- ▣ 變數是將標籤貼在物件上，就指向那個物件，程式在運算過程中，對資料進行處理與運算，就是對變數進行處理與運算，就是對變數所對應的物件進行處理與運算

2-2 變數

- 程式中命名變數的方式通常有固定的規則，如成績就用 **score** 表示，加總就用 **sum** 等，再對變數進行運算

sum = score1 + score2

加總就用sum表示

成績就用score表示

圖 2-2 變數命名規則示意圖

Programming codes 3



範例2-2-1b:ch2\2-2-1b-var2.py

```
1 a=1
2 print(a, id(a))
3 a='Python'
4 print(a, id(a))
```



WordPad
Document

□ 執行結果

1 500586224

Python 3136064

範例2-2-1c:ch2\2-2-1c-var3.py

```
1 x = 1
2 y = x
3 print(id(x),id(y))
4 print(x,y)
```



WordPad
Document

□ 執行結果

```
493311728 493311728
```

```
1 1
```

2-2-2 變數的命名

▣ 變數的命名有一定的規則

- ▣ 1.變數的第一個字母一定只能是英文大小寫字母、Unicode字元或底線(_), 其後可以接英文大小寫字母、Unicode字元、底線(_) 或數字, 也就是不能以數字開頭, Unicode字元表示可以使用中文命名變數

表 2-6 變數的命名範例

正確	不正確	不正確原因
SCORE_1	1_SCORE	無法使用數字開頭
成績	成績?	包含英文半形問號「?」, 英文半形問號「?」不是英文字母

2-2-2 變數的命名

- 2. 小寫英文字母視為不同變數
 - A 與a 視為不同的變數
- 3. Python關鍵字無法命名為變數名稱
 - 例如:if、else、elif、for 等, 不能使用這些名稱命名變數
- 4. 變數名稱可以利用多個有意義的小寫單字組合而成, 單字之間使用底線() 串接, 程式設計者較容易閱讀與瞭解
 - 如表示數學成績的變數可以使用math_score來表示, 這個規定並沒有強制性

2-3 運算子

- 將數值或變數進行運算，需要使用運算子
- 運算子分成

指定運算子

算數運算子

字串運算子

比較運算子

邏輯運算子

2-3-1 指定運算子

- 用等號(=) 表示, 意思是等號右邊先運算, 再將運算結果指定給左邊的變數

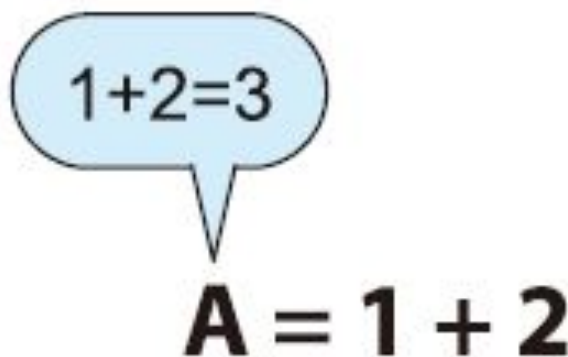


圖 2-3 指定運算子示意圖

2-3-2 算術運算子

表 2-7 算術運算子

運算子	說明	舉例	結果
+	加	$A=5+2$	$A=7$
-	減	$A=5-2$	$A=3$
*	乘	$A=5*2$	$A=10$
/	浮點除法	$A=5/2$	$A=2.5$
//	整數除法 (去除小數點)	$A=5//2$	$A=2$
%	相除後求餘數	$A=5\%2$	$A=1$
**	次方	$A=5**2$	$A=25$

2-3-2 算術運算子

表 2-8 程式中數學運算子範例

運算式	結果
$a = (2 + 3 * 2) * (4 - 1)$	變數 a 的值為 24，因為左邊括弧內 $3 * 2$ 先運算，結果為 6，再加上 2 得到 8，右邊括弧內 $4 - 1$ ，運算結果為 3，最後 8 乘以 3 得 24，獲得最後結果。

2-3-2 算術運算子

表 2-9 變數遞增或遞減範例

分類	範例	範例說明
遞增 $a = a + 1$	$a = 5$ $a = a + 1$	變數 a 的值為 6。 首先變數 a 參考到數字 5，接著執行「 $a = a + 1$ 」，等號「 $=$ 」右邊先計算，先執行「 $a + 1$ 」，結果為「6」，變數 a 經由等號「 $=$ 」參考到數字 6，變數 a 的數值就會變成 6，如此達成變數 a 遞增的效果。
遞減 $a = a - 1$	$a = 5$ $a = a - 1$	變數 a 的值為 4。 首先變數 a 參考到數字 5，接著執行「 $a = a - 1$ 」，等號「 $=$ 」右邊先計算，先執行「 $a - 1$ 」，結果為「4」，變數 a 經由等號「 $=$ 」參考到數字 4，變數 a 的數值就會變成 4，如此達成變數 a 遞減的效果。

2-3-2 算術運算子

表 2-10 算術運算子範例

運算子	說明	範例	將範例進行縮寫
+=	加	A=A+2	A += 2
-=	減	A=A-2	A -= 2
*=	乘	A=A*2	A *= 2
/=	浮點除法	A=A/2	A /= 2
//=	整數除法 (去除小數點)	A=A//2	A //= 2
%=	相除後求餘數	A=A%2	A %= 2
=	次方	A=A2	A **= 2

2-3-3 比較運算子

表 2-11 比較運算子

比較運算子	說明	舉例
<	判斷是否小於	A=(5<2) 結果：A=False(False 表示條件不成立，結果為假)
<=	判斷是否小於等於	A=(5<=2) 結果：A=False(False 表示條件不成立，結果為假)
>	判斷是否大於	A=(5>2) 結果：A=True(True 表示條件成立，結果為真)
>=	判斷是否大於等於	A=(5>=2) 結果：A=True(True 表示條件成立，結果為真)
!=	判斷是否不等於	A=(5!=2) 結果：A=True(True 表示條件成立，結果為真)
==	判斷是否等於	A=(5==2) 結果：A=False(False 表示條件不成立，結果為假)

2-3-4 邏輯運算子

X and Y	Y=True	Y=False
X=True	True	False
X=False	False	False

2-3-4 邏輯運算子

X or Y	Y=True	Y=False
X=True	True	True
X=False	True	False

2-3-4 邏輯運算子

	not X
X=True	False
X=False	True

充電時間

舉例	X 值	結果	說明
$((X > 60) \text{ and } (X < 80))$	70	True	條件 $(70 > 60)$ 為 True，而條件 $(70 < 80)$ 為 True，經由 and(且) 運算結果為 True
$((X > 60) \text{ and } (X < 80))$	60	False	條件 $(60 > 60)$ 為 False，只要有一個條件 False，經由 and(且) 運算結果就為 False
$((X > 60) \text{ or } (X < 80))$	60	True	條件 $(60 < 80)$ 為 True，經由 or(或) 運算只要有一個條件為 True，結果就為 True。
$\text{not}(X > 60)$	60	True	條件 $(60 > 60)$ 為 False，取 not(非) 運算，結果變成 True。

2-3-5 in 與 is 運算子

表 2-12 in 與 is 運算子

運算子	說明	舉例	結果
in	是否包含	<pre>x = 1 y = [1, 2, 3] print(x in y)</pre>	True 註：[1, 2, 3] 為串列，下一章會介紹。
not in	是否不包含	<pre>x = 1 y = [1, 2, 3] print(x not in y)</pre>	False
is	是否為相同物件	<pre>x = [1, 2, 3] y = [1, 2, 3] print(x is y)</pre>	False
is not	是否不為相同物件	<pre>x = [1, 2, 3] y = [1, 2, 3] print(x is not y)</pre>	True

2-3-6 位元運算子

表 2-13 位元運算子

運算子	說明	舉例	結果
&	位元且運算，當兩個都 True，才將該位元設定 True，其他都設定為 False。	A=5&4	5 的二進位為 0101 4 的二進位為 0100 0101&0100=0100 A=4
	位元或運算，當兩個都 False，才將該位元設定 False，其他都設定為 True。	A=5 4	5 的二進位為 0101 4 的二進位為 0100 0101 0100=0101 A=5
^	位元互斥或運算，當一個為 True，另一個為 False，才將該位元設定為 True，其他都設定為 False。	A=5^4	5 的二進位為 0101 4 的二進位為 0100 0101^0100=0001 A=1

2-3-6 位元運算子

運算子	說明	舉例	結果
~	位元取相反	$A = \sim 5$	5 的二進位為 0101 $\sim(0101) = 1010$ ，開頭是 1 表示為負數，所以獲得「 $A = -6$ 」
<<	位元左移運算，左移指定的位元個數。	$A = 5 << 2$	5 的二進位為 0101 $101 << 2 = 10100$ $A = 20$
>>	位元右移運算，右移指定的位元個數。	$A = 5 >> 2$	5 的二進位為 0101 $0101 >> 2 = 0001$ $A = 1$

2-3-7 運算子優先權次序

- 運算子的運算先後順序是有規則的

表 2-14 運算子優先權次序

優先權	運算子	說明
<div>高</div> <div>↓</div> <div>低</div>	<div>()</div> <div>[]</div> <div>{ 鍵 : 值 }</div> <div>{ }</div>	<div>建立 tuple 運算子</div> <div>建立串列運算子</div> <div>建立字典運算子</div> <div>建立集合運算子</div> <div>這些資料容器將於下一章會進行介紹。</div>
	**	次方
	+X 、 -X 、 ~X	取正號、取負號與取位元相反
	* 、 / 、 // 、 %	乘法、浮點除法、整數除法、求餘數

優先權	運算子	說明
高  低	+、-	加法、減法
	<<、>>	位元左移與位元右移
	&	位元且運算
	^	位元互斥或運算
		位元或運算
	<	判斷是否小於
	<=	判斷是否小於等於
	>	判斷是否大於
	>=	判斷是否大於等於
	==	判斷是否相等
	!=	判斷是否不相等
	in	是否包含
	not in	是否不包含
	is	是否是
	is not	是否不是

2-3-7 運算子優先權次序

優先權	運算子	說明
<div>高</div> <div>低</div>	not	邏輯運算子的非
	and	邏輯運算子的且
	or	邏輯運算子的或
	if elif else	條件判斷
	lambda	匿名函式

2-3-7 運算子優先權次序

範例一	乘除先運算	接著加減運算
$F=2+3*5-14/7$	$F=2+15-2$	$F=15$

範例二	括號先運算	接著求餘數運算
$F=(2+3)\%4$	$F=5\%4$	$F=1$

2-4 字串

- 使用單引號「'」與雙引號「"」所包夾的文字，在Python 會被視為字串
 - 字串內文字可以儲存Unicode 編碼的文字，支援中文
- 單引號內使用雙引號，可以正確顯示雙引號，雙引號內也可以使用單引號，也可以正確顯示單引

範例2-4:ch2\2-4-string1.py

程式碼	執行結果
<pre>s1 = '作者 " 孟浩然 " 詩名 " 春曉 " print(s1) s2 = " 作者 ' 孟浩然 ' 詩名 ' 春曉 '"print(s2)</pre>	<pre>作者 " 孟浩然 " 詩名 " 春曉 " 作者 ' 孟浩然 ' 詩名 ' 春曉 '</pre>

使用三個單引號「'''」或三個雙引號（"""）可以用於顯示多行文字，且每行前面的空白也會正常顯示，連換行字元也會被保留。

程式碼	執行結果
<pre>s3 = ''' 春眠不覺曉，處處聞啼鳥。 夜來風雨聲，花落知多少。 作者 " 孟浩然 " 詩名 " 春曉 " ''' print(s3)</pre>	<pre>春眠不覺曉，處處聞啼鳥。 夜來風雨聲，花落知多少。 作者 " 孟浩然 " 詩名 " 春曉 "</pre>

Readings



2-4-1 字串運算子

- 字串運算子用於處理字串，可以
 - 串接字串
 - 存取字串
 - 複製字串
 - 切割字串

範例2-4-1:ch2\2-4-1-string2.py

□ 1. 串接字串

- 使用「+」串接字串, 可以將兩個字串合併成一個字串

程式碼	執行結果
<pre>s1 = '01234' s2 = '56789' s3 = s1 + s2 print(s3)</pre>	<pre>0123456789</pre>

範例2-4-1:ch2\2-4-1-string2.py

□ 2. 複製字串

- 使用「*」複製字串，執行「字串 * 2」會產生一個新字串，該字串複製字串一份串接原來字串的後面，執行「字串 * 3」會產生一個新字串，該字串複製字串兩份串接原來字串的後面，依此類推。

程式碼	執行結果
<pre>s1 = '01234' s2 = s1 * 2 print(s2)</pre>	0123401234

範例2-4-1:ch2\2-4-1-string2.py

□ 3. 取出字串元素

- 使用「[]」取出字串元素, s[0] 取出字串s 的第1 個元素, s[1] 取出字串s 的第2個元素, s[-1] 取出字串s 的最後1 個元素, s[-2] 取出字串s 的右邊數過來第2 個元素

程式碼	執行結果
<pre>s1 = '0123456789' print(s1[0]) print(s1[1]) print(s1[-1]) print(s1[-2])</pre>	<pre>0 1 9 8</pre>

□ 4. 切割字串

- 使用「[開始: 結束: 間隔]」切割字串, 從「開始」到「結束」(不包含結束的字元) 每隔「間隔」個字元取一個字元出來

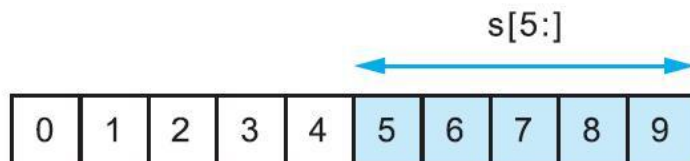
程式碼	執行結果
<pre> s = '0123456789' print('s=', s, 's[:]=' , s[:]) print('s=', s, 's[5:]=' , s[5:]) print('s=', s, 's[-2:]=' , s[-2:]) print('s=', s, 's[:5]=' , s[:5]) print('s=', s, 's[:-2]=' , s[:-2]) print('s=', s, 's[7:9]=' , s[7:9]) print('s=', s, 's[-4:-1]=' , s[-4:-1]) print('s=', s, 's[5:-2]=' , s[5:-2]) print('s=', s, 's[2:10:2]=' , s[2:10:2]) print('s=', s, 's[::-1]=' , s[::-1]) print('s=', s, 's[-1::-1]=' , s[-1::-1]) </pre>	<pre> s= 0123456789 s[:]= 0123456789 s= 0123456789 s[5:] = 56789 s= 0123456789 s[-2:] = 89 s= 0123456789 s[:5] = 01234 s= 0123456789 s[:-2] = 01234567 s= 0123456789 s[7:9] = 78 s= 0123456789 s[-4:-1] = 678 s= 0123456789 s[5:-2] = 567 s= 0123456789 s[2:10:2] = 2468 s= 0123456789 s[::-1] = 9876543210 s= 0123456789 s[-1::-1] = 9876543210 </pre>

程式解說

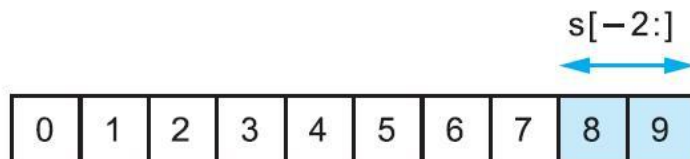
- `s[:]` 表示取字串 `s` 的每一個元素，如下圖，輸出「0123456789」到螢幕上。



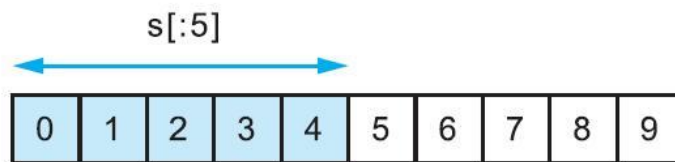
- `s[5:]` 表示取從字串 `s[5]` 元素到字串 `s` 結束的所有元素，如下圖，輸出「56789」到螢幕上。



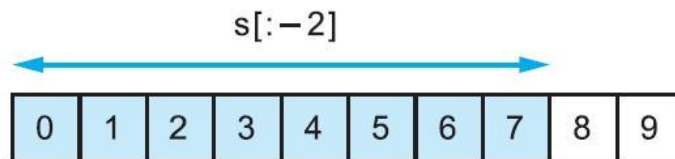
- `s[-2:]` 表示取從字串 `s[-2]` 元素 (倒數第 2 個元素) 到字串 `s` 結束的所有元素，如下圖，輸出「89」到螢幕上。



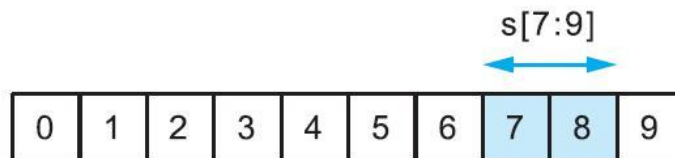
- `s[:5]` 表示取從字串 `s[0]` 元素到 `s[5]` 所指定元素的前 1 個元素為止的所有元素，如下圖，輸出「01234」到螢幕上。



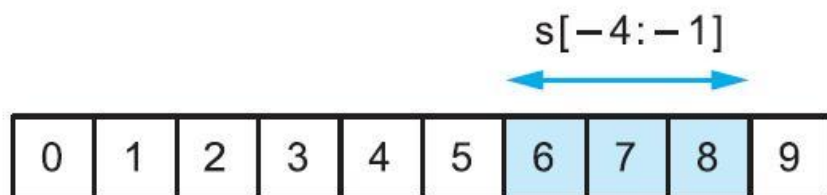
- `s[:-2]` 表示取從字串 `s[0]` 元素到 `s[-2]` (倒數第 2 個元素) 所指定元素的前 1 個元素為止的所有元素，如下圖，輸出「01234567」到螢幕上。



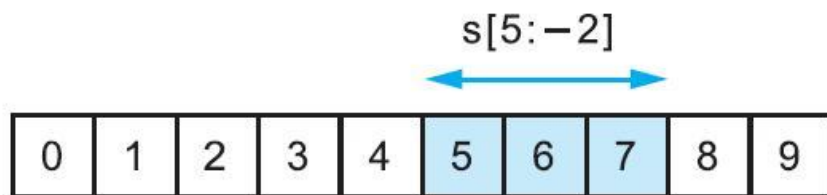
- `s[7:9]` 表示取從字串 `s[7]` 元素到 `s[9]` 所指定元素的前 1 個元素為止的所有元素，如下圖，輸出「78」到螢幕上。



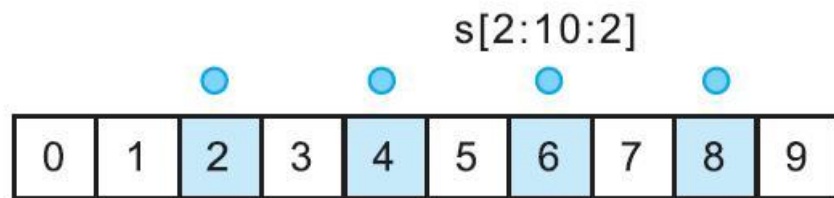
- `s[-4:-1]` 表示取從字串 `s[-4]` (倒數第 4 個元素) 元素到 `s[-1]` (倒數第 1 個元素) 所指定元素的前 1 個元素為止的所有元素，如下圖，輸出「678」到螢幕上。



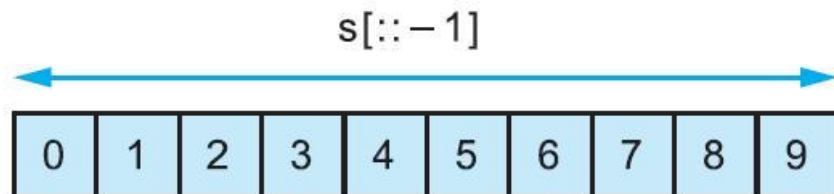
- `s[5:-2]` 表示取從字串 `s[5]` 元素到 `s[-2]` (倒數第二個元素) 所指定元素的前 1 個元素為止的所有元素，如下圖，輸出「567」到螢幕上。



- `s[2:10:2]` 表示取字串 `s[2]` 元素到 `s[10]` 所指定元素的前 1 個元素為止的所有元素中，每隔 2 個元素取一個元素，如下圖，輸出「2468」到螢幕上。



- `s[::-1]` 與 `s[-1::-1]` 都表示反轉字串，如下圖，輸出「9876543210」到螢幕上。



範例2-4-1:ch2\2-4-1-string2.py

□ 5. 串接多行

- 使用「\」串接多行, 若在python 中同一行的程式碼過長, 在該行最後使用「\」當最後一個字元, 就可以寫到下一行, 這兩行會被視為同一行

程式碼	執行結果
<pre>s = '春眠不覺曉，處處聞啼鳥。\\n\\n夜來風雨聲，花落知多少。\\n\\n\\t作者 "孟浩然" 詩名 "春曉" \\n\\nprint(s)</pre>	<pre>春眠不覺曉，處處聞啼鳥。\\n\\n夜來風雨聲，花落知多少。\\n\\n\\t作者 "孟浩然" 詩名 "春曉" \\n\\n</pre>

2-4-2 字串的內建函式

- ▣ 範例2-4-2: ch2\2-4-2-string3.py
- ▣ 字串.split(切割字元)

● 範例

```
s1='春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
list1=s1.split('，')
```

● 執行結果

```
['春眠不覺曉','處處聞啼鳥','夜來風雨聲','花落知多少。']
```


範例2-4-2:ch2\2-4-2-string3.py

□ 連結字串.join(要串接的串列)

● 範例

```
list1=['春眠不覺曉','處處聞啼鳥','夜來風雨聲','花落知多少。']  
s2=', '.join(list1)  
print(s2)
```

● 執行結果

春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.replace(原始字串, 取代字串)

● 範例

```
s1=' 春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
s3=s1.replace(' 春 ',' 冬 ')  
print(s3)
```

● 執行結果

```
冬眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。
```

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.find(要找的字串)

● 範例

```
s1='春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
print(s1.find('花落 '))
```

● 執行結果

18

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.rfind(要找的字串)

● 範例

```
s1='春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
s1.rfind('處')
```

● 執行結果

7

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.startswith(要找的字串)

● 範例

```
s1='春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
print(s1.startswith('春眠 '))
```

● 執行結果

```
True
```

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.endswith(要找的字串)

● 範例

```
s1='春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
print(s1.endswith('多少。'))
```

● 執行結果

```
True
```

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.count(要找的字串)

● 範例

```
s1='春眠不覺曉，處處聞啼鳥，夜來風雨聲，花落知多少。'  
print(s1.count('處'))
```

● 執行結果

2

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.center(數值)

● 範例

```
s1='春眠不覺曉 '  
print(s1.center(10))
```

● 執行結果

春眠不覺曉

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.rjust(數值)

● 範例

```
s1='春眠不覺曉 '  
print(s1.rjust(10))
```

● 執行結果

```
春眠不覺曉
```

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.ljust(數值)

● 範例

```
s1='春眠不覺曉 '  
print(s1.ljust(10))
```

● 執行結果

```
春眠不覺曉
```

範例2-4-2:ch2\2-4-2-string3.py

□ 英文.capitalize()

● 範例

```
s1='an apple a day'  
print(s1.capitalize())
```

● 執行結果

```
An apple a day.
```

範例2-4-2:ch2\2-4-2-string3.py

□ 英文.title()

● 範例

```
s1='An apple a day'  
print(s1.title())
```

● 執行結果

```
An Apple A Day.
```

範例2-4-2:ch2\2-4-2-string3.py

□ 英文.swapcase()

● 範例

```
s1='An apple a day'  
print(s1.swapcase())
```

● 執行結果

```
aN APPLE A DAY.
```

範例2-4-2:ch2\2-4-2-string3.py

□ 英文.upper()

● 範例

```
s1='An apple a day.'  
print(s1.upper())
```

● 執行結果

```
AN APPLE A DAY.
```

範例2-4-2:ch2\2-4-2-string3.py

□ 英文.lower()

● 範例

```
s1='An apple a day.'  
print(s1.lower())
```

● 執行結果

```
an apple a day.
```

範例2-4-2:ch2\2-4-2-string3.py

□ 字串.zfill(width)

● 範例

```
s1='123'  
print(s1.zfill(5))
```

● 執行結果

```
00123
```


範例2-4-2:ch2\2-4-2-string3.py

□ 字串.strip(chars)

● 範例

```
s1=' Hello,Mary. '  
print(s1.strip())
```

● 執行結果

```
Hello,Mary.
```

【執行結果說明】移除左右兩邊的空白字元。

範例2-4-2:ch2\2-4-2-string3.py

▣ 字串.lstrip(chars)

● 範例

```
s1=' Hello,Mary. '  
print(s1.lstrip(' H'))
```

● 執行結果

```
ello,Mary.
```

【執行結果說明】 移除左邊的空白字元與字元「H」。

範例2-4-2:ch2\2-4-2-string3.py

▣ 字串.rstrip(chars)

● 範例

```
s1=' Hello,Mary. '  
print(s1.rstrip(' .'))
```

● 執行結果

```
Hello,Mary
```

【執行結果說明】 移除右邊的空白字元與字元「.」。

□ Exercise



2-5 範例練習

- ▣ **範例2-5-1 服裝訂購系統**: ch2\ex- 服裝訂購系統.py
- ▣ 假設上衣300元、褲子350元與背心400元, 使用者可以自行輸入三種服裝的數量, 請設計一個程式計算訂購服裝的總金額

□ 解題想法

- 將上衣、褲子與背心訂購數量依序指定到三個整數變數中，乘以對應的價格，再加總起來。本題會使用到運算子的乘法(*)、加法(+) 與指定運算子(=)。

範例2-5-1 服裝訂購系統:ch2\ex- 服裝訂購系統.py

1
2
3
4
5

Exercise 1 (2 pt)



WordPad
Document

□ 執行結果

依序輸入上衣 2 件、褲子 3 件與背心 1 件，結果顯示在螢幕。

請輸入上衣數量？ 2

請輸入褲子數量？ 3

請輸入背心數量？ 1

訂購服裝的總金額為 2050

範例2-5-4 複利計算:ch2\2-5-4- 複利計算.cpp

- 寫一個程式協助使用者計算定存一筆錢，依照所輸入的利率，定存一年到三年的本金與利息和，使用複利方式計算
- 解題想法
 - 將本金與利率指定到兩個變數，再依照複利公式計算前三年的本利和，將計算結果分別儲存到三個浮點數變數。本題會使用到運算子的加法(+)、乘法(*)、除法(/)、次方(**) 與指定運算子(=)。

$$\text{複利計算公式： 本金} \times (1 + \text{利率})^{(\text{總期數})} = \text{本利和 (終值)}$$

範例2-5-4 複利計算:ch2\2-5-4- 複利計算.cp

□ 程式碼

Exercise 2 (2 pt)



WordPad
Document

範例2-5-4 複利計算:ch2\2-5-4- 複利計算.cp

□ 執行結果

本金輸入「10000」，利率輸入「1.5」，計算結果顯示在螢幕。

請輸入本金？ 10000

請輸入年利率 (%)？ 1.5

第一年本利和為 10149.999999999998

第二年本利和為 10302.249999999996

第三年本利和為 10456.783749999997

To be continued.....

Instructor: Cheng-Chun Chang (張正春)
Department of Electrical Engineering