

程式設計 (二)

Python CH1. Hello World

Ming-Hung Wang 王銘宏

tonymhwang@cs.ccu.edu.tw

Department of Computer Science and Information Engineering
National Chung Cheng University

Spring Semester, 2022

本章目錄

1. Python 介紹
2. Anaconda+VSCode
3. 變數與基本型態
4. 運算子
5. 輸入輸出
6. 流程控制
7. 基本 Python 風格

Python 介紹



One of the world's most popular
programming languages

<https://www.python.org/>

TIOBE Index for February 2022







February Headline: TIOBE index top 3 benefits from technology changes

As of the 1st of May, the Alexa web traffic ranking engine is going to stop its services. Alexa was used to select the search engines for the TIOBE index until now. So now something has to change. Similarweb has been chosen as the alternative for Alexa. We have used Similarweb for the first time this month to select search engines and fortunately, there are no big changes in the index due to this swap. The only striking difference is that the top 3 languages, Python, C, and Java, all gained more than 1 percent in the rankings. We are still fine-tuning the integration with Similarweb, which is combined with a shift to HtmlUnit in the back-end. Some websites are not onboarded yet, but will follow soon. Now that HtmlUnit is applied for web crawling, it will become possible to add more sites to the index, such as Stackoverflow and Github. This will hopefully happen in the next few months. --Paul Jansen
CEO TIOBE Software

The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. Popular search engines such as Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings. It is important to note that the TIOBE index is not about the *best* programming language or the language in which *most lines of code* have been written.

The index can be used to check whether your programming skills are still up to date or to make a strategic decision about what programming language should be adopted when starting to build a new software system. The definition of the TIOBE index can be found [here](https://www.tiobe.com/tiobe-index/).

Feb 2022	Feb 2021	Change	Programming Language	Ratings	Change
1	3	▲	 Python	15.33%	+4.47%
2	1	▼	 C	14.08%	-2.26%
3	2	▼	 Java	12.13%	+0.84%
4	4		 C++	8.01%	+1.13%
5	5		 C#	5.37%	+0.93%

<https://www.tiobe.com/tiobe-index/>

Python 格言

The Zen of Python ~ by Tim Peters	Python之禪 ~ 提姆彼特斯
Beautiful is better than ugly.	優美優於醜陋，
Explicit is better than implicit.	明瞭優於隱晦；
Simple is better than complex.	簡單優於複雜，
Complex is better than complicated.	複雜優於凌亂，
Flat is better than nested.	扁平優於嵌套，
Sparse is better than dense.	稀疏優於稠密，
Readability counts.	可讀性很重要！
Special cases aren't special enough to break the rules.	即使實用比純粹更優，特例亦不可違背原則。
Although practicality beats purity.	錯誤絕不能悄悄忽略，
Errors should never pass silently.	除非它明確需要如此。
Unless explicitly silenced.	面對不確定性，拒絕妄加猜測。
In the face of ambiguity, refuse the temptation to guess.	任何問題應有一種，且最好只有一種，
There should be one-- and preferably only one --obvious way to do it.	顯而易見的解決方法。
Although that way may not be obvious at first unless you're Dutch.	儘管這方法一開始並非如此直觀，除非你是荷蘭人。
Now is better than never.	做優於不做，
Although never is often better than *right* now.	然而不假思索還不如不做。
If the implementation is hard to explain, it's a bad idea.	很難解釋的，必然是壞方法。
If the implementation is easy to explain, it may be a good idea.	很好解釋的，可能是好方法。
Namespaces are one honking great idea -- let's do more of those!	命名空間是個絕妙的主意，我們應好好利用它。

為什麼學 Python?

- 語法簡單、撰寫方便
- 套件很多，幾乎什麼都能做到
- 用戶多，網路上很多參考資料

為什麼學 C?

- 寫出來的程式效率很高 (很快)
- 跟硬體相關的幾乎都要使用到 C
- Python 的底層是用 C 寫的
- 學 C 對建構電腦科學的觀念很有幫助

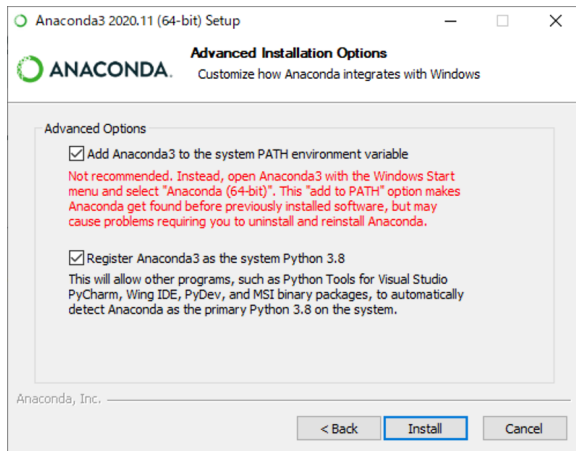
Anaconda+VSCode

Anaconda

Anaconda 致力於簡化軟體套件管理系統和部署，
是一個免費開源的 Python 和 R 語言的發行版本，
可用於計算科學。

下載官網：

<https://www.anaconda.com/products/individual#Downloads>



安裝時 PATH 選項記得打勾



- Spyder: 一個使用 Python 語言的 IDE
- Jupyter Notebook: 一個基於 Web 的互動式計算環境

VSCode

由微軟開發，支援 git 和許多組件的強大記事本。

下載官網：

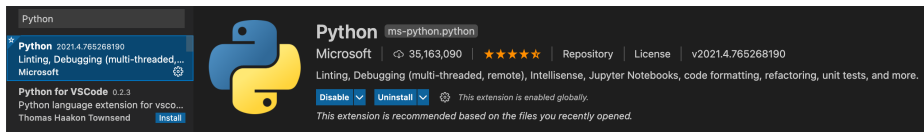
<https://code.visualstudio.com/>

VSCode 的 Python 組件

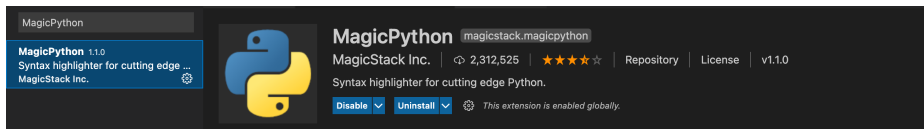
組件從左側這個按鈕安裝



Python：運行 Python 使用

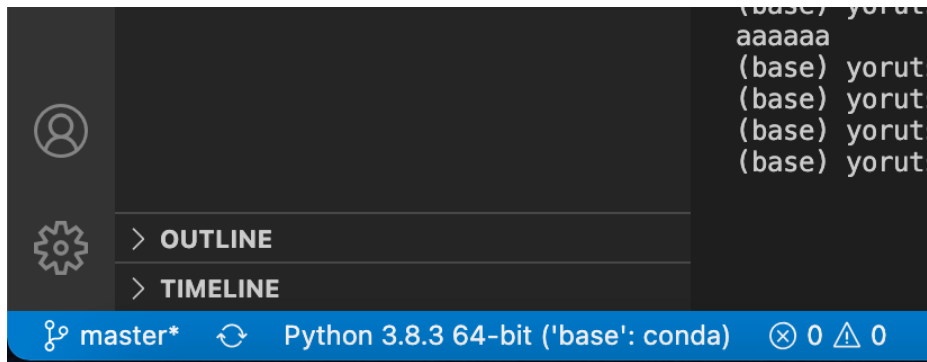


MagicPython：Python 語法 highlight



Anaconda+VSCode

若開啟.py 檔，左下角可選擇運行的環境，
請選擇 anaconda 目錄下的 python 版本



變數與基本型態

基本型態

- 數字
 - 整數：0、-11、+33、123456、...
 - 值域：位數無上限
 - 浮點數：3.14、-2e-8、...
 - 精度：等於 C 的 double
 - 布林值：True、False
 - 複數：(3+4j)、(-2-1j)、(-1+0j)、...
- 字串：'APPLE'、'BANANA'、...
- None

變數宣告

不要註明型態。指派變數時，若該變數第一次出現，則被當作宣告。

```
1 i = 48763      #宣告整數
2 f = 3.14       #宣告浮點數
3 b = True       #宣告布林值
4 s = 'APPLE'    #宣告字串，可用單引號或雙引號
5 ss = '''APPLE
6 AND
7 BANANA'''      #宣告多行字串，可用3個單引號或雙引號
```

若想宣告變數但還不想給值，可以：

```
1 i = int()      #宣告整數，預設0
2 f = float()    #宣告浮點數，預設0.0
3 b = bool()     #宣告布林值，預設False
4 s = str()      #宣告字串，預設''
```

輸出-print

```
1 i = 48763    #宣告整數
2 f = 3.14     #宣告浮點數
3 b = True     #宣告布林值
4 s = 'APPLE'  #宣告字串，可用單引號或雙引號
5 ss = '''APPLE
6 AND
7 BANANA'''   #宣告多行字串，可用3個單引號或雙引號
8 print(i)     #印出 i，並換行
9 print(i, f, b, s) #印出 i, f, b, s，間隔為' '，並換行
10 print(ss)   #印出 ss
```

```
48763
48763 3.14 True APPLE
APPLE
AND
BANANA
```

弱型別語言

變數的型態可任意更改

```
1 a = 48763    #宣告整數a
2 print(a, type(a))
3 a = '喵喵'   #將a改為字串
4 print(a, type(a))
```

```
48763 <class 'int'>
喵喵 <class 'str'>
```

None

除了字串與數字等標準型別之外，python 還有一個特殊的標準型別 None，用以表示空值。
None 不等於除了 None 以外的任何數值。

```
1 print(None == 0)
2 print(None == False)
3 print(None == None)
4 print(0 == False)
```

```
False
False
True
True
```

註解

Python 的單行註解使用 #
多行註解使用 3 個單引號''' 或 3 個雙引號"""

```
#我是單行註解
```

```
'''
```

```
我是多行註解  
可以用3個單引號  
'''
```

```
"""
```

```
也可以用3個雙引號  
啊，其實啊  
多行註解其實就是多行的字串啦  
"""
```

運算子與轉型

運算子

整數、浮點數運算

運算子	名稱	說明
+	加法	
-	減法	
*	乘法	
/	除法	除到小數 (整數/整數 = 浮點數)
//	整除	取小於等於相除後的值的最大整數
%	餘數	除數或被除數是浮點數也能做，餘數與除數同號
**	次方	

運算子

除法在 C 語言是取整數部分 (無條件捨去)
Python 是取小於等於相除後的值的最大整數

C語言

```
#include <stdio.h>

int main(){
    printf("%d", -1 / 2);
}
```

D:\FCU\TA\程設IV\Untitled1.exe

0

python

```
print(-1 / 2)
```

-0.5

```
print(-1 // 2)
```

-1

整除與餘數 (mod)

`-3 // 2`

-2

`-3 % 2`

1

$-3 \div 2 = -2 \dots 1$

`3.2 // -2`

-2.0

`3.2 % -2`

-0.7999999999999998

$3.2 \div -2 = -2.0 \dots -0.8$

`-3 // 2.5`

-2.0

`-3 % 2.5`

2.0

$-3 \div 2.5 = -2.0 \dots 2.0$

`-4.3 // -2.1`

2.0

`-4.3 % -2.1`

-0.09999999999999964

$-4.3 \div -2.1 = 2.0 \dots -0.1$

字串運算 +、*

可以使用 + 和 * 做字串的連接或重複

```
1 a = 'apple'  
2 b = 'pie'  
3 t1 = a + b  
4 t2 = a * 3  
5 print(t1)  
6 print(t2)
```

```
applepie  
appleappleapple
```

字串運算%

可以使用% 將數值使用 format 方式插入字串

```
1 a = 'I have %d apples.'  
2 n = 10  
3 t = a % n  
4 print(t)  
5 s = '%s got %.2f points at the exam.'  
6 name = 'Tony'  
7 score = 66.8  
8 t = s % (name, score) # 有多個數值請用小括號括起  
9 print(t)
```

```
I have 10 apples.  
Tony got 66.80 points at the exam.
```

轉型

使用「型態名稱 ()」函式可以將數值轉型

```
1 a = 10
2 print(a, type(a))
3 t = float(a) # int to float
4 print(t, type(t))
5 t = str(a) # int to str
6 print(t, type(t))
7 s = '123'
8 t = int(s) # str to int
9 print(t, type(t))
```

```
10 <class 'int'>
10.0 <class 'float'>
10 <class 'str'>
123 <class 'int'>
```

比較運算子、布林運算子

運算子	名稱	運算子	名稱
<	小於	>	大於
<=	小於等於	>=	大於等於
==	值相等	and	邏輯 AND
!=	值不相等	or	邏輯 OR
is	reference 相等	not	邏輯 NOT

C 語言中的

`!(-4 <= y && y < 3) || y == 0`

在 Python 要寫成

`not(-4 <= y and y < 3) or y == 0`

或者可以更簡略寫成

`not -4 <= y < 3 or y == 0`

但是 C 語言不能像 Python 寫成 `-4 <= y < 3`，不要搞混！

運算子

比較運算子與布林運算子所回傳的數值是 bool(布林值)

```
1 y = 10
2 b = not -4 <= y < 3 or y == 0
3 print(b)
4 y = -3
5 b = not -4 <= y < 3 or y == 0
6 print(b)
7 print(type(b))
```

```
True
```

```
False
```

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

輸入輸出

print 函式

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

我們主要用到的參數有 objects、sep、end

- objects 前的 * 代表可以放多個參數
- sep 為每個 objects 的間隔 (預設為' ')
- end 為結尾印出的文字 (預設數值為'\n')

使用多個參數或使用% 做 format 輸出

```
1 a = 10
2 b = 5
3 print(a, '+', b, '=', a + b)
4 print('%d + %d = %d' % (a, b, a + b))
```

```
10 + 5 = 15
10 + 5 = 15
```

設定 sep 與 end 參數

```
1 a = 10
2 b = 5
3 print(a, 'apples', 'and', b, 'bananas')
4 print(a, 'apples', 'and', b, 'bananas', sep='_')
5 print(a, 'apples', 'and', b, 'bananas', end='~~\n')
6 print(a, 'apples', 'and', b, 'bananas',
7       sep=' ~ ', end='\\(≥∀≤)/\n')
```



```
10 apples and 5 bananas
10_apples_and_5_bananas
10 apples and 5 bananas~~
10 ~ apples ~ and ~ 5 ~ bananas\\(≥∀≤)/
```

input 函式
`input([prompt])`

參數 `prompt` 為輸入之前的提示字元，
中括號代表可以不用填寫參數。
此函式會回傳字串。

input 函式會回傳字串

```
1 inp1 = input('輸入數字：')
2 inp2 = input('輸入數字：')
3 print(inp1, '+', inp2, '=', inp1 + inp2)
4 print(type(inp1))
```

```
輸入數字：14
輸入數字：23
14 + 23 = 1423
<class 'str'>
```

若要輸入數字，請轉型

```
1 inp1 = int(input('輸入數字：'))
2 inp2 = int(input('輸入數字：'))
3 print(inp1, '+', inp2, '=', inp1 + inp2)
4 print(type(inp1))
```

```
輸入數字：14
輸入數字：23
14 + 23 = 37
<class 'int'>
```


流程控制

Python 流程控制

- if ... elif ... else
- while
- for in

if ... elif ... else

- Python 沒有 switch case, 只有 if elif else
- 而 Python 也沒有大括號, 取而代之的是冒號 (:) 和縮排 (且不能 tab 和空白混用)
- Python 的 if 判斷格式如下:

```
if x < 0:  
    printf("negative")  
elif x == 0:  
    printf("zero")  
    printf("not negative or positive")  
else:  
    printf("positive")
```

大多數的 Python 物件都能當作布林值

- 數字 0、0.0、0+0j 均被當作 false，其餘當作 true
 - 空字串'' 被當作 false，其餘當作 true
 - 空 list[]、空 tuple()、空 dict{} 被當作 false，其餘當作 true
 - None 被當作 false
- (list、tuple、dict 是 Python 的基本資料結構，會在第 2 章之後介紹。)

while

- Python 沒有 do while，只有 while
- Python 的 while 範例如下：

```
1 i = 0
2 while i < 5:
3     print('Loop:', i)
4     i += 1 #Python有+=，但沒有++
```

```
Loop: 0
Loop: 1
Loop: 2
Loop: 3
Loop: 4
```

for in

- Python 的 for 迴圈與 C 語言的 for 有些差別，是作「走訪」而不是遞增遞減。
- 若要做遞增遞減，for 迴圈可以搭配 range 函式產生遞增或遞減的容器作走訪。
- Python 的 for 範例如下：

```
1 for i in range(5):  
2     print('Loop:', i)
```

```
Loop: 0  
Loop: 1  
Loop: 2  
Loop: 3  
Loop: 4
```

range 函式

```
range(stop)  
range(start, stop[, step])
```

- 若參數只有 1 個，則建立 0~stop 的遞增數列 (不包含 stop)
- 若參數有 2 個以上，則建立 start~stop，間隔為 step(預設為 1) 的遞增/遞減數列 (不包含 stop)

for in range 範例

```
1 for i in range(1, 5):  
2     print(i, end=' ')  
3 print() #換行
```

```
1 2 3 4
```

```
1 for i in range(1, 10, 2):  
2     print(i, end=' ')  
3 print() #換行
```

```
1 3 5 7 9
```

```
1 for i in range(5, 1, -1):  
2     print(i, end=' ')  
3 print() #換行
```

```
5 4 3 2
```


關鍵字 break、continue、pass

- break：跳出迴圈
- continue：跳過這次迴圈，繼續下一次迴圈
- pass：不做任何事情

break、continue、pass 範例

```
1 i = 0
2 while True:
3     i += 1
4     print(i, end='')
5     if i % 3 == 0:
6         print(' ###')
7         continue # 若i為3的倍數，進入下一次迴圈
8     elif i % 2 == 0:
9         pass # 若i為偶數，不做事
10    elif i % 7 == 0:
11        break # 若i為7的倍數，跳出迴圈
12    print(' @@@')
13 print(' done')
```

```
1 @@@
2 @@@
3 ###
4 @@@
5 @@@
6 ###
7 done
```

while ... else 與 for ... else

while 迴圈與 for 迴圈也可以加上 else 區塊，
在迴圈正常結束時會進入 else 區塊。

```
1 x = 1
2 while x < 10:
3     print(x)
4     x += 2
5 else:
6     print('done')
```

```
1
3
5
7
9
done
```

但如果是因為 `break` 而跳出迴圈則不會進入 `else` 區塊。

```
1 x = 1
2 while x < 10:
3     print(x)
4     if x == 7:
5         break
6     x += 2
7 else:
8     print('done')
```

```
1
3
5
7
```

基本 Python 風格

Python 的語法相對簡單
但是官方有一些建議的程式撰寫風格慣例
詳見

<https://www.python.org/dev/peps/pep-0008/>

(下頁列出常用的命名慣例)

常用的命名慣例

- 模組、套件名稱：短名詞、全小寫、必要時使用底線
- 函式名稱：全小寫，增加可讀性可用底線，如：my_func()
- 變數名稱：全小寫，增加可讀性可用底線，如：my_var
- 類別名稱：第 1 個字大寫，如：MyClass
- 常數名稱：單字之間加底線，如：MY_CONST
- 縮排：4 個空白鍵，不要用 tab

END