

基礎程式設計

Data Type 、 Variable 、 Operator



Python Fundamental

Outline

- ◆ 型態
- ◆ 變數
- ◆ 運算子
- ◆ 輸出
- ◆ 輸入
- ◆ 註解
- ◆ 格式化

型態介紹

◆ Python 3 內建許多型態，主要包括：

- 數值型態 (numeric type) : **int**、**float**、**bool**
- 文字型態 (text sequence type) : **str**
- 序列型態 (sequence type) : **list**、**tuple**
- 集合型態 (set type) : **set**
- 對映型態 (mapping type) : **dict**

◆ 想知道某個資料的型態，可以使用 **type()** 函數。

```
In [1]: print(type(10))  
<class 'int'>
```

```
In [2]: print(type(10.0))  
<class 'float'>
```

```
In [3]: print(type('XYZ'))  
<class 'str'>
```

數值型態

◆ 數值型態 (**int**、**float**、**bool**)

數值很直覺，諸如3、20、-10、8.5、-2.5都是數值，Python將數值型態分為以下三種：

- **int**：表示整數，沒有小數部分，例如10、-53、10000。
- **float**：表示浮點數，有小數部分，例如3.14159、-123.45、1.5E-3 (1.5×10^{-3})。
- **bool**：表示布林 (boolean)，有**True** (真) 和**False** (假) 兩種。

數值型態 - int (整數) & float (浮點數)

- ◆ int 型態表示整數 (integer)，沒有小數的部分。
- ◆ int 型態不能加上千分位符號，不能使用例如1,000這樣的寫法。

```
In [1]: print(1000)  
1000
```

```
In [2]: print(1000+2000)  
3000
```

```
In [3]: print(1,000)  
1 0
```

- ◆ float 型態表示浮點數 (float point number)，指的是有小數的部分，例如：3.14159、123.321。

```
In [1]: print(123.321)  
123.321
```

```
In [2]: print(123.321+123.321)  
246.642
```

數值型態 - bool (布林)

- ◆ bool 型態表示布林 (boolean) , 有True(真)和False(假)兩種值。
- ◆ bool() 函數會將 0、None、False 與「空的內容」轉換為 False , 其它的值轉換為 True 。

```
In [1]: print(5>3)  
True
```

```
In [2]: print(5<3)  
False
```

```
In [3]: print(bool(0))  
False
```

```
In [4]: print(bool(0.0))  
False
```

```
In [5]: print(bool(0.1))  
True
```

```
In [6]: print(bool())  
False
```

```
In [7]: print(bool(None))  
False
```

```
In [8]: print(bool(True))  
True
```

```
In [9]: print(bool(False))  
False
```

```
In [10]: print(bool(()))  
False
```

```
In [11]: print(bool([]))  
False
```

```
In [12]: print(bool({}))  
False
```

文字型態 - str (字串)

- ◆ Python 使用字串(str) 處理文字資料，包含文字、數字、符號等。
- ◆ Python 使用下列三種語法表示字串：
 - 單引號 (')：例如 'Python程式設計'。(多數 Python 開發者的習慣)
 - 雙引號 ("")：例如 "Python程式設計"。
 - 三個單引號 ("")、三個雙引號 ("""")：例如 ""程式設計""、"""Python程式設計""”。這種語法允許多行字，中間的空白亦包含在內，例子如下：

```
In [1]: print('it is a good day')
it is a good day
```

```
In [2]: print("it is a good day")
it is a good day
```

```
In [3]: print(''' today is Wednesday
....: it is a good day'''')
today is Wednesday
it is a good day
```

跳脫字元

- ◆ 跳脫字元是指該字元具有特殊含義，必須特別對待，通常以倒斜線加上某一固定字元來表示。
- ◆ Python 在遇到某些特定的字元組合時，會把它辨認成其他的意思。常見的跳脫字元如下。

符號	說明
\\	反斜線。
\'	單引號，當你使用 '' 來表示字串，又要表示單引號時使用，例如 'Justin\'s Website'。
\"	雙引號，當你使用 "" 來表示字串，又要表示單引號時使用，例如 "\text\" is a string"。
\n	換行。
\t	Tab。

```
In [1]: print('C:\todo')  
C:      todo
```

```
In [2]: print('C:\\\\todo')  
C:\\todo
```

```
In [3]: print('it\' is')  
it' is
```

```
In [4]: print('\\\"it\\\" is')  
"it" is
```

```
In [5]: print('is\\tit?')  
is      tit?
```

```
In [6]: print('is it ok?\nYes!')  
is it ok?  
Yes!
```

變數介紹與命名規則

- ◆ 變數是我們在程式中所使用的一個名稱 (name)，電腦會配置記憶體位置跟空間給這個名稱，然後我們可以來存放數值、字串、布林，稱為變數的值(value)，而且值可以重新設定或經由運算更改。
- ◆ 第一個字元可以是英文字母、底線(_) 或中文，其它字元可以是英文字母、底線、數字或中文。
- ◆ 英文字母有大小寫之分。
- ◆ 不能使用關鍵字，內建常數、內建函數、內建類別等的名稱。
- ◆ 在多數Python整合開發環境(IDE)中，都會以特殊顏色顯示關鍵字。

合法的變數名稱。

studentID
_studentID
studentName
student_name
myCar1

不合法的變數名稱

class # 不能使用關鍵字
customer@ID # 不能使用特殊符號@
7eleven # 不能以數字開頭
!myName # 不能使用特殊符號！
my Name # 不能包含空白

變數指派方式 - 賦值運算

- ◆ 「=」是賦值運算的運算子，也稱為「指定運算子」，是經常使用的運算子。
- ◆ 「=」代表將右邊運算的結果(右值)賦予(存到)左邊(左值)，跟一般在數學上的「相等」意義有所不同。

left value = right value



- ◆ Python 屬於動態型態程式語言，變數在使用之前無須宣告型態。
- ◆ 我們使用指定運算子 (=) 來指定變數的值，而且是以最近一次指定的值為準。

In [1]: myName='王大明'

In [2]: myName
Out[2]: '王大明'

In [3]: x=3

In [4]: x=x+1

In [5]: x
Out[5]: 4

In [6]: x=y=z=123

In [7]: x
Out[7]: 123

In [8]: y
Out[8]: 123

In [9]: z
Out[9]: 123

變數指派方式 - 賦值運算

- ◆ Python 對於變數不用宣告型態，但仍須設定變數的值，否則會發生錯誤。
- ◆ 以下例子中，我們尚未設定變數x就要使用，導致名稱錯誤(NameError)。

```
In [1]: 123+x
Traceback (most recent call last):

  File "<ipython-input-1-9205c1cfb15e>", line 1, in <module>
    123+x

NameError: name 'x' is not defined
```

變數刪除

◆ 刪除變數 : del 變數

```
In [1]: A=1
```

```
In [2]: B=2
```

```
In [3]: C=3
```

```
In [4]: print(A)  
1
```

```
In [5]: print(B)  
2
```

```
In [6]: print(C)  
3
```

```
In [7]: del A
```

```
In [8]: del C
```

```
In [9]: print(A)  
Traceback (most recent call last):
```

```
File "<ipython-input-9-939330b76721>",<br>    print(A)
```

```
NameError: name 'A' is not defined
```

```
In [10]: print(B)  
2
```

```
In [11]: print(C)  
Traceback (most recent call last):
```

```
File "<ipython-input-11-085b32debabd>",<br>    print(C)
```

```
NameError: name 'C' is not defined
```

變數型態

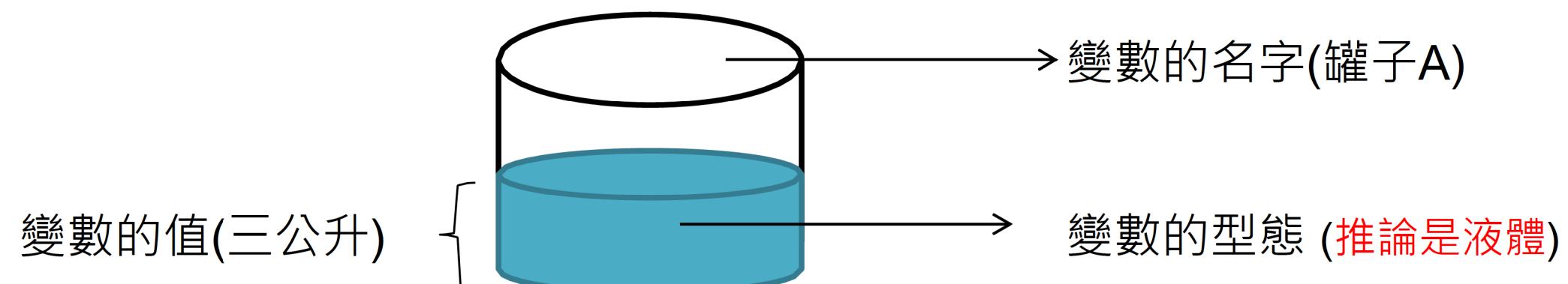
◆ $A = 3$

變數值同時可以看出變數型態

- A : 變數名稱。
- 3 : 變數的值。
- 型態 : int (從變數的值推論而來)



變數名稱必須符合規範



變數的型態 (推論是液體)

變數型態

◆ Python的變數可以更換型態：

- C=5 → 從5回推型態是整數 int
- C='ABC' → 從'ABC'回推型態是字串str
- C=8.567 → 從8.567回推型態是浮點數 float

◆ 我們可輸入type(變數名稱)函數 知道其型態，輸入變數名稱可看到變數的值。

```
In [1]: X=3
```

```
In [2]: print(type(X))  
<class 'int'>
```

```
In [3]: X='ABC'
```

```
In [4]: print(type(X))  
<class 'str'>
```

```
In [5]: X=True
```

```
In [6]: print(type(X))  
<class 'bool'>
```

```
In [7]: X=789.987
```

```
In [8]: print(type(X))  
<class 'float'>
```

型態轉換

- ◆ 型態轉換是將不同型態的變數轉換成同一個型態。

➤ `a = 1`

➤ `b = 3`

➤ `c = 5.5`

➤ `d = 7.5`

➤ `print(int(a + c), int(a +d))`

6 8 (小數以下無條件捨去)

➤ `print(float(a + b))`

4.0

型態轉換

◆ 型態轉換範例

1. 字串轉整數

```
In [1]: print(int('10'))  
10
```

2. 浮點數轉整數

```
In [2]: print(int(3.1415))  
3
```

3. 布林轉整數

```
In [3]: print(int(True))  
1
```

```
In [4]: print(int(False))  
0
```

4. 將數值轉換為字串

```
In [5]: print(str(1.2345))  
1.2345
```

型態轉換

◆ 型態轉換範例

1. 字串轉整數

```
In [1]: int('10')
Out[1]: 10
```

2. 浮點數轉整數

```
In [2]: int(3.14)
Out[2]: 3
```

3. 布林轉整數

```
In [3]: int(True)
Out[3]: 1
```

```
In [4]: int(False)
Out[4]: 0
```

4. 十進位整數轉八進位

```
In [5]: oct(10)
Out[5]: '0o12'
```

5. 十進位整數轉十六進位

```
In [6]: hex(10)
Out[6]: '0xa'
```

6. 將數值轉換為字串

```
In [1]: str(1.23)
Out[1]: '1.23'
```

運算子

- ◆ Python 「運算式」是由「運算元」和「運算子」組成。

Expression = Operands + Operators

運算式 = 運算元 + 運算子

- ◆ 運算元 (operand) 是運算子進行運算的對象。
- ◆ 運算子 (operator) 是一種用來進行運算的符號。
- ◆ 運算式 (expression) 則是運算元與運算子組成的敘述。
- ◆ $6 * 8$ 是運算式，其中「*」為乘法運算子，「6」和「8」為運算元。



運算子

◆ Python的運算子主要分為下列幾種類型：

- 算術運算子 (arithmetic operator) : + 、 - 、 * 、 / 等
- 比較運算子 (comparison operator) : > 、 < 、 >= 、 <= 、 ++ 、 !+ 等
- 指派運算子 (assignment operator) : += 、 -= 、 *= 、 /= 、 >>= 、 %= 等
- 邏輯運算子 (logical operator) : and 、 or 、 not 等

算術運算子

◆ + (加法) : $a + b$, 例如 :

```
In [1]: print(1+2)  
3
```

```
In [2]: print(1.5+2.3)  
3.8
```

◆ - (減法) : $a - b$, 例如 :

```
In [1]: print(15-3)  
12
```

```
In [2]: print(20-True)  
19
```

◆ * (乘法) : $a * b$, 例如 :

```
In [1]: print(12*21)  
252
```

```
In [2]: print('ABC'*2)  
ABCABC
```

◆ / (浮點數除法) : a / b , 例如 :

```
In [1]: print(6/3)  
2.0
```

```
In [2]: print(12/True)  
12.0
```

◆ // (整數除法) : $a // b$, 例如 :

```
In [1]: print(8//3)  
2
```

```
In [2]: print(108//11)  
9
```

◆ % (餘數) : $a \% b$, 例如 :

```
In [1]: print(28%10)  
8
```

```
In [2]: print(12.5%10)  
2.5
```

◆ ** (指數) : $a ** b$, 例如 :

```
In [1]: print(5**3)  
125
```

```
In [2]: print(3**5)  
243
```

比較(關係)運算子

◆ 比較(關係)運算子用來比較兩個運算元的大小或相等與否，真的傳回True，否則傳回False。

運算子	語法	說明	範例
>	$a > b$	若 a 大於 b ，就傳回 True，否則傳回 False	$18 + 3 > 18$ 會得到 True
<	$a < b$	若 a 小於 b ，就傳回 True，否則傳回 False	$18 + 3 < 18$ 會得到 False
\geq	$a \geq b$	若 a 大於等於 b ，就傳回 True，否則傳回 False	$18 + 3 \geq 21$ 會得到 True
\leq	$a \leq b$	若 a 小於等於 b ，就傳回 True，否則傳回 False	$18 + 3 \leq 21$ 會得到 True
\equiv	$a \equiv b$	若 a 等於 b ，就傳回 True，否則傳回 False	$21 + 5 \equiv 18 + 8$ 會得到 True
\neq	$a \neq b$	若 a 不等於 b ，就傳回 True，否則傳回 False	$21 + 5 \neq 18 + 8$ 會得到 False

In [1]: `print(123=='123')`
False

In [4]: `print(False==0)`
True

In [2]: `print('ABC'=='abc')`
False

In [5]: `print(3>5==True)`
False

In [3]: `print(True==1)`
True

指派運算子

- ◆ 指派運算子用來進行指派運算，以下為Python所提供的指派運算子。

運算子	範例	說明
=	a = b	將 b 指派給 a，也就是將 a 的值設定為 b 的值
+=	a += b	相當於 $a = a + b$ ，+ 為加法運算子， 也就是將 a 的值設定為 a 原來的值加 b 的值
-=	a -= b	相當於 $a = a - b$ ，- 為減法運算子
*=	a *= b	相當於 $a = a * b$ ，* 為乘法運算子
/=	a /= b	相當於 $a = a / b$ ，/ 為浮點數除法運算子
//=	a //= b	相當於 $a = a // b$ ，// 為整數除法運算子
%=	a %= b	相當於 $a = a \% b$ ，% 為餘數運算子
=	a **= b	相當於 $a = a ** b$ ， 為指數運算子

指派運算子

◆ 指派運算子範例

```
In [1]: x,y,z=10,20,30 # 將變數 x, y, z 設定為 10, 20, 30
```

```
In [2]: x*=y # 相當於 x=x*y
```

```
In [3]: print(x) # 顯示變數x的值  
200
```

```
In [4]: z%=5 # 相當於 z=z%5
```

```
In [5]: print(z)  
0
```

邏輯運算子

◆ 邏輯運算子可以用來進行邏輯運算，Python提供如下的邏輯運算子。

- and 語法為`a and b`，若兩者均為True就傳回True，否則傳回False。

a	b	a and b
True	True	True
True	False	False
False	True	False
False	False	False

```
In [1]: print(3>2 and 2>3)  
False
```

- or語法為`a or b`，若兩者均為False就傳回False，否則傳回True。

a	b	a or b
True	True	True
True	False	True
False	True	True
False	False	False

```
In [2]: print(3>2 or 2>3)  
True
```

- not語法為`not a`，若a的值為True，就傳回False，否則傳回True。

a	not a
True	False
False	True

```
In [3]: print(not 3>2)  
False
```

Python 程式碼註解

- ◆ Python提供下列兩種註解符號：

- #：標示單行註解，例如：

```
# 我的第一個Python程式  
print("Hello, World!")
```

- ""：標示多行註解，例如：

```
"" 我的第一個Python程式  
它將會印出Hello, World!  
""
```

輸入

- ◆ 使用 `input()` 函數可以取得使用者輸入的資料，語法如下。
- ◆ 提示句是選擇性參數，用來設定提示文字。若省略不寫，表示採取預設值 `None` (無)，也就是沒有提示文字。

`input('提示句')`

- ◆ 我們可以輸入下面第一行敘述，然後按「Enter」鍵，此時會出現提示文字「請輸入姓名：」。
輸入「John」，然後按「Enter」鍵，所輸入的姓名就會被指派給變數 `name`。

In [1]: `name=input('請輸入姓名：')`

[請輸入姓名 : Peter]

In [2]: `print(name)`
Peter

輸入

- ◆ `input()` 函數所輸入的值均為字串，若要將輸入的值進行運算時，可以使用`eval()`函數來處理。
- ◆ `eval()` 函數：將 `input()` 函數取得的字串轉換成數值型態(整數或浮點數) 或是bool型態(`True`或`False`)。
- ◆ 輸入範例：

```
PI = 3.14159
```

```
radius = eval(input("請輸入圓半徑 : "))
```

```
print("半徑為", radius, "的圓面積為", PI * radius * radius)
```

請輸入圓半徑 : 5

半徑為 5 的圓面積為 78.53975

輸出

- ◆ 大部份程式在執行完畢後會將結果輸出到螢幕。
- ◆ 使用內建的 `print()` 函數在螢幕上印出指定的字串，語法如下：

`print(value, values ,..., sep = ' ', end = '\n', file= sys.stdout)`

- `value`：這個參數用來設定要印出的值。若有多個值，中間以逗號隔開。
- `sep`：選擇性參數，用來隔開兩個值的字串，可省略不寫，表示採取預設空格。
- `end`：選擇性參數，用來印出最後一個值後所要加上的字串，可以省略不寫，表示採取預設值 '`\n`' (換行)。
- `file`：選擇性參數，用來輸出的裝置，可以省略不寫，表示採取預設值 `sys.stdout` (標準輸出，即螢幕)。

輸出

print(*value, values ,..., sep = ' ', end = '\n', file= sys.stdout*)

- ◆ *value*：這個參數用來設定要印出的值。若有多個值，中間以逗號隔開。

```
In [1]: print(1)  
1
```

```
In [2]: print(1,2.5,True,'xyz')  
1 2.5 True xyz
```

- ◆ *sep*：選擇性參數，用來隔開兩個值的字串，可省略不寫，表示採取預設空格。

```
In [1]: name='Bob'
```

```
In [2]: print('Hello',name,sep='***')  
Hello***Bob
```

輸出

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

- ◆ end : 選擇性參數，用來印出最後一個值後所要加上的字串，可以省略不寫，表示採取預設值 '\n' (換行)。

```
1 print(123,23.56,True, "abc", sep=":::")
2 print("testa")
3
4 print(123,23.56,True, "abc", sep=":::", end="---")
5 print("testb")
6 print("testc")
```

```
In [1]: runfile('C:/Users/ASUS/.spyder-py3/temp.py',
               wdir='C:/Users/ASUS/.spyder-py3')
123:::23.56:::True:::abc
testa
123:::23.56:::True:::abc---testb
testc
```

輸出

◆ 輸出範例：

```
In [1]: print('I', 'am', 'Jean')  
I am Jean
```

```
In [2]: print('I', 'am', 'Jean', sep='@')  
I@am@Jean
```

```
In [3]: print('I', 'am', 'Jean', end='~~~')  
I am Jean~~~
```

```
In [4]: myName = 'Jean'
```

```
In [5]: print('I', 'am', myName)  
I am Jean
```

印出三個字串，中間以空白隔開

印出三個字串，中間以@隔開

印出三個字串，最後加上 ~~~

將變數myName的值設定為 'Jean'

印出兩個字串和變數 myName 的值

格式化字符串

◆ 目前的Python 3 支援兩種格式化方式

- 舊式（從Python 2 就存在）
- 新式（從Python 2.6、2.7 開始支援）

◆ 舊的字串格式化是使用 `string % data`或 `string % (data1, data2, ...)`

```
>>> '哈囉！%s！' % '世界'
'哈囉！世界！'
>>> '你目前的存款只剩 %f 元' % 1000
'你目前的存款只剩 1000.000000 元'
>>> '%d 除以 %d 是 %f' % (10, 3, 10 / 3)
'10 除以 3 是 3.333333'
>>> '%d 除以 %d 是 %.2f' % (10, 3, 10 / 3)
'10 除以 3 是 3.33'
>>> '%5d 除以 %5d 是 %.2f' % (10, 3, 10 / 3)
'    10 除以      3 是 3.33'
>>> '%-5d 除以 %-5d 是 %.2f' % (10, 3, 10 / 3)
'10      除以 3      是 3.33'
>>> '%-5d 除以 %-5d 是 %10.2f' % (10, 3, 10 / 3)
'10      除以 3      是          3.33'
>>>
```

格式化字符串輸出相關符號

符號	說明
%%	因為%符號已經被用來作為控制符號前置，所以規定使用%%才能在字符串中表示%。
%d	10 進位整數。
%f	10 進位浮點數。
%g	10 進位整數或浮點數。
%s	字串格式符號。

格式化字符串輸出

- ◆ 將數值轉換為不同的進制。
- ◆ 可以使用如下字母來將數字轉換成字母代表的進制：

- d ecimal：十進位
- he x：十六進位
- o ctal：八進位
- b inary：二進位

```
In [1]: print('{0:d}-{0:x}-{0:o}-{0:b}'.format(20))
20-14-24-10100
```

格式化字符串输出

- ◆ `print('格式化方式'.format(數字/字串))`

```
In [1]: print('{:.2f}'.format(3.1415926))
3.14
```

```
In [2]: print('{:10s}'.format('ABCDE'))
ABCDE
```

格式化字符串輸出

數字	格式	輸出	說明
3.1415926	{:.2f}	3.14	保留小數點後兩位
3.1415926	{:+.2f}	+3.14	帶符號保留小數點後兩位
-1		-1.00	
2.71828	{:.0f}	3	不帶小數
5	{:0>2d}	05	數字補零(填充左邊, 寬度為2)
10	{:x<4d}	10xx	數字補x (填充右邊, 寬度為4)
1000000	{:,}	1,000,000	以逗號分隔的數字格式
0.25	{:.2%}	25.00%	百分比格式
1000000000	{:.2e}	1.00e+09	科學指數表示法
13	{:10d}	13	右對齊(默認預設值, 寬度為10)
13	{:<10d}	13	左對齊(寬度為10)
13	{:^10d}	13	中間對齊(寬度為10)

格式化字符串输出

```
In [1]: print('{0:d} 除以 {1:d}是{2:f}'.format(10, 3, 10/3))  
10 除以 3是3.333333
```

```
In [2]: print('{0:5d} 除以 {1:5d}是{2:10.2f}'.format(10, 3, 10/3))  
10 除以 3是 3.33
```

```
In [3]: print('{n1:5d} 除以 {n2:5d}是{n:,.2f}'.format(n1=10, n2=3, n=10/3))  
10 除以 3是3.33
```

```
In [4]: print('{n1:<5d} 除以 {n2:<5d}是{n:,.2f}'.format(n1=10, n2=3, n=10/3))  
10    除以 3    是3.33
```

```
In [5]: print('{n1:>5d} 除以 {n2:>5d}是{n:,.2f}'.format(n1=10, n2=3, n=10/3))  
10  除以 3  是3.33
```

```
In [6]: print('{n1:!^5d} 除以 {n2:?^5d}是{n:,.2f}'.format(n1=10, n2=3, n=10/3))  
!10!! 除以 ??3??是3.33
```

Q & A