Python

常用格式

- 用;號來縮排
- 用\來換行打指令
- ()也有換行的功用

data type

- 有原始(primitive)跟複合(composite)兩種

Python所有的(?)內建資料型別如下:

None (虛無)	NoneType (唯一有大寫字母的內建型別)
Boolean (眞假)	bool
Text (文字)	str
Numeric (數字)	int, float, complex
Sequence (序列)	tuple, range, list
Set (集合)	set, frozenset
Mapping (對應)	dict
Binary (二元)	bytes, bytearray, memoryview

原始資料型別(primitive)

- NoneType:只有None這個元素
- bool:只有True False兩種資料
- bool(False) = bool(0) = bool("") = False

  and 語法
  會回傳第一個是 False 的元素，若沒有則回傳最後一個元素
  or 語法
  會回傳第一個是 True 的元素，若沒有則回傳最後一個元素

- str運算
- 特殊字元

- \newline: Backslash and newline ignored
- \\: Backslash (\)
- \': Single quote (')
- \": Double quote (")
- \b: ASCII Backspace (BS)
- \a: ASCII Bell (BEL)
- \f: ASCII Formfeed (FF)
- \n: ASCII Linefeed (LF)
- \r: ASCII Carriage Return (CR)
- \t: ASCII Horitonal Tab (TAB)
- \v: ASCII Vertical Tab (VT)
- \ooo: Character with octal value ooo
- \xoo: Character with hexadecimal value oo

- "abc"*3="abcabcabc"
- "abc" + "cbd" = "abccdb"
- 字串比大小是看第一個字的字典序: 中文 > 英文小寫 >英文大寫 >數字
- length

```
> len("akdjlfjdk")
> 9
```

- u b r f 格式

```
> ord("台")
> 21488
> chr(21488)
> "台"
> "台灣大學".encode("utf-8")
> b'\x  \x  \x.....
> b'\x  \x  \x..... .decode("utf-8")
> "台灣大學"
```

- 尋找字串

```
> "djfdklsa".find("f")
> 2
> "fdjksala".count("a")
> 2
> "djfdklsa".find("p")
> -1 #找不到會回傳-1
> "fdjksala".count("w")
> 0  #沒數到會回傳 0
```

- 運算字元

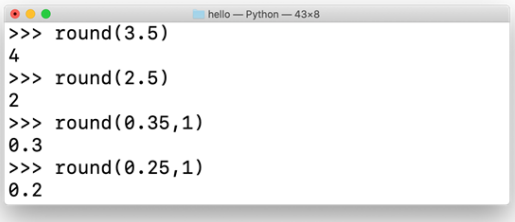
```
1+1=2
1-1=0
1*1=1
1/1=1
1**1=1
1//1=1  #除完後無條件捨去 商
1%1=0   #取餘數
```

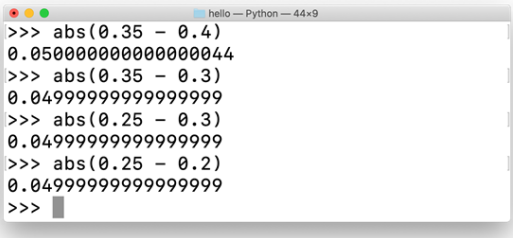
- 進位制

```
二進位
0b開頭
bin(str)
八進位
0o開頭
oct(str)
十進位
.zfill(8)
十六進位
0x開頭
hex(str)
```

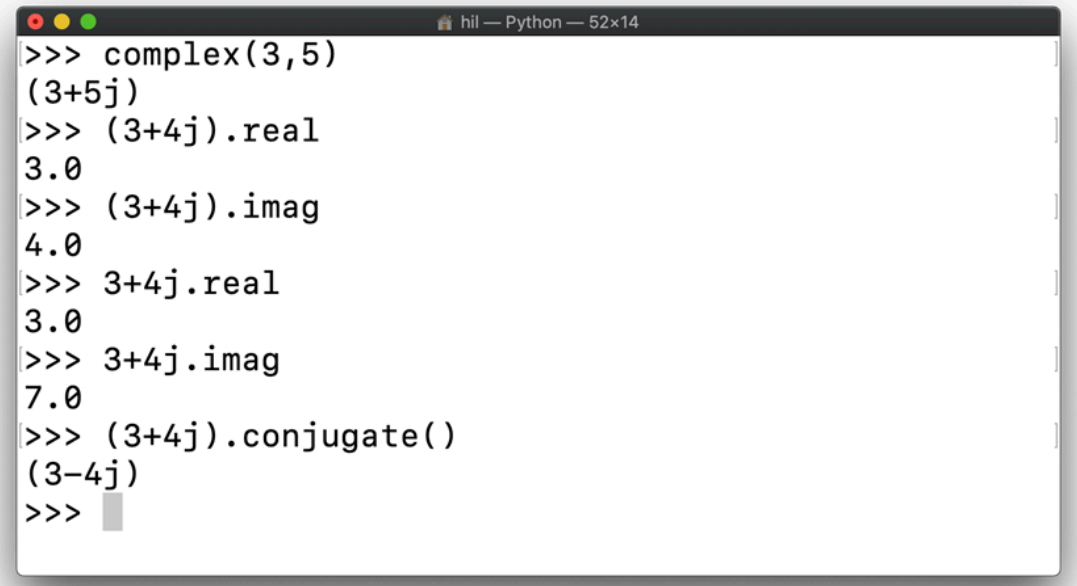
  - 估算

```
round 是將數字往最近的整數移動，如果一樣時以偶數為優先
abs   與原點的距離
```





- 複數運算(cmath::)



複合資料型

tuple

- 運算

```
> ((((("1")))) == "1" #因為都表示"1"這個字元
> True
in #找元素有沒有在tuple中
```

```
> 5, 5, 6, 6 == (5, 5, 6, 6) #先比較6 == (5, 5, 6, 6)
> 5, 5, 6, False
index #找元素第一次出現的位置
> (5, 5, 6, 6).index(6) #找不到會error
> 2
> max((1,2,3)) #只能放相同資料型態
> 3
> (3)   #宣告一個整數
> 3
> (3,)   #宣告一個tuple
> (3,)
> len(5, 5, 6, 6) #len裡只能放一個資料因此會出錯
> error!
```

- 轉換成tuple

```
> tuple(5423)
> (5, 4, 2, 3)
```

- 將tuple變成str

```
> "**".join(("我","愛","吃飯"))
> 我**愛**吃飯
```

- tuple(range)

```
> tuple(range(0,6,2))
> (0, 2, 4)
```

- enumerate tuple

```
> tuple(enumerate("有","你","真","好"), 2020)
> ((2020, "有"),(2021, "你"),(2022, "真"),(2023, "好"))
```

list

- 就是mutable tuple 可變動的tuple

```
> [[[["1"]]]] == "1" #一邊是list一邊是字元
> False
```

- 運算

```
list.pop(index)
list.insert(index, element)
list.remove(element)
list.append(element)
list.extend(list_2)
del list[]
```

- 排列

```
list.reverse
list.sort #小排到大
sorted(list)
list.sort(reverse = T) #大排到小
list[] #有將list變動的特性
```

- 用range 宣告list

```
> list[range(2)]
> [0, 1]
> list[i for i in range(5)]
> [0, 1, 2, 3, 4, 5]
```

- list enumerate

```
> list(enumerate(["a","b","c","d"]))
> [(0,"a"),(1,"b"),(2,"c"),(3,"d")]
```

- list split

```
> str.split(str)
> list
```

set

- 運算

```
><=是集合中的包含
- 是差集的意思
^反交集
|聯集
&交集
+*不存在
.add()
.discard()
.remove()
.clear()
```

- 集合的成員只能是immutable的，不能是list或是set

frozenset

- 宣告

```
R = frozenset({1,2})
```

- 資料轉換中的參數只能有一個

```
> set([1,2,3])
> {1,2,3}
> set(1,2,3)
> error
```

- 速度　tuple比list快；frozenset比set快

dict

- key要immutable
- dict比list慢
- dict的元素

```
dict.pop
dict.items
dict.keys
dict.values
dict.update
dict_1|dict_2 #python3.9後才能聯集
```

- 沒有><=的運算
- 聯集時是後面蓋掉前面
- fromkeys宣告dict

```
a = dict.fromkeys(key, value) #key可以是一個複合資料型別，value沒宣告則是N
```

變數

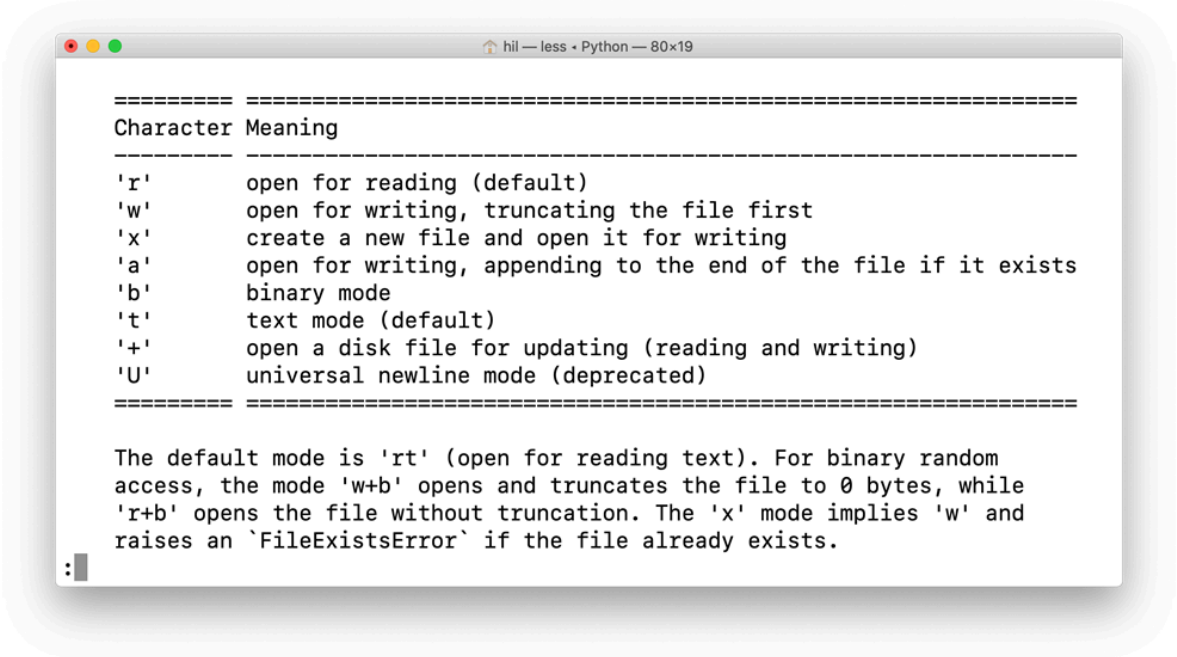
輸出

- print

```
print(variable, sep = "", end = "\n", file = sys.stdout, flush = False)
#flush 是強制寫入檔案中的意思 #寫入預設的檔案sys.stdout
```

- file 變數

```
x = open("anc.txt", "x, w, a or r") #x只能寫入不存在的檔案，w是複寫，r是讀檔
```



輸入

- input

```
> input("請輸入整數n =")
> 3 #3是字串
> int(input("請輸入整數n ="))
> 3 #3是整數
> int(input("請輸入整數n ="))
> 6*3
> error #input無法判別函式或運算
```

- eval #eval 可以自動把數字、算式判別成 int 或是 float

```
> n = eval(input("請輸入整數n ="))
> 59+98*8
> n
> 843
```

- 檔案指令

```
> myfile = open("myfile.txt", "r")
> a = myfile.readline(); a  #讀入一行
> a = myfile.readline(); a  #再讀入下一行
> a = myfile.readline(9); a #再讀入下一行且不超過9個字
> a = myfile.read(9); a     #讀入檔案不超過9個字
> a = myfile.readlines(9); a#讀入若干行，那一行超過9個字就不再讀下一行
```

判斷

- if

```
if condition_1:
    task_1
elif condion_2:
    task_2
else:
    task_3
```

- upper

```
str.upper() #將str的所有文字換成大寫
```

- 縮寫

```
x if b else y
#上下是一樣的
if b:
    x
else:
    y
```

迴圈

- while

```
while condition:
    task_loop
task_break
```

- for

```
for variable in something:
    task_loop
e.g.
> for i in "168":
        print i
> 1
> 6
> 8
> for i in {1:6, 5:8}:
        print(i)
> 1
> 5
```

- python 的資料型態項便利貼
- 變數特性
    - 特殊情況下寫在不同行也會全等

特殊狀況:真假,小整數(-5..256)與許多短字串(<4097)

```
#原始資料型別在資料大小很小時，同樣的值會存在相同的地址；兩種情況1.值是很
                                                    #2.值是比較大的
> x = 3
> y = 3
> x is y
> True
#但在值太大時會獨立建立在不同地址，若是寫在同一行還是全等
```

```
> x = 5e+9999
> y = 5e+9999
> x is y
> False
```

- 地址函式

```
> id(variable)
> address
> a is b #等價於 id(a) == id(b)
```

- 交換變數值

```
a, b = b, a
> a = 5
> b = 4
> a, b = b, a
> a, b
> 4, 5
```

- 複合型變數的例外

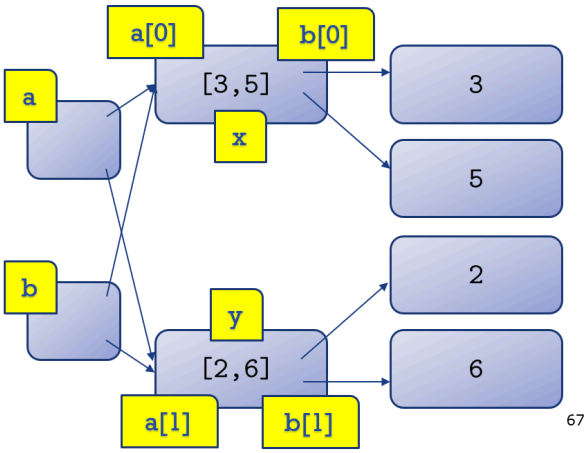
```
> x = ()
> y = ()
> x is y
> True
> x = frozenset({})
> y = frozenset({})
> x is y
> true                    #dict 跟 set 就沒有這種性質
```

- sys.intern

```
#強迫把資料放在同一個位置
> import(sys)
```

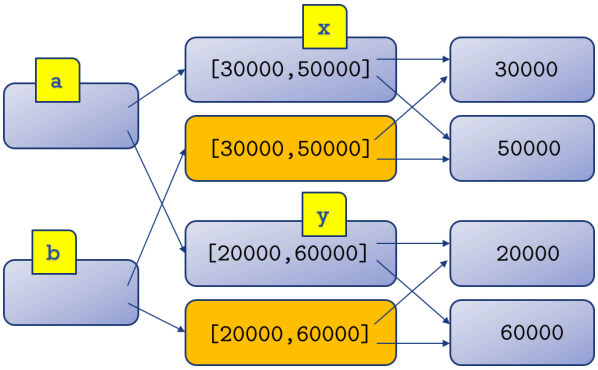
```
> x = sys.intern("a"*999)
> y = sys.intern("a"*999)
> x is y
> True
```

- 變數沒貼上便利貼很快就會被清除
- 深拷貝與淺拷貝



67

```
a = b.copy
```




```
a = copy.deepcopy(b)
```

函式

- exec

```
#將字串消除
a = "def function(n):print(n)"
> exec(a)
> function(n)
> n
```

參數

- 加*號(但是下一個參數就很麻煩)

```
#有*號代表為許多參數
> def f(x, *y):print(y) print(*y)
> f(1, 5, 6)
> [5, 6]
> 5, 6
```

```
> def f(x, *y, z):print(y) print(*y)
> f(1, 2, 3, 4)
> error!
> f(1, 2, 3, z=4)
> [2, 3]
> 2, 3
```

- 參數有命名的話就可以調換順序

```
def f(x, y, z):print(xyz)
f(y=5, x=6, z=9)
```

- 定位參數與關鍵字參數

```
#/前的參數不能指定名稱，因此也不能調換順序
#*號的參數要指定名稱，而且輸入時也要指定名稱
```

- 加雙重星號式dict的意思

```
> def f(**d):print(d) print(**d)
> f(a = 5,b = 3,c = 9)
> {"a":5, "b":3<, "c":9}
> "a":5, "b":3<, "c":9
```

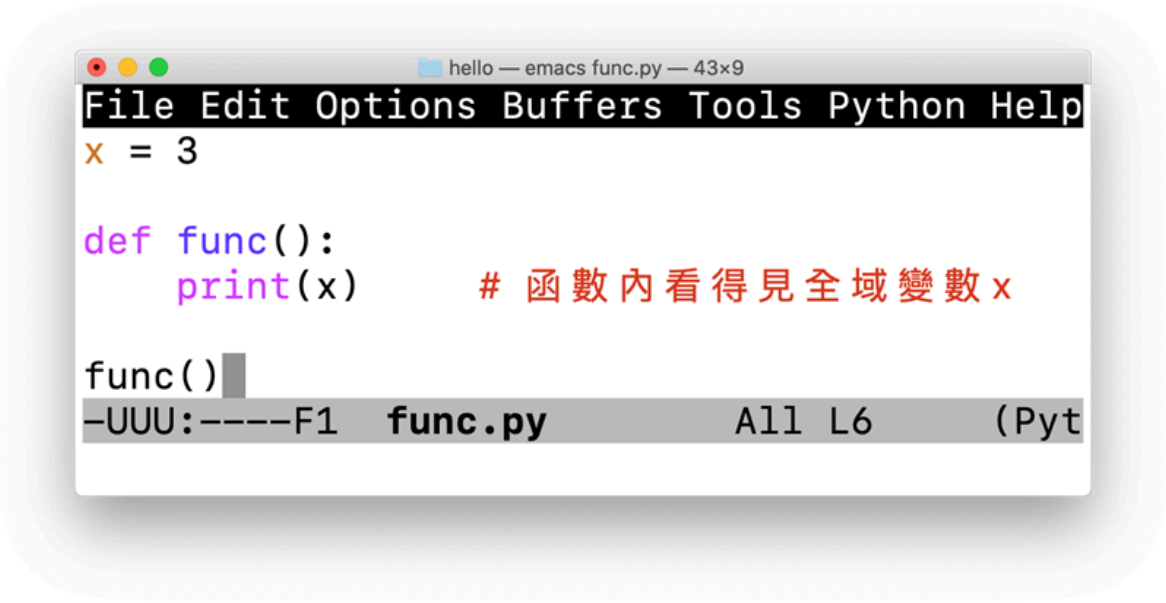
- sorted, filter, map, reduce

```
map(function, list)
filter([boolin], list)      #boolin的list如:[True, False]
reduce(function, list)      #將function兩兩套在list的元素中
```

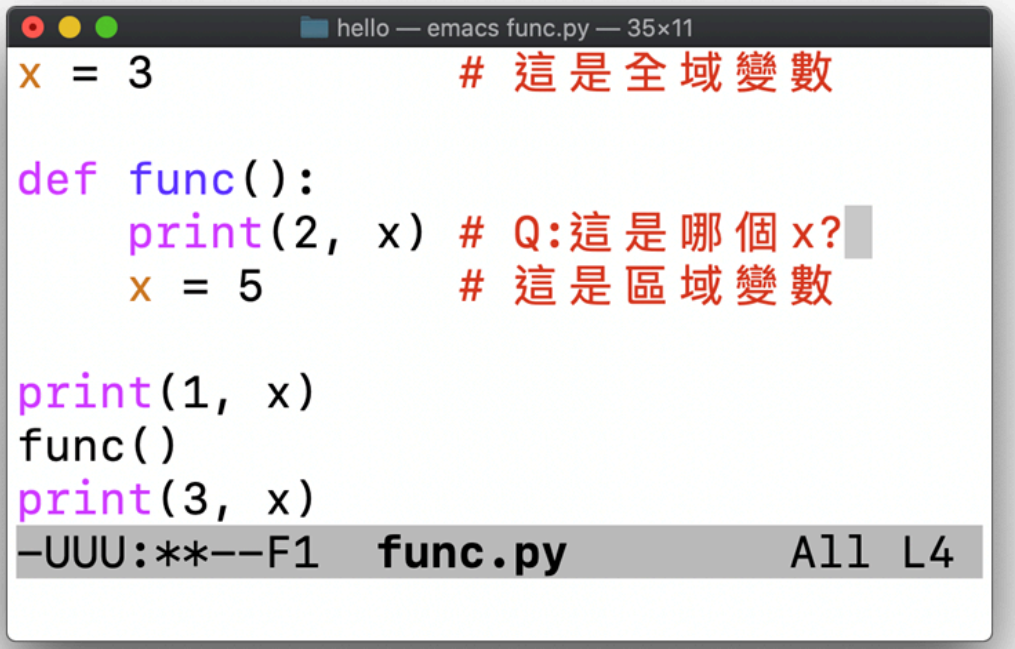
- 純賦值

```
:=
可以用在function的參數欄裡，因為用"="會跟參數命名衝突
```

- 最外面define的function在global
- 變數要賦值後才有global local的比較
- 沒有local的變數才會去找global同名的變數，下圖的結果是3



- 載入一個函數會先偵測整體的參數，但是不會偵測賦值，賦值在順向運算時才會載入，所以會error



- 給x 賦值就會產生上面的問題