

個人資料

- 鍾健雄
- 美國維吉尼亞大學 - 系統暨資訊工程博士
- 精誠樓 I-302
- jschung@cc.cust.edu.tw
- 0919341293



Python程式設計

- Python 基本語法 (3w)
- Python 物件導向程式 (3w)
- 基本資料結構與演算法 (2w)
- Python 資料庫處理 (3w)
- Python 科學繪圖 (matplotlib) (3w)
- Python 陣列與矩陣 (numpy) (2w)
- Python/Arduino API (self project)

Python程式語言介紹

Python 官方網站

[**https://www.python.org/**](https://www.python.org/)

Python程式語言概論

- Python是一種物件導向、直譯式的電腦程式語言。非常適合完成各種高階任務。
- 功能完備的標準程式庫，能夠輕鬆完成很多常見的任務。
- 語法簡單，它使用縮排來定義程式區塊(其它大多數程式語言使用大括弧)。
- 動態語言
- Python具備垃圾回收功能，自動管理記憶體使用。
- 常被當作腳本語言用於處理系統管理任務和網路程式編寫

- Python虛擬機本身幾乎可以在所有的作業系統中運行。
- 使用py2exe、PyPy、PyInstaller工具可以將Python原始碼轉換成可執行檔。
- Python的官方直譯器是Cpython，用C語言編寫，是一個自由軟體，由Python軟體基金會管理。
- Python支援命令式程式設計、物件導向程式設計、函數式編程、面向側面的程式設計、泛型編程多種編程範式。

Python歷史

- Python的創始人Guido van Rossum，目前仍是Python的主要開發者，決定整個Python發展方向。
- Python 2.0於2000年10月16日發布，增加了實現完整的垃圾回收，支援Unicode。開發過程透明，社群對開發進度的影響大。
- Python 3.0於2008年12月3日發布，此版不完全相容之前的Python原始碼。很多新特性後來也被移植到舊的Python 2.6/2.7版本。

程式語言特性

- Python是完全物件導向的語言。
- 函式、模組、數字、字串都是物件。
- 完全支援繼承、重載、衍生、多重繼承，有益於增強原始碼的複用性。
- Python支援重載運算符，因此Python支援泛型設計。
- Python對函數式設計只提供了有限的支援。
- 有兩個標準庫（`functools`, `itertools`）提供了與Haskell和Standard ML中類似的函數式程式設計工具。

- Python本身被設計為可擴充的。並非所有的特性和功能都整合到語言核心。
- Python提供了豐富的API和工具，以便程式設計師能夠輕鬆地使用C、C++、Cython來編寫擴充模組。
- Python編譯器本身也可以被整合到其它需要腳本語言的程式內。Python視為一種「膠水語言」（glue language）。
- Google儘量使用Python，在操控硬體的時候使用C++，在快速開發時候使用Python。」

Python的設計哲學

- 「優雅」、「明確」、「簡單」
- 「用一種方法，最好是只有一種方法來做一件事」

應用領域

- Python用於Web開發
 - 通過Python定義了WSGI標準應用介面來協調Http伺服器與基於Python的Web程式之間的溝通。
 - 一些Web框架，如Django、Pyramid、TurboGears、Tornado、web2py、Zope、Flask等，可以讓程式設計師輕鬆地開發和管理複雜的Web程式。
 - Python對於各種網路協定的支援很完善，常被用於編寫伺服器軟體、網路爬蟲。
 - Twisted函式庫支援非同步線上編寫程式和多數標準的網路協定（包含用戶端和伺服器），提供了多種工具，用於編寫高效能的伺服器軟體。
 - gevent函式庫，支援高效能高並行的網路開發。

- Python支援GUI開發
 - Python的Tkinter庫能夠支援簡單的GUI開發。
 - wxPython或PyQt等GUI套件來開發跨平台的桌面軟體。執行速度快，與用戶的桌面環境相契合。
 - PyInstaller還能將程式釋出為獨立的安裝程式包。

- 系統作業支援
 - 很多作業系統將Python納為標準系統元件。
 - 大多數Linux和Mac OS X整合了Python，在終端機下直接執行Python。
 - 一些Linux發行版的安裝器使用Python語言編寫，比如Ubuntu的Ubiquity安裝器、Red Hat Linux和Fedora的Anaconda安裝器。
 - Python標準庫包含了多個調用作業系統功能的函式庫。
 - 通過pywin32這個第三方軟體包，Python能夠存取Windows的COM服務及其它Windows API。
 - 使用IronPython，Python程式能夠直接調用.Net Framework。

- 支援科學運算
 - NumPy、SciPy、Matplotlib科學計算函式庫。
 - 機器學習函式庫--Scikit-learn
 - 資料分析函式庫—Panda
 - 支援Hadoop大數據運算環境—pyspark
 - Google開發維護的開源機器學習庫--TensorFlow

- 資料庫
 - SQL : MySQL, PostgreSQL, SQLite...
 - Nosql : MongoDB, Neo4j

- 其他
 - HTML/XML解析器—BeautifulSoup
 - 功能強大圖像處理，支援廣泛圖形檔案格式—PIL
 - 多媒體開發和遊戲軟體開發模組—PyGame
 - Kivy
 - 將python指令碼轉換為windows上可執行檔--Py2exe
 - HTTP函式庫，簡化HTTP請求所需要的代碼量--Requests

標準函式庫

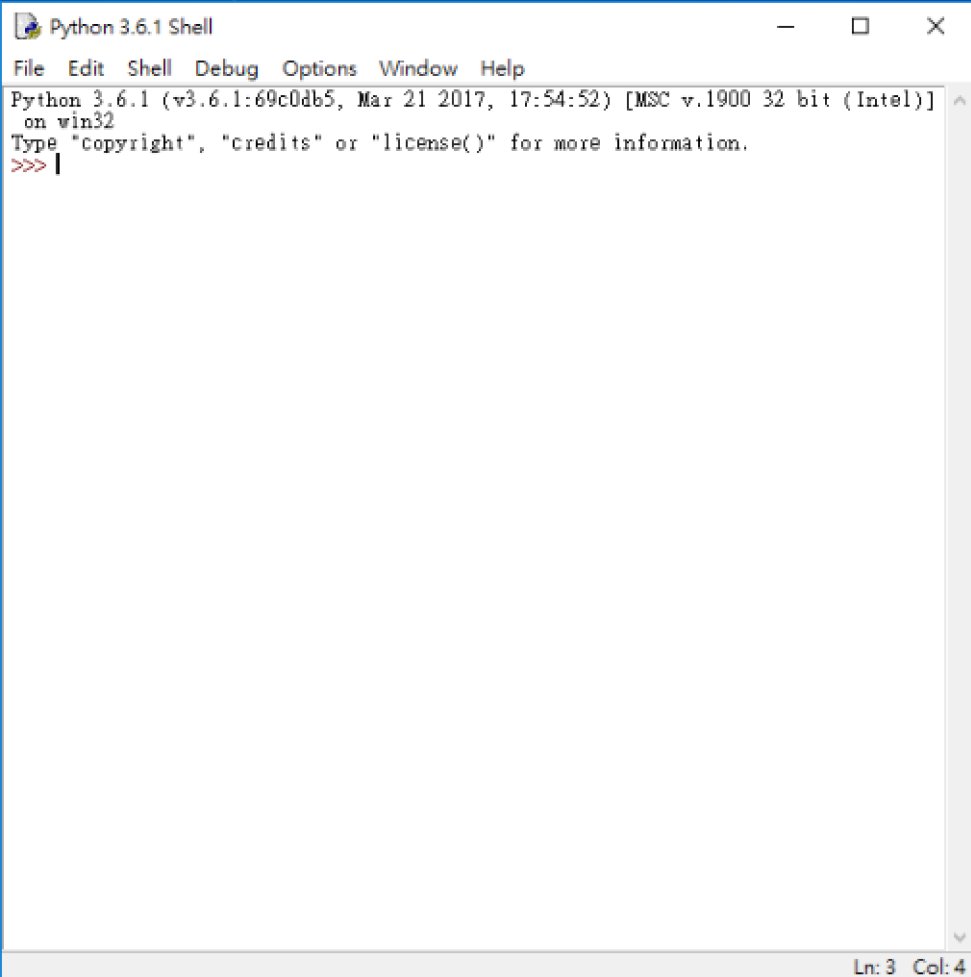
- Python語言的核心包含數字、字串、列表、字典、檔案等函式
- Python標準庫提供系統管理、網路通訊、文字處理、資料庫介面、圖形系統、XML處理等功能
 - 文字處理，包含文字格式化、正規表示式符合、文字差異計算與合併、Unicode支援，二進位資料處理等功能
 - 檔案處理，包含檔案操作、建立臨時檔案、檔案壓縮與歸檔、操作設定檔等功能
 - 作業系統功能，包含執行緒與行程支援、IO復用、日期與時間處理、呼叫系統函式、記錄檔（logging）等功能
 - 網路通訊，包含網路通訊端，SSL加密通訊、異步網路通訊等功能

- 網路協定，支援HTTP，FTP，SMTP，POP，IMAP，NNTP，XMLRPC等多種網路協定，並提供了編寫網路伺服器的框架
- W3C格式支援，包含HTML，SGML，XML的處理。
- 其它功能，包括國際化支援、數學運算、HASH、Tkinter等

Python 101

Python執行模式

- 交談式執行模式
 - 命列指示 >>>
- 程式執行模式
 - Notepad++



The screenshot shows a window titled "Python 3.6.1 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following information: "Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32", followed by "Type 'copyright', 'credits' or 'license()' for more information." and the interactive prompt ">>> |". The status bar at the bottom right indicates "Ln: 3 Col: 4".

資料型別

基本型別

- 電腦的基本工作處理資料
 - 文字資料
 - 字元(如 'H')
 - 字串(如 "Hello")
 - 數值資料
 - 整數(int)，如 123
 - 浮點數[帶有小數者](float)，如123.45、1.2345e10 (科學記號表示)
 - 複數，如 123+45j
 - 邏輯值 True/False (boolean)
- 不同型別資料佔據不同記憶體大小
- Type()函數可以用來表示資料的型別

```
>>> 3
```

```
>>> "Hello World"
```

```
>>> type(3)
```

```
>>> type("Hello World")
```

資料型別轉換

- `int()`函數將浮點數(float)轉換成整數(int)
 - `>>> int(123.45)`
 - `>>> 17/5`
 - `>>> int(17/5) # 17/5 = 3.4 (float)`
- `float()`函數將整數(int)轉換成浮點數(float)
 - `float(123)`
 - `float(17//5) # 17//5 = 3 (int)`

變數(variable)

- 程式設計時，變數可代表某個資料
 - `messag = "Jeremy Lin is a good NBA player"` (string)
 - `n = 20` (int)
 - `pi = 3.1415926535897932` (float)
- 變數如同黑盒子，內含資料，佔據一塊記憶體
 - `A = 5`
 - `A`表示變數，`5`表示資料
 - 「`=`」為「**指定**」運算子，表示將「**資料指定給變數**」。
- Python程式設計時，不需要先宣告變數的資料型別
 - 動態特性

Python 保留字(31 keywords)

and	del	from	not	while
as	elif	global	or	with
assert	else	if	pass	yield
break	except	import	print	
class	exec	in	raise	
finally	continue	is	return	
def	for	lambda	try	

運算式

運算元及運算子

- 運算式 = 運算元 + 運算子
- 資料的基本運算符號，稱之為運算子(operator)
 - 數值運算子包括： $+$, $-$, $*$, $/$, $\%$ (取餘數), $**$ (乘方), ...
- 運算過程中相關聯的資料，稱之為運算元(operand)
 - 運算元可以是常數、變數、函式
 - 如 $3+5$ ($+$ 代表運算子， $3, 5$ 代表運算元)
- 運算結果與資料的型別有相當關係
 - `>>> 5/3 = 1.666666666667 (5//3) (5.0//3)`
 - `>>> 5**2 = 25 (5.0**2)`
 - `>>> (-1)**(1/2)`

運算式

- 算術運算式
 - $5 + 3$ (簡單的運算式)
 - 求解一元二次方程式

$$ax^2 + bx + c = 0$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

- 若 $a = 1, b = 5, c = 4$ ，求 $x = ?$

比較運算子

運算子	意義	範例	結果
==	相等	5==5	True
!=	不相等	8!=5	True
>	大於	3>8	False
<	小於	5<8	True
>=	大於或等於	5>=10	False
<=	小於或等於	5<=5	True

boolean



邏輯運算子

- not 、 and 、 or 邏輯運算子

A	B	A or B	A and B	Not A	Not (A and B)
True	True	True	True	False	False
True	False	True	False	False	True
False	True	True	False	True	True
False	False	False	False	True	True

課堂練習

```
>>> 5==5
```

```
>>> 8 != 5
```

```
>>> 3>8
```

```
>>> 5<8
```

```
>>> 5>=10
```

```
>>> (5==5) and (5<5)
```

```
>>> (5==5) and (5>3)
```

```
>>> (5<3 or 5==5)
```

```
>>> not(5<3 or 5==5)
```


Python運算子優先等級

Operator	Description	Operator	Description
lambda	Lambda Expression	*, /, %	乘、除、求餘數
or	布林 OR 閘	+x, -x	正、負號
and	布林 AND	~x	位元 NOT
not x	布林 NOT	**	乘方(次方)
in, not in	成員測試	x.attribute	Attribute reference
is, is not	辨別測試	x[index]	Subscription
<, <=, >, >=, !=, ==	比較	x[index:index]	Slicing
 	位元 OR	f(arguments ...)	Function call
^	位元 XOR	(expressions, ...)	Binding or tuple display
&	位元 AND	[expressions, ...]	List display
<<, >>	位移 Shifts	{key:datum, ...}	Dictionary display
+, -	加、減	`expressions, ...`	String conversion

簡單輸入函式

- `a = input("提示字串")`
 - 輸入的資料為字串(string)，可存在變數a中

```
>>> a = input("Enter the name: ")
Enter the name: Jason
>>> a
```
- 如果輸入的資料為數值，可以經由轉換函數轉換成數值

```
>>> n = input("Enter your age: ")
Enter your age: 20
>>> n = int(n)
>>> print(n)
```

練習

- 計算BMI值
- 輸入身高(h)、體重(w)值－身高：公分，體重：公斤

$$\text{BMI} = \text{體重} / (\text{身高:公尺})^2$$

編程環境

- Notepad++
- Sublime Text 2
- Visual Studio
- Atom
- Anaconda

程式執行結構

條件流程

- 條件判斷式結果為布林值：**True**或**False**
- 條件判斷式應用**邏輯運算式**決定結果
 - If $6 \geq 3$, then True
 - If password=="hellokiky", then True or False
 - If (score>60) and (score<=90), then Print "You are a outstanding student"
- 程式執行依條件判斷式結果決定不同執行路徑

布林值

- 布林值(bool) 表示某個敘述的「對」或「錯」
- 例如：a = 5, b = 3, a > b ➔ True (a < b ➔ False)
- 例如："o" in "hello" = True, "n" in "hello" = False
- 在Python，
 - 任何非0、字串的條件式比對結果可視為True
 - if 1 ➔ True
 - 數字(0)、空字串、空值(none)的條件式比對結果視為False
 - if 0 ➔ False

簡單 if 結構

if (條件判斷式):

□□□□ <程式碼>

□□□□ <程式碼>

} True

else:

□□□□ <程式碼>

□□□□ <程式碼>

} False

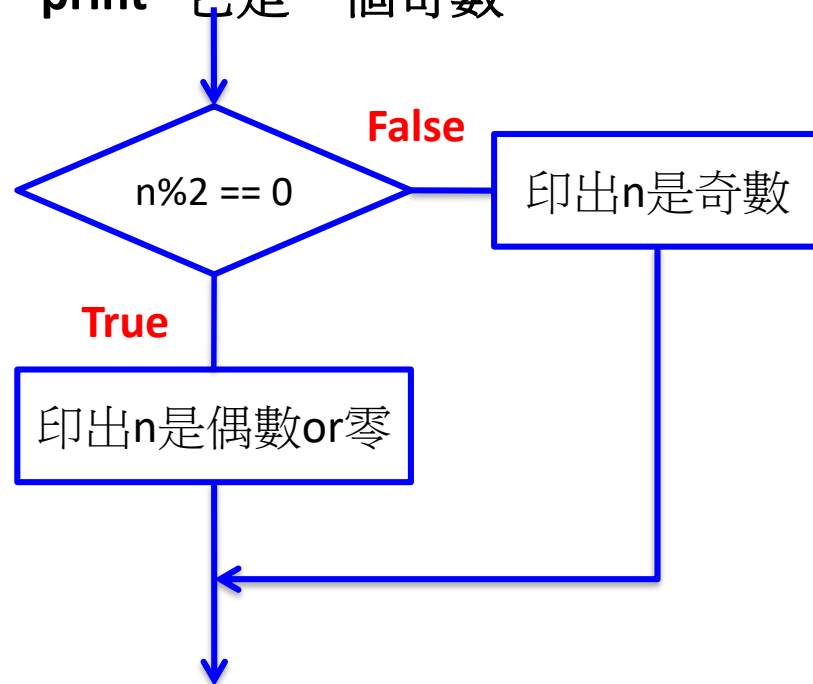
範例：

if $n \% 2 == 0$:

 print "它是一個偶數或0"

else:

 print "它是一個奇數"



鏈狀 if 結構

```
if (n == 0):
```

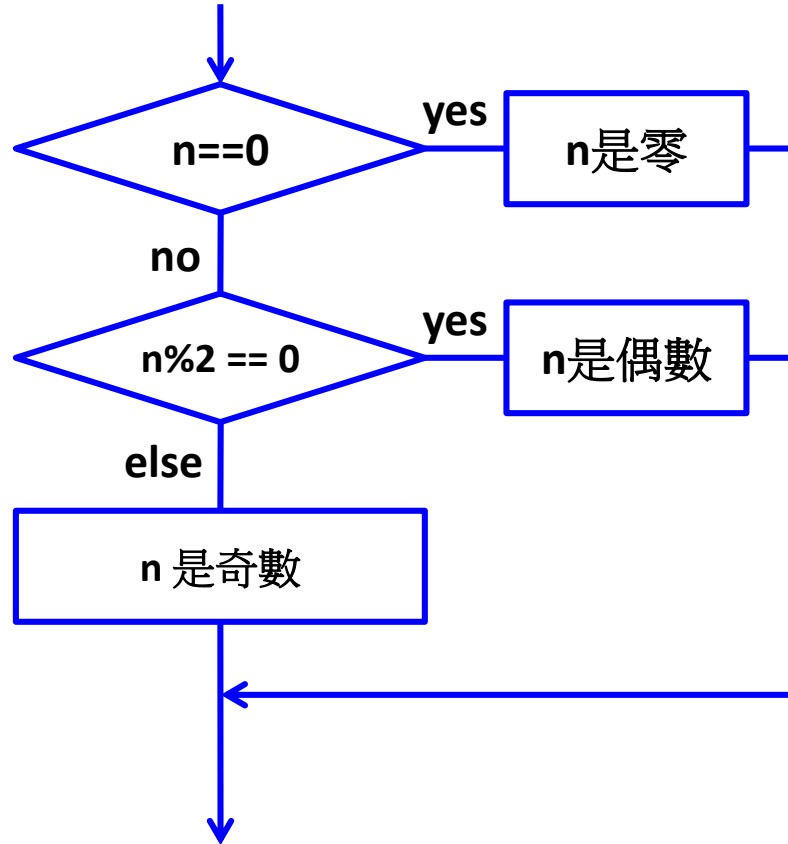
```
    print "它是0"
```

```
elif (n%2==0):
```

```
    print "它是偶數"
```

```
else:
```

```
    print "它是奇數"
```



槽狀 if 結構

```
if n%2 == 0:
```

```
    if n == 0:
```

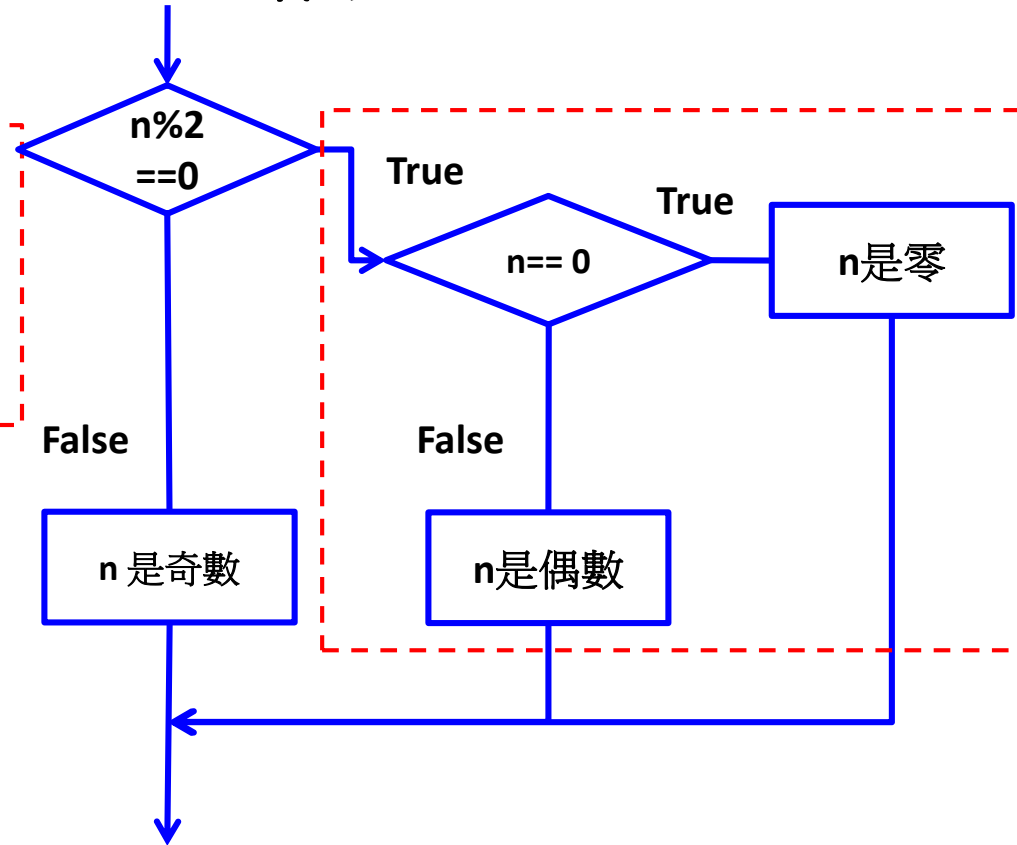
```
        print "它是0"
```

```
    else:
```

```
        print "它是一個偶數"
```

```
else:
```

```
    print "它是一個奇數"
```

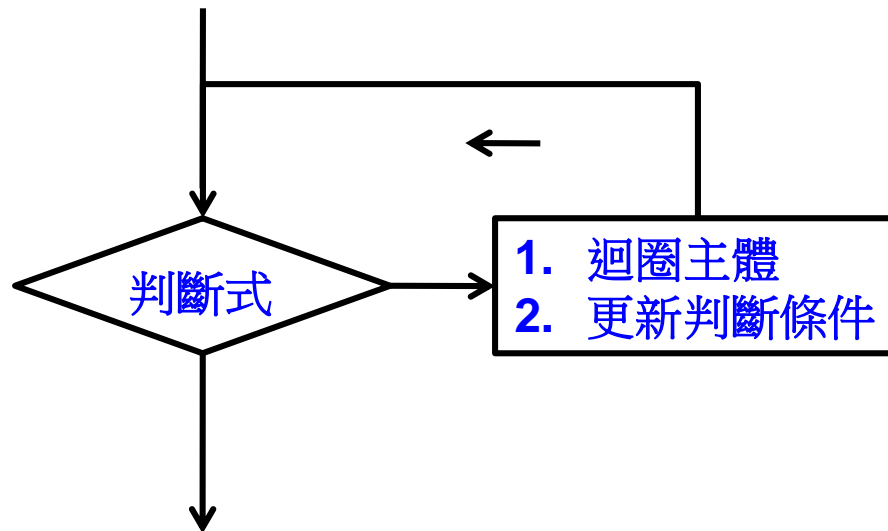


練習

- 請用input函數輸入a, b, c，計算 $d = b^2 - 4ac$
 - if $d < 0$, then 無解
 - if $d = 0$, then 求唯一解 x
 - if $d > 0$, then x 有二解

重複流程

- while 迴圈
- for 迴圈



For 迴圈

```
for i in [序列]
```

```
    <程式碼>
```

```
    <程式碼>
```

- for迴圈根據某一序列(sequence)重覆執行程式
- i 是指序列中的物件

Sequence型別

- 序列(Sequence)是一種資料型別
 - 表示在一容器中有次序的一組資料
 - 可能是一個字串(string)：“hello kiki”
 - 可能是一個序列(list)：[1, 2, 3, 4, 5] (或可稱為Array)
 - 可能是一個雜亂型別的序列容器：[“Love”, 123, 3.14, True]
 - 每一筆資料在序列(Sequence)中都有一個位置指標，表示該資料在序列中的位置
 - range()函數可以產生一組「整數」序列

range()函數

- range()函數產生一組有次序的整數序列
 - range(10) ➔ [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
 - range(1, 10, 3) ➔ [1, 4, 7]
 - range(5, 0, -1) ➔ [5, 4, 3, 2, 1]
- 利用for 迴圈及range()函數可將走訪容器(container)中的每一個資料值
- 範例
 - box = ["Love", 123, 3.14, True]
 - for *i* in range(4):
 - print box[*i*]

範例

- 求 $1+2+3\ldots+100$ 的和

```
s = 0
```

```
for i in range(1,100):
```

```
    s = s + i
```

```
print "sum of 1 to 100 = " + s
```


While迴圈

while (條件判斷式):

□□□□ <程式碼>

□□□□ <程式碼>

- 執行while迴圈內的程式，必須先通過條件測試(條件判斷式)
- 條件式後面的**冒號(:)**，迴圈內的程式碼須要**內縮**
- 每一次執行程式前，須先檢查**條件判斷式**，
 - 若測試結果為False，則不會執行**迴圈內的程式**。
 - 若測試結果為True，則執行**迴圈內的程式**。
- 如此達成重複執行目的。

範例

- 求 $1+2+3\ldots+100$ 的和

$n=1$

$s=0$

While ($n \leq 100$):

$s = s + n$

$n = n + 1$

Print “ $1+2+3+\ldots+100=$ “, s

範例

- 判斷某一數 n 是否為「質數」？
 - 質數除了 1 及本身沒有其他因數。

continue指令

- Continue 指令可以避開程式執行的步驟，直接跳到迴圈下一回合
- 印出1..100數字，每5個數字一列

1 2 3 4 5

6 7 8 9 10

11 12 13 14 15

16 17 18 19 20

.....

86 87 88 89 90

91 92 93 94 95

96 97 98 99 100

```
for i in range(100):  
    print i+1,  
    if (i+1)%5 == 0  
        print  
        continue
```

break指令

- break指令可以**強制**中斷迴圈(while , for)的執行
- 試試看以下程式碼

While 1:

print “我愛打LOL” , n

n += 1

— 有無方法使程式停止？

- 讓break指令強迫停止迴圈的執行

While 1:

print “無窮迴圈結果” , n

n+=1

if n > 10000:

break

練習

- 請寫一個python程式找出100至999間的質數？

資料結構

Python資料結構

- List
- Tuple
- Dictionary
- String

List (列表)

- List是一種有序的資料除存容器
 - $A = [1, 2, 3, 4, 5]$
 - $B = [\text{"boy"}, \text{"girl"}, \text{"friend"}, \text{"sister"}, \text{"brother"}]$
- List每一個資料都有一參考指標(index)
 - $A[1] = 2$
 - $B[3] = \text{"sister"}$
- List中的資料型態可以不必一樣
 - $y = [1, 2, \text{"hello"}]$
- `[]` 代表空列表(list)

二維list

- List 可以包含另一個List
 - `C = [[1, 2], ['a', 'b', 'c'], [3.14]]`
 - `>>> C[1] = ['a', 'b', 'c']`
 - `>>> C[0][1] = 2`
 - `>>> C[1][1] = 'a'`
 - `>>> C[2][1] = ???`

- List的合併
 - $X = [1, 3, 5]$
 - $Y = [2, 4, 6]$
 - $Z = X + Y + [7, 8, 9]$
- List的子集
 - $Z = [1, 2, 3, 4, 5, 6, 7, 8, 9]$
 - $Z[-1] = 9, Z[0] = 1$
 - $Z[2:5] = [3, 4, 5]$ (不包含 $z[5]$)
 - $Z[3:] = [4, 5, 6, 7, 8, 9]$
 - $Z[\text{len}(z)] = ?$
- List複製
 - $Y = Z[:]$

List函數

- `len()` 可得到List的長度
 - `X = [1,2,3,4,5]`
 - `len(X)=5`
- `del()`可刪除List的元素
 - `del(X[1])`

List方法

- **append**
 - `X.append(99)`
- **extend**
 - `X.extend(100)`
- **insert**
 - `X.insert(3, 3)`
- **pop**
 - `X.pop()`
- **remove**
 - `X.remove(3)`
- **count**
 - `X.count(5)`
- **index**
 - `X.index(3)`
- **reverse**
 - `X.reverse()`
- **sort**
 - `X.sort()`

List 走訪

- `n = [1, 2, 3, 4, 5]`
- `for e in n:`
 `print e,`
- `for i in range(len(n)):`
 `print n[i],`

Tuple

- Tuple是一種immutable的資料容器(list)
 - `X = (1, 2, 3)`
 - `X[1]=???`
 - `X[1]=5` 不允許發生
- 內建函式`list(X)`可以將tuple轉換成list
 - `T = (1, 2, 3, 4, 5)`
 - `list(T) ➔ [1, 2, 3, 4, 5]`

字串

- 在Python中**字串是一個物件**，包括屬性、方法
- 物件方法的呼叫方式為**物件.方法**
 - Eg., "abc".upper() ➔ ABC
 - 方法執行結果產生新的字串，原字串不變

有用的字串方法

- `upper()`：將字串中所有字母轉換為大寫
- `lower()`：將字串中所有字母轉換為小寫
- `swapcase()`：將字串中所有字母大小互換
- `capitalize()`：將字串的第一個字母大寫
- `title()`：將字串中每一個字的第一個字母大寫
- `strip()`：將字串中的頭尾空白去除
- `replace()`：代換字串中的子字串
- `find()`：尋找字串中的子字串的位置
- `count()`：計算某一字在自串中出現的次數
- `split()`：將字串轉換為 `list`

strip()

- 將字串中的頭尾空白去除
 - `x = " This is a happy day. \t\t"`
 - `x.strip()`
 - `'This is a happy day.'`
- `lstrip()`：去除字串左側的空白
- `rstrip()`：去除字串右側的空白

replace()

- 代換字串中的子字串
 - `x = " This is a happy day. \t\t "`
 - `x.replace("happy", "rainy")`
 - `' This is a rainy day. \t\t '`

find

- 尋找字串中的子字串的位置
 - `x = " This is a happy day. \t\t "`
 - `x.find("happy")`
13
- `rfind`尋找字串中的子字串的最後位置
- `index`及`rindex`
- `count`
- `startswith`及`endswith`

split

- Str = “This is good day.”
- Str.split()
- ['this', 'is', 'a', 'good', 'day']

轉換函數

- 內建函數`repr()`可將任何資料轉換成字串
 - `repr(123)`
 - `repr("abc")`
 - `repr([1,2,3])`
- 將字串轉換成數字
 - `float("123.45")`
 - `int("12345")`
 - `int("1000",8)`
 - `int("ff",16)`

列印格式字元

- Print(I am **%s**, and my age is **%d** years old" **%**
(Obama, 50))
 - %c : 印出字元
 - %s : 印出字串
 - %d : 印出整數
 - %f : 印出小數
 - %e : 印出科學符號數
 - %o : 印出八進位數
 - %u : 印出萬國碼
 - %x : 印出**16**進位數

模組

Modules

- 模組(modules)是由常數變數、函式所組成
- 使用模組前必須先呼叫(import)
- 如有使用數學模組
 - `import math`
 - `math.pi` 是一個常數變數，代表圓周率
 - `math.sqrt()` 代表數學的開根號
- 可以用 `help(math)` 來看看 `math` 模組中有哪些函式

範例

- 計算一個圓的週長及面積

```
>>> import math
```

```
>>> radius = 5
```

```
>>> P = 2*math.pi*radius
```

```
>>> A = math.pi*radius**2
```

- 如果不想每次都要鍵入 math

```
>>> from math import pi, sqrt
```

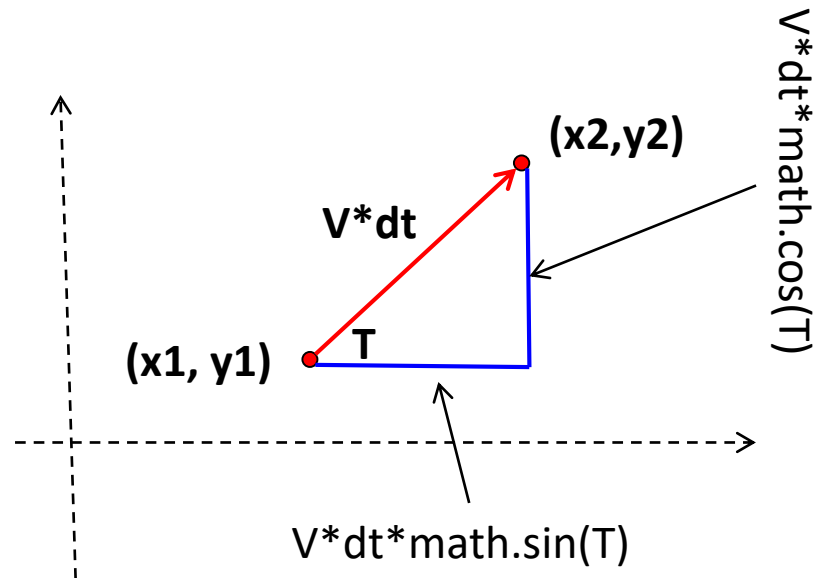
```
>>> radius = 5
```

```
>>> P = 2*pi*radius
```

```
>>> A = pi*radius**2
```

範例

- 已知物體運動方向夾角(T)、運動速度(V)及啟始位置($x1, y1$)，計算物體下一點位置
 - $x2 = x1 + V*dt*\sin(T)$
 - $y2 = y1 + V*dt*\cos(T)$
 - dt 代表經過時間
 - T 單位為經度量
- Python code
 - `import math`
 - $x2 = x1 + V*dt*\mathit{math.sin}(T)$
 - $y2 = y1 + V*dt*\mathit{math.cos}(T)$



亂數模組

- `import random`
 - `randint(1, 100)` 產生一個1..100之間的隨機整數
 - `random()` 產生一個 0..1 之間的隨機數
 - `shuffle([])` 模擬洗牌
 - `choice([...])` 從 [...]的物件中隨機取出一個物件

時間模組

- `import time`
 - `time()`
 - `time.localtime(time.time())`
 - `time.asctime(time.localtime(time.time()))`
 - `clock()`
 - `sleep(secs)`

系統模組

- `import sys`
 - `sys.exit(0)`
 - `sys.args[0]`

OS模組

- `import os`
- 改變路徑
 - `os.chdir`
- 目前目錄
 - `cwd = os.getcwd()`
- 顯示memo.txt的絕對路徑
 - `os.path.abspath('memo.txt')`
- 檢查memo.txt是否在目前目錄下
 - `os.path.exists('memo.txt')`
- 檢查memo.txt是否為一目錄
 - `os.path.isdir('memo.txt')`
 - `os.path.isdir('music')`
- 列出目錄檔案
 - `os.listdir(cwd)`
- 結合dir及name成為一路徑
 - `os.path.join(dir, name)`

自訂函式

Python 函式

- 在程式設計中有一些特定的程式碼將會重複的被執行，這些程式碼可以分別出來，設計一個可重覆使用的機制，我們稱它為副程式或函式。
 - 副程式(Subroutine)是一段執行某些任務的程式碼，沒有回傳值，僅負責執行某一特定工作。
 - 函式(Function)，如同數學函數一樣 $y = f(x)$ ，具有輸入參數(x)及輸出結果(y)
- 副程式或函式的功用程式設計中
 - 使程式模組化(將複雜的問題分為需多小的問題解決)
 - 可再使用
- 二種函數類型
 - 程式語言所提出的內建函數或函數庫，如數值與字串轉換函數...
 - 使用者自行開發的函數

定義程式函數

- 函式定義規則
 - 使用「def」關鍵字
 - 一個適當的函數名稱
 - 名稱後要接一組「小括號」
 - 小括號中是「參數」
 - 小括號後一定要有「：」

- 範例1

#將兩個成本相加

cost1 = 15

cost2 = 20

def sumCart():

totalCost = cost1 + cost2

print totalCost

sumCart()

函數參數

- 執行函數所需的參數(Arguments)必須放在小括號內，如有一個以上的參數可用「，」隔開
- 範例二：

cost1 = 15

cost2 = 20

cost3 = 5

cost4 = 10

```
def sumCart(item1, item2):
```

```
    totalCost = item1 + item2
```

```
    print totalCost
```

```
sumCart(cost1, cost2)
```

- 參數是區域性的變數，在函數外部參數沒有作用的
- 執行函數時，傳入的參數個數必須與定義的參數個數一致，除非定義函數時，參數有設定預設值
- 範例三：

cost1 = 15

cost2 = 20

cost3 = 5

cost4 = 10

def sumCart(item1, item2 = 5):

totalCost = item1 + item2

print totalCost

sumCart(cost1)

sumCart(cost3, cost4)

函數回傳值

- 利用「**return**」關鍵字，回傳函數執行結果
- 範例4

cost1 = 15

cost2 = 20

cost3 = 5

cost4 = 10

def sumCart(item1, item2):

totalCost = item1 + item2

return totalCost

cart1 = sumCart(cost1, cost2)

cart2 = sumCart(cost3, cost4)

print cart1

print cart2

函數中變數影響範圍

```
j, k = 1, 2
```

```
def proc1():  
    j, k = 3, 4  
    print "j == %d and k == %d" % (j, k)  
    k = 5
```

```
def proc2():  
    j = 6  
    proc1()  
    print "j == %d and k == %d" % (j, k)
```

```
k = 7  
proc1()  
print "j == %d and k == %d" % (j, k)
```

```
j = 8  
proc2()  
print "j == %d and k == %d" % (j, k)
```

遞迴函式

- def factorial(n):
 if n == 0 or n == 1: # $0! = 1! = 1$
 return 1
 else:
 return (n * factorial(n-1))

全域及區域

```
x = 1
def fun(a):
    b=3
    x=4
    def sub(c):
        d=b
        global x
        x = 7
        print "\nNested Function\n"
        print locals()

    sub(5)
    print "\nFunction\n"
    print locals()
    print locals()["x"]
    print globals()["x"]

print "\nGlobals\n"
print globals()

fun(2)
```


Lambda 及 Apply函數

- `lambda x: x**2+2*x+4`

- `g=`**lambda** `x: x**2+2*x+4`
- `g(5)`

- `lambda x,y: x**2+y**2`

- `g=`**lambda** `x,y: x**2+y**2`
- `g(5,8)`

- 常與**filter**, **map**, **reduce**一同使用

- `foo = [2, 18, 9, 22, 17, 24, 8, 12, 27]`
- `print` **filter**`(lambda x: x % 3 == 0, foo)`
- `print` **map**`(lambda x: x * 2 + 10, foo)`
- `print` **reduce**`(lambda x, y: x + y, foo)`

```
def foo(x, y):
```

```
    return x**2+y**2
```

```
G = apply(foo, (5,4))
```

檔案處理

有用的字串方法

- `upper()`：將字串中所有字母轉換為大寫
- `lower()`：將字串中所有字母轉換為小寫
- `swapcase()`：將字串中所有字母大小互換
- `capitalize()`：將字串的第一個字母大寫
- `title()`：將字串中每一個字的第一個字母大寫
- `strip()`：將字串中的頭尾空白去除
- `replace()`：代換字串中的子字串
- `find()`：尋找字串中的子字串的位置
- `count()`：計算某一字在串中出現的次數
- `split()`：將字串轉換為 `list`

計算字元頻率

- `Str = ""`
- Please note that CNN Student News is designed for use in middle and high school classrooms. It's always a good idea for you to preview each program before showing it to students. [Click here](#) to learn more about how to use Student News in your classroom.
- `""`
- `str = str.replace('.', '')`
- `strlow = str.lower()`
- `for i in range(26):`
 - `count.append(0)`
- `for c in str:`
 - `if ord(c) in range(97, 123):`
 - `count[ord(c)-97] = count[ord(c)-97]+1`
- `for i in range(26):`
 - `print chr(i+97), count[i]`

計算字出現頻率

- `str = """ Please note that CNN Student News is designed for use in middle and high school classrooms. It's always a good idea for you to preview each program before showing it to students. Click here to learn more about how to use Student News in your classroom. """`
- `str = str.replace(".", "")`
- `str = str.replace("\n", "")`
- `str = str.replace("'s", "")`
- `str = str.lower()`
- `strlist = str.split(" ")`
- `counts = {}`
- `for word in strlist:`
 - `counts[word] = str.count(word)`
- `counts[' a '] = str.count(' a ')`
- `del counts['a']`
- `sortedcounts = sorted(counts.iterkeys())`
- `sortedcounts = sorted(counts.iteritems(), key=lambda (k,v): (v,k))`
- `for key, value in sortedcounts:`
 - `print key, value`