程式設計概論
Programming 101
—函式(function)

授課老師：邱淑怡

Date:4/1/2021

Outline

- 函式(Function): 自訂函式
  - 優點
- Global variables and local variables
- 遞迴函式(Recursive function)
- lambda function

Function

- function 是一個建構程式時的小區塊，它就像是一台機器，你可以自行指定它的功能，以及所需要的原料（輸入）、產出（輸出）
- Example: 自動販賣機就像是一個 function ，他的 input 是硬幣和商品的選擇， output 則是你所選的商品

Function 優點

1. 程式的重複利用性
2. 程式的易讀性
3. 程式的易除錯性
4. 程式的一致性
5. 程式的模組化：模組化是一個在撰寫程式時的概念
   - 將一個完整的程式視為蓋一棟房子，那麼函式就像是房子的鋼筋、水泥、磚塊這樣的東西
   - 程式是由許多個函式以及其他東西所建構出來的，而且函式與函式之間的分工十分明確，
   - 每一個函式有它自己所負責的東西，並且可以獨立於這個程式

如何發展出function

- 當發現到兩個程式片段極為類似，只有當中幾個計算用到的數值或變數不同時

```
# get max. value
x=0
if a>b:
        x=a
else:
        x=b
```

可以使用函式來封裝程式片段，將流程中引用不同數值或變數的部份設計為參數

```
def max1(a, b):
    x=0
    if a>b:
        x=a
    else:
        x=b
    return x
```

區域變數 (local variable)

Function 定義及呼叫(call)

```
def 函式名稱(參數1,參數2,...):
    程式碼(statements)
    [return value]
```

如何呼叫function?

函式名稱(引數1,引數2)

1.def 是定義我們的function，包含函式名稱及這個函式所需要的參數(parameter)
2.函式名稱與變數名稱規定一致
3.Return value 是指function傳回值，可傳回0, 1,或多個值，也可以省略不寫
4. 函式呼叫中的數叫引數(argument)

Function 實例1(沒有傳回值): 判斷閏年

定義函式(function)

```
def check_leap(year):
    if(year%400==0) or (year%100!=0 and year%4==0):
        print("leap year")
    else:
        print("common year")
```

呼叫函式(function)

```
year= int(input("input year:"))
check_leap(year)
```

練習題

1. 撰寫一個function(CtoF1)，它可以將攝氏溫度轉換於華氏溫度(F=C*1.8+32)，並把華氏溫度印出來
2. 呼叫CtoF1 function並傳遞攝氏溫度作為參數後印出華氏溫度

Global variable vs. local variable

Global variable

```
X='global'
def f1():
    print('X inside:', X)

f1()
print('X outside:',X)
```

Local variable

```
def f1():
    y='local'
    print('y inside:',y)

f1()
print('y outside',y)
```

How to use global variable and local variable with same name?

```
x=5
def f1():
    x=10
    print('X inside:', x)

f1()
print('X outside:',x)
```

Outputs: local x: 10

Outputs: global x: 10

Solve the problem: use global variable and local variable in the same code?

```
x=5
def f1():
    global x
    x=10
    print('X inside:', x)

f1()
print('X outside:',x)
```

Function 實例2(有傳回值): 判斷閏年

定義函式(function)

```
def check_leap(year):
    if(year%400==0) or (year%100!=0 and year%4==0):
        return “leap year”
    else:
        return “common year”
```

呼叫函式(function)

```
year= int(input("input year:"))
result=check_leap(year)
print(result)
```

練習題

1. 撰寫一個function(CtoF2)，它可以將攝氏溫度轉換於華氏溫度(F=C*1.8+32)
2. 呼叫CtoF2 function並傳遞攝氏溫度作為參數，並將華氏溫度當作回傳值
3. 最後，把華氏溫度印出來

Function 實例3(有傳回值): 計算x的y次方

Define function

```
def power(x, y):
    r=1
    while y>0:
        r=r*x
        y=y-1
    return r
```

x,y 稱作參數；依據位置對應

3,5 稱作引數

Call function

```
Res= power(3,5)
```

```
Res =power(3)
```

Function實例3: 參數有預設值

Define function

```
def power(x, y=2):
    r=1
    while y>0:
        r=r*x
        y=y-1
    return r
```

沒有預設值的參數一定要擺在前面

Call function

```
Res= power(3,5)
```

=

```
Res= power(y=5,x=3)
```

```
Res =power(3)
```

遞迴函式(recursive function)

遞迴函式(recursive function)

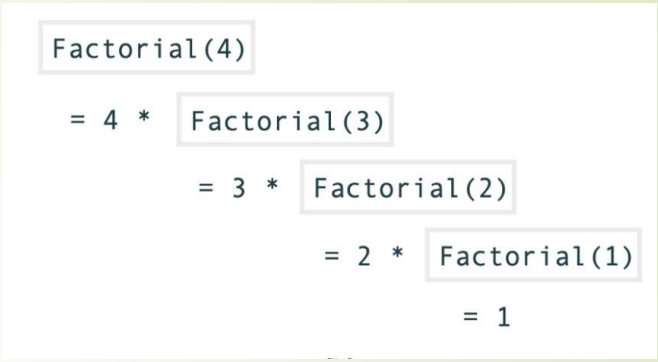
- 在解決問題時，我們經常會遇到較難使用 loop 或是 if statement就能處理的問題，像是走迷宮問題遇到死路時需要回到上一層計算、或是在解決河內塔問題時儘管操作相同，卻因為每次要進行操作的參數不同而需要寫重複的程式碼等等
- 遞迴的基本概念
  - 解決一個問題的方法是將其拆解，再各自將小問題解決以後得到答案，這樣的概念我們稱之為 “Divide and Conquer” (分治法)
  - 遞迴(recursion)這個方法就是依據此概念形成
- 當一個函式會在執行當中，會不斷地自己**呼叫自己**時，我們便認為這個函式具有遞迴的性質
- 為避免函式永無止盡地自我呼叫 (self-calling)，我們也需要設計一個明確的**終止條件**

遞迴函式(recursive function)

- 設計一個遞迴函式的兩個重點
  - 遞迴自我呼叫的方式
  - 結束呼叫的終止條件
- 遞迴函式通常可以被for或while迴圈取代，但由於遞迴函式邏輯性、可讀性及彈性均比迴圈來得好

遞迴函式_實例: 計算階乘(4!)

- 當n=0時，F(n)=n!=0!=1
- 當n>0時，F(n)=n!=n*(n-1)!=n*F(n-1)
- 當n<0時，F(n)=-1 表示無法計算階乘



define recursive function

```
def factorial(n):
    if n==0 or n==1:
        return 1
    else:
        return n* factorial(n-1)
```

Call function

```
Print("0!=", factorial(0))
Print("4!=", factorial(4))
```

一般函式_實例: 計算階乘

define function

```
def factorial_loop (n):
    factor = 1
    for I in range(1,n+1):
        factor *= I
    return factor
```

Call function

```
print("4!=", factorial_loop(4))
```

lambda function

Function 另一種方式:用**lambda**函式來定義函式，執行運算式時將會產生函式物件

- lambda用來建立小的匿名函式(沒有函式名稱)
- lambda不是函式名稱，而是指令
- lambda的語法是(只有一行)：
  lambda arg1, arg2, ... : expression
  - lambda中arg1、arg2等就相當於定義函式時的參數，之後你可以在expression中使用這些參數。
  - lambda是運算式，不是陳述句，在:之後的也必須是運算式，lambda中也不能有區塊
- 這表示一些小的運算任務你可以使用lambda，而較複雜的邏輯你可以使用def來定義。

lambda函式實例: find_max()函式

- Define lambda function
  - f = lambda num1, num2: num1 if num1 > num2 else num2

Python code

- Call lambda function
  - f(5,9)
  - f(-5, -9)
  - f(num2=5, num1=-9)

lambda函式練習題

- 將lambda運算式所產生的匿名函式指派給變數Add，這個匿名函式會傳回參數x和參數y相加的結果
- 接著，呼叫該lambda函式計算
  1. 1+2
  2. 50+(-100)
  3. “abc”+”de”

lambda函式特性

- lambda 參數1, 參數2, ... : 運算式A if 關係運算式 else 運算式B
  - lambda函式的語法中沒有函式名稱
  - 語法最前面的lambda不是函式名稱，而是指令
  - lambda函式的特性就是「用過即丟」
  - lambda函式最典型的用法就是搭配filter()和map()這二個內建函式，對資料組進行處理。需要的話，我們也可以把lambda函式設定給一個物件，如此一來，lambda函式就可以當成一般函式使用。

Lambda函式實例: 讓lambda函式傳回list資料，參數有給定預設值

題目：產生一個資料串列(list)起始點從5開始，結束點預設值10

Python code

- start =5 #全域變數(global variable)
- f = lambda stop = 10: [i for i in range(start, stop)]
  - 建立一個lambda函式，幫參數stop設定預設值
  - 這個lambda函式會傳回一個list資料組
- 利用f呼叫lambda函式:
  - f() # [5, 6, 7, 8, 9]
  - f(12) # [5, 6, 7, 8, 9, 10, 11]

介紹filter() function

- filter, map, reduce 都是針對集合物件處理的特殊函式，可有助於python的資料處理以及程式簡化
- filter函式過濾序列，過濾不符合條件的元素
- Format: filter(function, sequence)
  - 以傳入的boolean function作為條件函式，重複所有的sequence的元素並收集 function(元素) 為True的元素到一個list

```
def is_odd(n):
    if n%2 ==1
        return n
```

```
def is_odd(n):
    return n % 2 == 1

tmplist = filter(is_odd, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
newlist = list(tmplist)
print(newlist)
```

lambda函式搭配filter()

- filter()函式是用來從list中篩選出想要的資料，它需要二個參數，第一個是函式，第二個是list
- list中的每一項資料，都會傳給該函式檢查，當函式執行結果為True的時候，該項資料才會被挑選
- 利用filter()函式和lambda函式

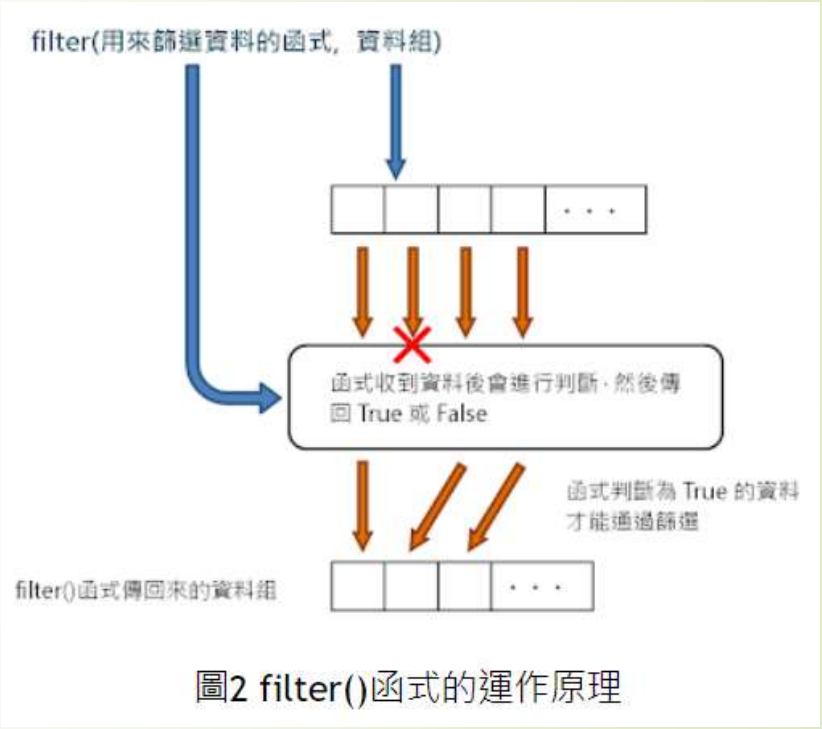


圖2 filter()函式的運作原理

https://sites.google.com/site/ezpythoncolorcourse/lambdafunction

lambda函式搭配filter()實例

- 題目：從學生成績資料組中，挑出不及格的成績

Python code

```
scores = [90, 50, 80, 40, 100]
f = lambda x: True if x < 60 else False
# 呼叫filter()函式，傳入篩選函式和list
fail_scores = list(filter(f, scores))
print(fail_scores)
```

=

```
scores = [90, 50, 80, 40, 100]
fail_scores = list(filter(lambda x: True if x < 60 else False, scores))
print(fail_scores)
```

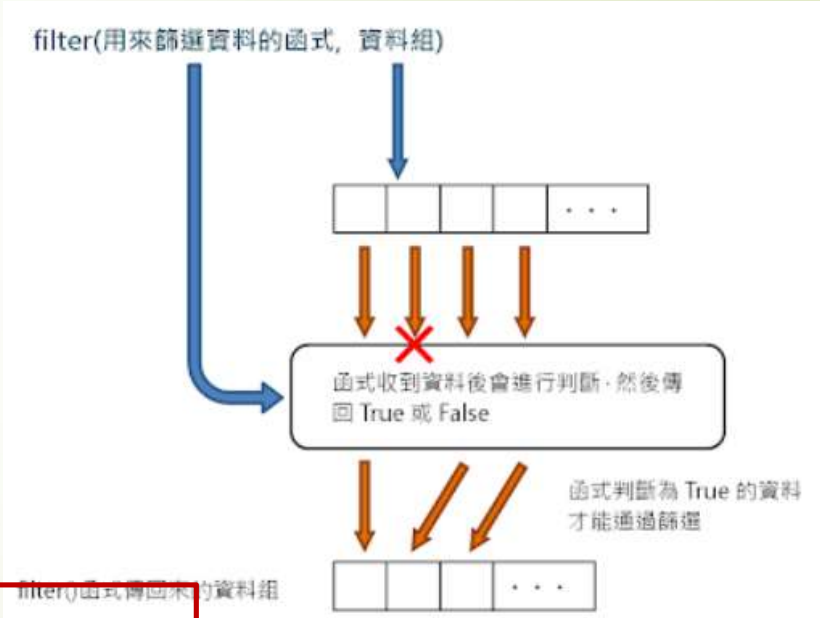


圖2 filter()函式的運作原理

lambda函式搭配filter()的練習題

- `x1 = ['Justin', 'caterpillar', 'openhome' ]`
  1. 顯示x1內元素的字串長度(len)>6
  2. 顯示x1內元素的字串內容有'i'的字串(in)

lambda運算式 v.s. function

- 在 Python 中，可以使用 lambda 表示式來定義一個函式

該函式稱為 λ 函式或是匿名函式 (Anonymous function)

```
lambda a, b : a if a>b else b
```

lambda運算式不能有程式區塊

一般的函式

```
def max1(a, b):
    return a if a>b else b
```

課後練習

- 何謂"河內塔(Tower of Hanoi)問題"?
  - https://jason-chen-1992.weebly.com/home/-tower-of-hanoi
- Python code: https://scipython.com/book/chapter-2-the-core-python-language-i/examples/the-tower-of-hanoi/