1 基本程式結構

1.1 判斷

In [ ]:
```
score = 90
if (score >= 60):
    print("及格")
    print('恭喜!')
else:
    print("不及格")
    print('加油!')
```

In [ ]:
```
score = int(  input("Enter your score:")  )
```

In [ ]:
```
if (score >=60):
    print("及格")
    print('恭喜!')
else:
    print("不及格")
    print('加油!')
```

In [ ]:
```
score = 90
if (score >=80):
    print("A")
elif (score >= 60):
    print("B")
else:
    print("不及格")
```

In [ ]:
```

```

1.2 迴圈

1.2.1 for loop

In [ ]:
```
for i in range(1,6):
    print (i)
```

In [ ]:
```
for i in range(1,11,2):
    print (i)
```

In [ ]:
```
for i in ( 7, 3, 8, 1, 4 ):
    print(i)
```

In [ ]:
```
names =['Albert','Marry']
for n in names:
    print(n);
```

In [ ]:
```
a =[10,20,30]
for i in a:
    print(i);
```

In [ ]:
```
for i in range(0, 6):
    print(" "*(5-i)+"* "*(i))
```

1.2.2 while

In [ ]:
```
k=1
while k <=5:
    print(k)
    k=k+1
```

In [ ]:
```
#
k=1
while k <=5 :
    if k==7:
        k+=1
        continue
    print (k)
    k=k+1
```

In [ ]:
```

```

1.3 User Input使用者輸入

In [ ]:
```
name = input('Enter your name:')  #若是輸入字串需用引號
print('Hi,',name)
```

In [ ]:
```
n = input('Enter an integer:')
n = int(n)
print(n**2)
```

In [ ]:
```

```

2 如何定義函數function 方法method

In [ ]:
```
def larger(a,b):
    if a>b:
        return a
    else:
        return b
```

In [ ]:
```
print (larger(5,6))
print larger("abc","abd")
```

In [ ]:
```
def plus10(a,b):
    a+=10
    b+=10
    return a, b
```

In [ ]:
```
print plus10(2,5)
x,y= plus10(1,6)
print x, y
```

In [ ]:

2.1 lambda

實作出很簡單的function(只處理一個運算式)。

lambda param1, param2, ... : expression

等於以下的函數功能

def fun(param1, param2, ...) : return expression

In [ ]:
```
func1 = lambda x,y : x+y
func1(2,3)
```

In [ ]:

In [ ]:

2.2 file read and write

In [ ]:
```
#寫資料至檔案
file = open('mydata.txt', 'w')
file.write('資管系')
file.close()
```

In [ ]:
```
#讀取檔案：
file = open('mydata.txt', 'r')
for line in file.readlines():
    print(line)
file.close()
```

In [ ]:
```
matter='''this is my data
this is my data
this is my data
'''
```

In [ ]:
```
f = open('aboutbook.txt', 'w')
f.write(matter)
f.close()
```

In [ ]:
```
f = open('aboutbook.txt')
while True:
    line = f.readline()
    if len(line) == 0:
        break
    print (line)
f.close()
```

In [ ]:

In [ ]:

3 方法中區域變數與全域變數 的存取

In [ ]:
```
x = 10
def func():
    print(x) ## 這行 x 是global 可以讀取 不可以變更(基於安全理由)
func()
```

In [ ]:
```
x = 10
def func():
    x = 20
func()
```

In [ ]:
```
x = 10
def func():
    x = 20 #這行是區域變數
    print("區域變數x",x)
func()
print(x)
```

In [ ]:
```
x = 10
def func():
    x = 20
    print(x)
func()
```

In [ ]:
```
x = 10
def some():
    print(x) # 這行OK x 是global 可以讀取

    #x = 20   # 這行 不OK 因為上一行的 x 是global， 這一行會混淆! local variable 'x' referenced before assignment

some()
```

In [ ]:
```
x = 10
def func():
    global x
    print(x) # 這行 x 是global 可以讀取
    x = 20   # 這行 x 是global 可以變更
    print(x)
func()

print(x)
```

In [ ]:
```

```

In [ ]:
```
globvar = 0

def set_globvar_to_one():
    #global globvar    # Needed to modify global copy of globvar
    globvar = 1

def print_globvar():
    print(globvar)     # No need for global declaration to read value of globvar

set_globvar_to_one()
print_globvar()       # Prints 1
```

In [ ]:
```

```

3.1 Applying Functions to Sequences

In [ ]:
```
def evenval(x):
    return x % 2 ==0
```

#使用filter

filter(function, sequence)，按照所定义的函数过滤掉列表中的一些元素，function必须返回布尔值

In [ ]:
```
x=range(1,11)
evens=filter(evenval, x)
```

In [ ]:
```
evens
```

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

#使用map

map(function, sequence)：将一个列表映射到另一个列表

In [ ]:
```
x=range(1,11)
evens =map(evenval, x)
```

In [ ]:
```
evens
```

In [ ]:
```
def square(x):
    return x*x
```

In [ ]:
```
#使用map
sqr=map(square, range(1, 11))
print(list(sqr))
```

In [ ]:
```

```

3.2 [x for item in sequence < if (conditions) >]

In [ ]:
```
x=[1,2,3,4,5,6,7]
[i**2 for i in x if i<= 5]
```

3.3 Iterating Over Multiple Sequences Simultaneously

In [ ]:
```
headers = ['name', 'shares', 'price']
values = ['ACME', 100, 490.1]
```

In [ ]:
```
for name, val in zip(headers, values):
    print(name, '=', val)
```

In [ ]:
```
s = zip(headers,values)
for name, val in s:
    print(name, '=', val)

#s1 = dict(zip(headers,values))  #Dictionary不能像上面這樣操作
```

In [ ]:

In [ ]:

4 Exercise: find primes 練習: 印出質數

In [ ]:
```
x = 4

isPrime=True
if x==0 or x==1:
    isPrime=False
for divisor in range(2, x):
    if x%divisor == 0:
        isPrime=False
        break
if isPrime:
    print(x,"是質數")
else:
    print(x,"不是質數")
```

In [ ]:

In [ ]:
```
# 印出 1~20(含) 之間的質數
primes=[]
for x in range(1, 20+1):
    isPrime=True
    if x==0 or x==1:
        continue
    for i in range(2, x):
        if x%i == 0:
            isPrime=False
            break
    if isPrime:
        print(x,"是質數")
        primes.append(x)
    #else:
        #print(x,"不是質數")
sum(primes)
```

In [ ]:
```

```

4.1 修改版:可以少算一半次數 或是 開根號次數

In [ ]:
```
import math

x = 4
isPrime=True
if x==0 or x==1:
    isPrime=False
for divisor in range(2, int(x/2)+1):
    if x%divisor == 0:
        isPrime=False
        break
if isPrime:
    print(x,"是質數")
else:
    print(x,"不是質數")
```

In [ ]:
```
# 注意:range是不包含右邊界值
for i in range(2,5):
    print(i)
```

In [ ]:
```
# 條件沒滿足
for i in range(2,1):
    print(i)
```

In [ ]:
```
# 條件沒滿足
for i in range(2,2):
    print(i)
```

In [ ]:
```

```

In [ ]:
```
x=5
print(x/2)
print(int(x/2))
```

In [ ]:
```
x=4
print(x/2)
print(int(x/2))
```

In [ ]:
```

```

In [ ]:
```
# 印出 1~20(含) 之間的質數
primes=[]
for x in range(1, 100+1):
    isPrime=True
    if x==0 or x==1:
        continue
    for i in range(2, int(x/2)+1):
        if x%i == 0:
            isPrime=False
            break
    if isPrime:
        print(x,"是質數")
        primes.append(x)
    #else:
        #print(x,"不是質數")
sum(primes)
```

In [ ]:
```

```

4.2 寫成函數方便呼叫

In [ ]:
```
# 寫成函數方便呼叫
def isPrime(x):
    isPrime=True
    for i in range(2, x):
        if x%i == 0:
            isPrime=False
            break
    return isPrime
```

In [ ]:
```
isPrime(5)
```

In [ ]:
```
isPrime(4)
```

In [ ]:
```
# 右邊界可以改成  < x/2
```

In [ ]:
```

```