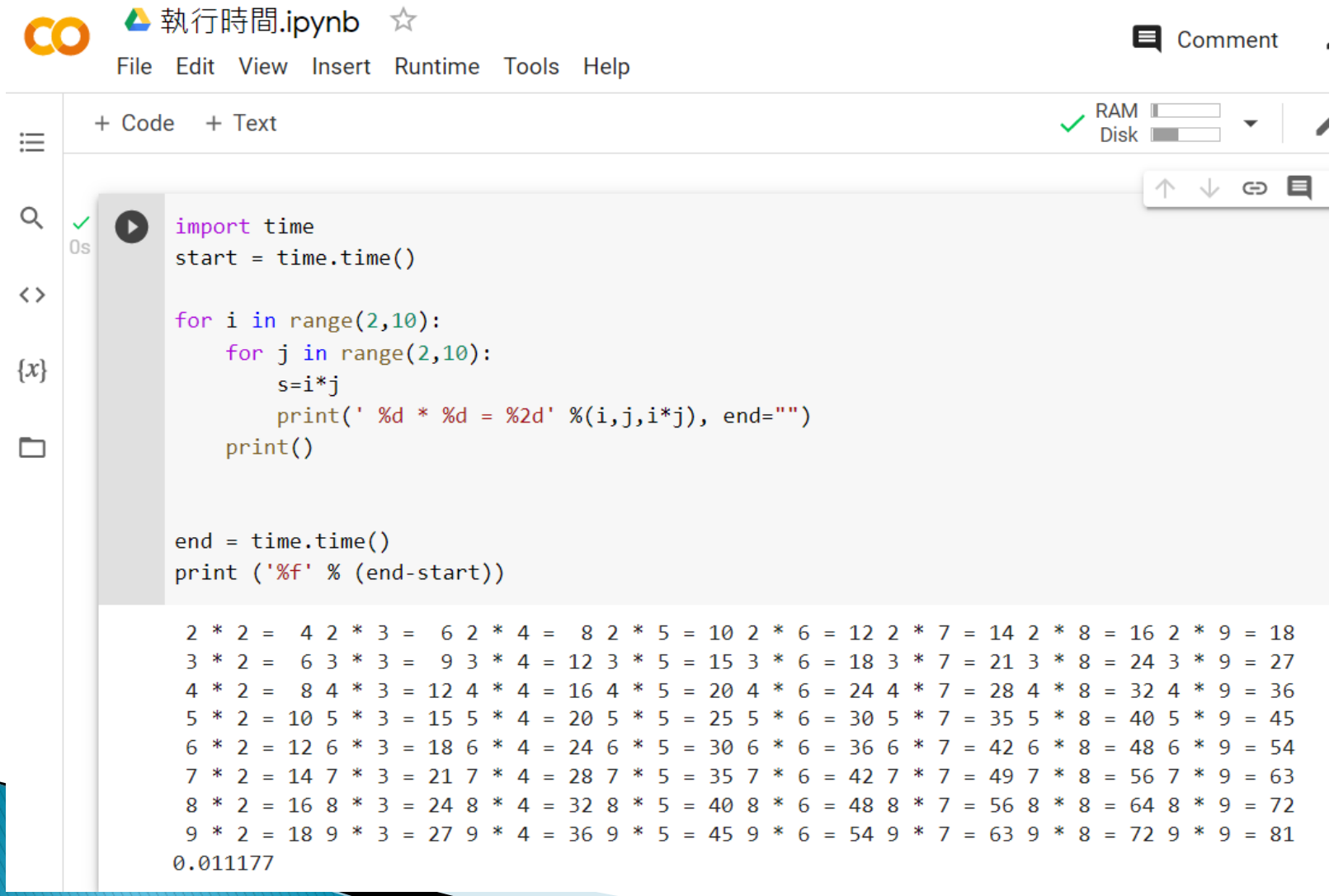
程式設計基本概念
Python語言程式設計(二)

國立臺中教育大學 數位內容科技學系
吳智鴻 教授

用google colab雲端環境開發Python



Example #1 計算程式執行時間

- 目的:計算出程式執行時間
- 可以了解程式寫的好壞
- 好的程式設計師，寫出來的程式品質較高
  - 容易閱讀(註解)
  - 容易維護(結構化)
  - 程式執行效率快
  - 程式沒有BUGS

取得執行時間的程式

- 概念: 執行時間= (程式結束時間-程式開始時間)/執行頻率

```
#  計算執行時間
import  time
start  =  time.time()

#  程式

end  =  time.time()
print("%f"  %  (end-start))
```

```
0.000032
```

需引用time

程式開始時間

把需要紀錄執行時間的程式碼加在這邊

程式結束時間

計算程式執行時間

教學網站上 程式碼連結

取得執行時間

```
#計算執行時間  99乘法表
import  time
start  =  time.time()

#  程式
for  i  in  range(2,10):
        for  j  in  range(2,10):
            print('%d  *  %d  =  %2d  '  %(i,  j  ,  i*j),end="")
        print()

end  =  time.time()
print("執行時間  %f  秒"  %  (end-start))
```

```
2 * 2 =  4 2 * 3 =  6 2 * 4 =  8 2 * 5 = 10 2 * 6 = 12 2 * 7 = 14 2 * 8 = 16 2 * 9 = 18
3 * 2 =  6 3 * 3 =  9 3 * 4 = 12 3 * 5 = 15 3 * 6 = 18 3 * 7 = 21 3 * 8 = 24 3 * 9 = 27
4 * 2 =  8 4 * 3 = 12 4 * 4 = 16 4 * 5 = 20 4 * 6 = 24 4 * 7 = 28 4 * 8 = 32 4 * 9 = 36
5 * 2 = 10 5 * 3 = 15 5 * 4 = 20 5 * 5 = 25 5 * 6 = 30 5 * 7 = 35 5 * 8 = 40 5 * 9 = 45
6 * 2 = 12 6 * 3 = 18 6 * 4 = 24 6 * 5 = 30 6 * 6 = 36 6 * 7 = 42 6 * 8 = 48 6 * 9 = 54
7 * 2 = 14 7 * 3 = 21 7 * 4 = 28 7 * 5 = 35 7 * 6 = 42 7 * 7 = 49 7 * 8 = 56 7 * 9 = 63
8 * 2 = 16 8 * 3 = 24 8 * 4 = 32 8 * 5 = 40 8 * 6 = 48 8 * 7 = 56 8 * 8 = 64 8 * 9 = 72
9 * 2 = 18 9 * 3 = 27 9 * 4 = 36 9 * 5 = 45 9 * 6 = 54 9 * 7 = 63 9 * 8 = 72 9 * 9 = 81
執行時間 0.008610 秒
```

副程式概念

- 把經常會使用到的一段程式碼寫成副程式
- 好處
  - 容易維護
  - 主程式更簡潔
  - 使用有彈性

Example #2 計算加總 (副程式練習)

- 要求
  - 使用者輸入數字n
  - 使用迴圈計算1+2+3+....+n的總數
  - 使用副程式，來做計算這個功能
  - 加入在前述的執行時間計算

如何撰寫副程式

```
def  函數名稱（參數）：
    程式碼
    return  回傳值
```

Prg2-2
計算加總1+2+...n，並寫成副程式

```
#  prg3  加總副程式
import  time
start  =  time.time()

n  =  int(input('請輸入數字  ?'))

def  mysum(s):
    sum=0
    for  i  in  range(0,s+1):
        sum  =  sum  +  i
    return  sum

#你的程式
print('1  +  2  +  ....+  ',n,  '=',mysum(n))

end  =  time.time()
print("花費時間  %f  秒"  %  (end-start))
```

副程式sub_sum範圍

② 把加總結果sum傳回主程式

① 把n傳給副程式的參數s

```
請輸入數字 ?10
1 + 2 + ....+  10 = 55
花費時間 2.239918 秒
```

執行結果

副程式

- 瞭解副程式能夠提升你的程式寫作層次，又更提高一階了。
- 善用副程式可以提高程式閱讀性與結構性。
- 副程式有幾種格式
  - 需回傳資料至主程式，並需要傳參數給副程式
    - int sub_prg (int a)
  - 不需回傳資料至主程式，並需要傳參數給副程式
    - void sub_prg(int a)　　//void表示不用傳資料
  - 不需回傳資料至主程式，不需傳參數給副程式
    - void sub_prog(void)

實例

```
1. void no_return(void);              // 無引數，無傳回值
2. void sub_prog(int n);              // 一個引數，無傳回值
3. double sub_dbl(void);              // 無引數，有傳回值
4. char ch_sub(char ch);              // 一個引數，有傳回值
5. float ft_sub(double d, short si, long li);   // 三個引數，有傳回值
```

課堂練習實做

多重迴圈練習

以星星列印為例

Prg stara

- 輸入一個整數n, 顯示以下星星

```
#  prg4  startA
import  time
start  =  time.time()

n  =  int(input('請輸入數字  ?'))


def  mysum(s):
    sum=0
    for  i  in  range(0,s+1):
        sum  =  sum  +  i
    return  sum

#你的程式

for  i  in  range(0,  n+1):
    for  j  in  range(0,  i+1):
        print('*',end="")
    print('\n')

end  =  time.time()
print("花費時間  %f  秒"  %  (end-start))
```

```
請輸入數字 ?5
*

**

***

****

*****

******

花費時間 1.783933 秒
```

進階練習
把程式改寫成副程式

starA
把列印星星改寫成副程式

```python
# starA
import time
start = time.time()

n = 5

def starA(n):
  for i in range(0, n):
    for j in range(0,i+1):
      print('*',end="")
    print()

starA(n)

end = time.time()
print('%f 秒' % (end-start))
```

```
*
**
***
****
*****
0.007276 秒
```

starB
把列印星星改寫成副程式

```
#  prg4  star_sub
import  time
start  =  time.time()

n  =  int(input('請輸入數字  ?'))

#副程式
def  starA(n):
    for  i  in  range(0,  n+1):
        for  j  in  range(0,  i+1):
            print('*',end="")
        print('\n')

def  starB(n):
    for  i  in  range(0,  n+1):
        for  j  in  range(0,  n-i):
            print('*',end="")
        print('\n')

#主程式
starA(n)
starB(n)

end  =  time.time()
print("花費時間  %f  秒"  %  (end-start))
```

```
*****

****

***

**

*
```

課堂作業

以多重迴圈，完成列印以下星星

- 使用輸入輸出
- 使用多重迴圈
- 使用副程式
- 計算總共執行時間

程式設計基本練習
用多重迴圈與輸入，完成以下的圖形。使用者可以輸入任何數字。
把所有圖形寫在同一隻程式裡面，並將程式碼＆執行檔上傳 E-learning。

Prg #A Please enter a integer: 5 * ** *** **** *****	Prg #D * ** *** **** *****
Prg #B ***** **** *** ** *	Prg #E * *** ***** ******* *********
Prg #C * ** *** **** ***** **** *** ** *	Prg #F * *** ***** ******* ********* ******* ***** *** *

主程式

- 每個星星列印，以
- starA ~
- starB….

```
import time
start = time.time()

n = 5

def starA(n):
  for i in range(0, n):
    for j in range(0,i+1):
      print('*',end="")
    print()

def starB(n):
  for i in range(0, n+1):
    for j in range(0,n-i):
      print('*',end="")
    print()

starA(n)
print()
starB(n)

end = time.time()
print('%f 秒' % (end-start))
```

```
*
**
***
****
*****

*****
****
***
**
*

0.008862 秒
```

執行結果

```
Please enter a integer: 5

--- 副程式 sub_sum (1 + ... n) ---
Total = 15

--- 副程式 prg#A ---
*
**
***
****
*****

--- 副程式 prg#B ---
*****
****
***
**
*

--- 副程式 prg#C ---
*
**
***
****
*****
****
***
**
*

--- 副程式 prg#D ---
    *
   **
  ***
 ****
*****

--- 副程式 prg#E ---
    *
   ***
  *****
 *******
*********

--- 副程式 prg#F ---
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *

進行運算所花費的時間： 2.34 Second
請按任意鍵繼續 . . .
```

先試著自己做做看。
真的做不出再看

教學網站上 程式碼連結