程式設計(二)-HW01

Due to 03/24 PM 11:59／授課老師：紀博文

一、基本資料

姓名：林育辰
系級：資工 111
學號：40771131H

二、檔案有哪些？

1) hw0101.c
2) mystring.c
3) mystring.h
4) hw0102.c
5) hw0103.c
6) hw0104.c
7) rational.c
8) rational.h
9) hw0105.c
10) hw0106.c
11) Makefile
12) README.pdf

◎每個.c 檔皆有詳細註解！問題的回答寫在 README 每題詳細說明中！
◎ hw0101.c, hw0104.c 為原先所提供，第 1 題主要為 mystring.c .h 兩個檔，第 4 題主要為 rational.h .c 兩個檔，在 hw0104.c 中，測資請勿使程式產生 warning

三、如何執行？

請輸入 make→編譯 hw0101.c~hw0106.c→產生 hw0101~hw0106 檔
指令如下：

```
$ make
$ ./hw0101
$ ./hw0102
…
```

以此類推，即可分別執行 hw0101~hw0106

四、索引

說明

1 My String Library (20 pts)

In this class, I have shown you some standard string functions. Of course, you should use them when coding. However, sometimes you may need some functions that are not included in the standard string library. Do not worry, you can implement on your own. For your practice, I want you to implement some existing string functions in your own way.

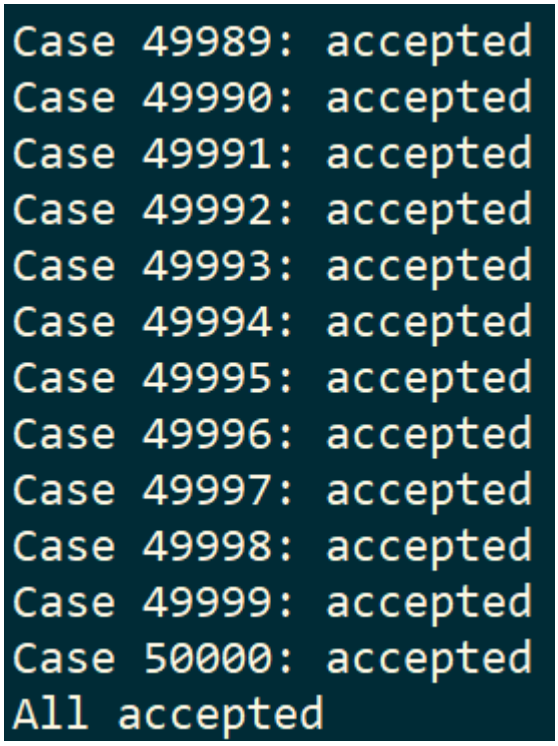
```
char *mystrchr(const char *s, int c);
char *mystrrchr(const char *s, int c);
size_t mystrspn(const char *s, const char *accept);
size_t mystrcspn(const char *s, const char *reject);
char *mystrpbrk(const char *s, const char *accept);
char *mystrstr(const char *haystack, const char *needle);
char *mystrtok(char *str, const char *delim);
```

The usage of these functions should be the same with the standard version, including their return values. You need to prepare **mystring.h** and TA will prepare **hw0101.c**. Of course, Makefile is your own business. I prepare an example code for you, but it will not be the TA test file.

◎題意說明→ 在本課程中，已操演過如何使用一些字串函數，然而有時候你需要使用一些功能但他不包含在標準函式庫內，不用擔心，你能夠自己寫，為了讓你自己練習，你必須實作 strchr, strrchr, strspn, mystrcspn, strpbrk, strstr, strtok 等 7 個函式，你的功能應該完全和標準的函式庫相同，並且準備 mystring.h，助教會準備 hw0101.c。

※注意

1) mystring.c 中的所有函式，皆經過自行產生測資並和原始標準函式庫比較，每個函式都已產生過 5 萬個測資去檢查結果是否正確



2) 有額外撰寫除以上 7 個函式庫以外的函式，例如 mystrcmp，此僅為撰寫後面的標準函式庫方便而寫，**和標準函式庫的 strcmp 無關**

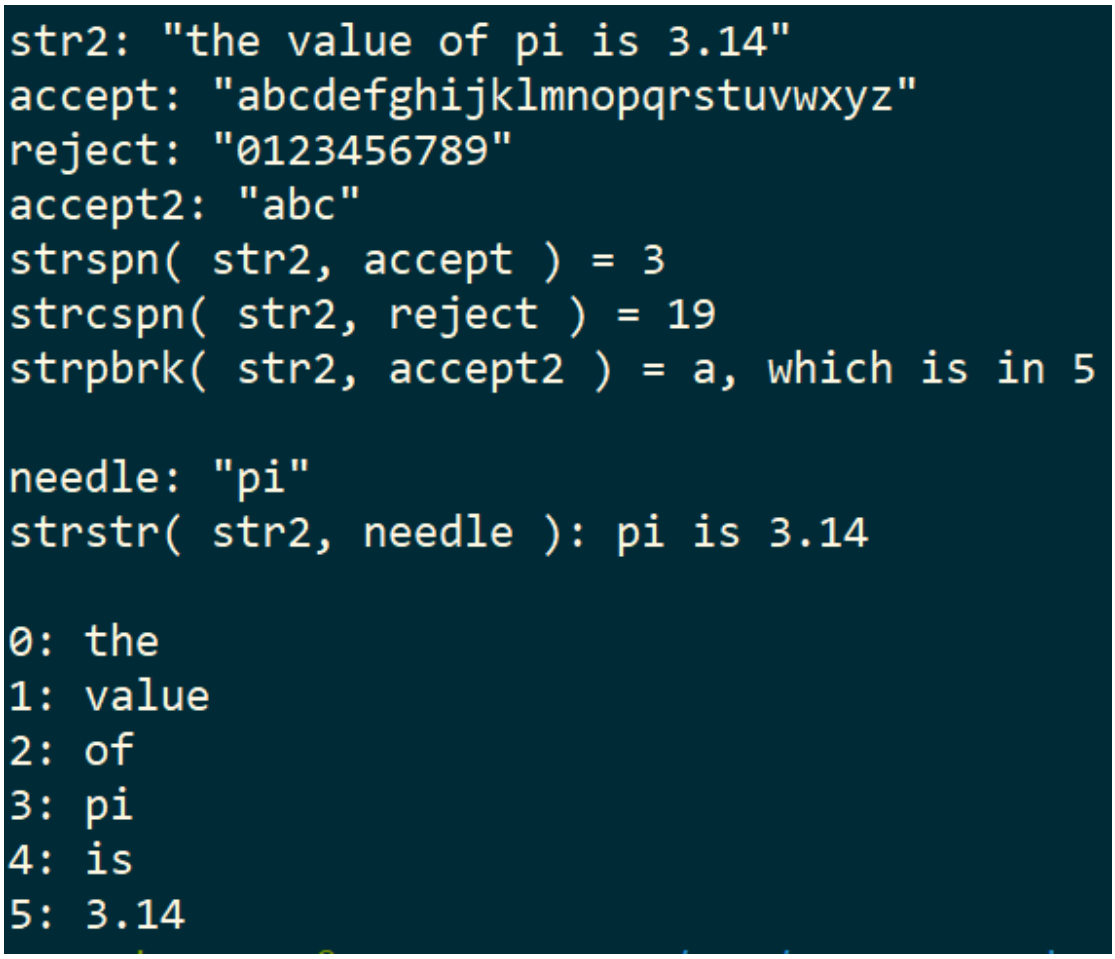
◎輸出格式

編譯後，執行"$ ./hw0101"

其結果依據助教所寫之 hw0101.c 之內容而定

按照範例 hw0101.c，則輸出如下：

```
┌─yuchen0515@NTNUMATHLIN ~/pra/40771131H_hw01 ‹master*›
└─$ ./hw0101
str: "I have an apple."
strchr( str, c ) = ave an apple.
location is 3
strrchr( str, c ) = apple.
location is 10
```



◎程式設計思路

1) mystrchr：單純往後比對字元，找到回傳位置，沒找到回傳 NULL

2) mystrrchr：一樣往後比對字元，但用一個指標去儲存找到的位置，如此一直比對到最後一個，最後一個被指標儲存的即為從後往前符合的第一個字元位置

3) mystrspn：第一個是掃瞄字串，第二個有點像集合的概念，我們可以很簡單使用 mystrchr 去比對，他會從頭開始往後找，直到找到不屬於第二個字串集合的字元就回傳到目前位置的長度

4) mystrcspn：和第 3 個函式庫剛好相反，改成!=即可

5) mystrpbrk：有點像 mystrspn，不過掃瞄時，在 accpet 中若找到符合的字元，則回傳 s 目前所在位置，都沒找到就回傳 NULL

6) mystrstr：利用 mystrchr 比對在第一個字串中，出現第二個字串第一個字的位置，接著開始往後掃看看是不是兩個相等，不相等我就再繼續用 mystrchr 找出等於第二個字串第一個字的位置…

7) mystrtok：比較特別就是最重大的技巧應該使用動態變數去存上一個 tok 的位置，接著就是善用 strpbrk 找到分割字元，然後將此位置設為 0 再往後一格

說明

2 MAC Filter (20 pts)

In this class, you have practiced how to write a program to verify if a given string is a valid IPv4 address. This time, I want you to develop a program to check if a given string is a valid MAC address.

A media access control address (MAC address) is a unique identifier assigned to a network interface controller (NIC) for use as a network address in communications within a network segment. MAC Address is a 12-digit hexadecimal number (6-Byte binary number), which is mostly represented by Colon-Hexadecimal notation. You can use **ifconfig** to get the MAC address of your network card. An example is as follows:

- 88:d7:f6:54:15:0a

Of course, your program should accept both the upper case and the lower case. The string **88:d7:f6:54:15:0a:aa** is definitely invalid. If a string is a valid MAC address, you should print the MAC address with the hypen-hexadecimal notation. Otherwise, print **invalid**.

◎題意說明→在課堂上，我們練習了 IPv4 位址的判別，因此現在想讓你們改練習 MAC address，他共有 12 位的 16 進位數字所組成，而兩個兩個間用”-“相連，你的程式必須大小寫都能夠接受，若輸入無效則輸出”invalid”，若合法則輸出使用 hypen-hexadecimal notation 的結果

◎輸入&輸出格式

```
$ ./hw0102
Please enter the string: 88:d7:f6:54:15:0a
88-d7-f6-54-15-0a
$ ./hw0102
Please enter the string: 88:d7:f6:54:15:0a:22
invalid
```

編譯後，執行”$ ./hw0102”，輸入：
MAC address，若您輸入不合法，甚至是非常長的字串，將顯示 invalid，若數字和數字間包含空白，我們會無視該空白，如 8 8: d 7 : f 6: 54: 1　5: 0 a，此為合法的輸入

◎程式設計思路

1) 我們只要思考，怎樣會是一個合法的輸入？合法的輸入應該為 0-9 或 a[A]-f[F]的數，而且兩個兩個相連，彼此用冒號隔開

2) 若有空白必須把他讀掉

3) 如果已經是不合法的輸入，必須將後面全部讀掉

4) 因此只要設置 count 變數去計算已經出現幾次合法數字，當 count=2 時，我要檢查下一個是不是冒號，如果是，就 count=1 再繼續計算(num_count 則計算冒號出現幾次，理應出現五次，若超過則視作無效)

5) 因我們只用 20 格陣列去儲存答案，其餘皆以 getchar 先檢查是否為合法再行處理，如此無論使用者輸入有多大，皆能占用最少記憶體的情況下處理完成

◎各情形範例

1) Tip: test case #1

```
┌─yuchen0515@NTNUMATHLIN ~/pra/40771131H_hw01 ‹master*›
└─$ ./hw0102
Please enter the string: 88:d7:f6:54:15:0a
88-d7-f6-54-15-0a
```

2) Tip: test case #2

```
Please enter the string: 88:d7:f6:54:15:0a:22
invalid
```

3)Tip:　包含空白的情況，但合法

```
Please enter the string: 88  : d7 : f 6 : 54: 15 : 0 a
88-d7-f6-54-15-0a
```

4) Tip: 有非十六進制的數

```
Please enter the string: 89:d7:fk:54:15:0a
invalid
```

5) Tip: 嚴重不符合格式

```
Please enter the string: 89:d7:f6:54:1p:0a:ok:ds:defewfwjkflewhfjkwfncewcnweidowhedhuwe
invalid
```

6) Tip: 無論輸入有多大，都要能讀掉

```
Please enter the string: 88:d7:f6:54:15:0a          ddddddddddddddddddddddddddddddddddddddddddddddddd
ddddddddddddddddddddddddddddddd
invalid
```

說明

3 Wildcard Matching (20 pts)

Undoubtedly, C standard string library provides some string matching functions, like **strcmp**, **strstr**. However, sometimes we want a string matching function that supports **patterns** instead of exactly words. Now I introduce a pattern called **wildcard**. First, you need to learn two symbols:

- ?: Matches any single character.
- *: Matches any sequence of characters (including the empty sequence).

For example, given a pattern **a?e**:

- **ae** does not match the pattern.
- **ace** matches the pattern.
- **ache** does not match the pattern.

For example, given a pattern **a*e**:

- **ae** matches the pattern.
- **ace** matches the pattern.
- **ache** matches the pattern.
- **apple** matches the pattern.

Now, given a pattern and an input string, please find all words that match the pattern. For your convenience, all test strings are composed of English lowercase alphabets only. The string is less than 512 bytes.

◎**題意說明**→C 的標準函式庫已經提供一些 string matching 的功能，像是 cmp, strstr 等等，然而有時候我們想要比對 pattern 樣式，以下定義兩個符號：

?: 只「**恰好**」匹配一個字元

*: 可匹配任何字元，包括**空字串**

現在給一個 pattern 和一個輸入字串，請找出所有匹配這個 pattern 的字串，為了同學的方便，所有測資**僅由小寫字母所組成**，並且 **string 低於 512 bytes**

※注意

1) pattern 不得輸入空白，如 a k p ddd 如此字串，但前後空白是可以的，例如「 *a?m 」

2) input string 是由空白所分割的，其中**不得輸入任何小寫以外的文字**，文字間**僅可包含空白做為分割字元**

3) 輸入不得超過 512 bytes 的大小

◎輸入&輸出格式

```
$ ./hw0103
Please enter the pattern: adam
Please enter the string: madam I am adam
Result: adam
$ ./hw0103
Please enter the pattern: ?adam
Please enter the string: madam I am adam
Result: madam
$ ./hw0103
Please enter the pattern: *a*m
Please enter the string: madam I am adam
Result: madam am adam
```

編譯後，執行"$ ./hw0103"

1) 請先輸入一個 pattern，並且遵守以上「注意」的事項

2) 接著輸入一個字串，其中僅可以「小寫字母」所組成

◎程式設計思路

1) 可以利用自動機的想法去思考，我們將 Input string 當作自動機的 state，pattern 當作輸入的判斷

2) 當 state 符合「目前指到的」pattern，或 pattern 等於「?」就能進入下一個 state，若等於「*」則有可能代表(a)空的 (b)好幾個字元 (c) 當作問號，*以 3 個遞迴去處理

3) 若不符合直接到 END，進行下一匹字串的比較

4) 最後利用 strtok 去分割字串，一個一個比較就能夠完成

◎各情形範例

1) Tip: test case #1

備註：%為 cmder 模擬終端機有時會出現的字元，並不是原本的輸出

```
┌─yuchen0515@NTNUMATHLIN ~/pra/40771131H_hw01 ‹master*›
└─$ ./hw0103
Please enter the pattern: adam
Please enter the string: madam I am adam
Result: adam %
```

2) Tip: test case #2

```
┌─yuchen0515@NTNUMATHLIN ~/pra/40771131H_hw01 ‹master*›
└─$ ./hw0103
Please enter the pattern: ?adam
Please enter the string: madam I am adam
Result: madam %
```

3) Tip: test case #3

```
Please enter the pattern: *a*m
Please enter the string: madam I am adam
Result: madam am adam %
```

4) Tip: 連續多重*?連環的情況

```
Please enter the pattern: ?*?aa*mm*
Please enter the string: psfrkaanfbemmadke kkaamm paamm kqdaapmml
Result: psfrkaanfbemmadke kkaamm kqdaapmml %
```

5) Tip: *連環的情況

```
Please enter the pattern: *****am?****?
Please enter the string: amil dsfeamjdkew samiu kjdeamk
Result: amil dsfeamjdkew samiu %
```

說明

4 Rational Number Arithmetic (20 pts)

In mathematics, a rational number is a number that can be expressed as the quotient or fraction $\frac{p}{q}$ of two integers, a numerator p and a non-zero denominator q.

Please develop a structure for the rational number and implement some arithmetic operations. You need to provide a header file **rational.h** and its .c file. TA will prepare hw0104.c which include your header file. Note that you should prepare a Makefile.

```
struct rational {
};

int rational_set( struct rational *r, int32_t p, int32_t q);
// return -1 if invalid; otherwise, return 0.
int rational_print( const struct rational r);
// in the form of (p,q)
void rational_add( struct rational *r, const struct rational r1, const struct
    rational r2);
// r = r1 + r2
void rational_sub( struct rational *r, const struct rational r1, const struct
    rational r2);
// r = r1 - r2
void rational_mul( struct rational *r, const struct rational r1, const struct
    rational r2);
```

```
// r = r1 * r2
void rational_div( struct rational *r, const struct rational r1, const struct
    rational r2);
// r = r1 / r2
```

Note that all rational number should be in the irreducible form.

◎題意說明→數學上，有理數可被表示成分數的型態，請使用 structure 去實作分數的加減乘除，並且準備 rational.h 的標頭檔，和他的.c 檔，助教會準備 hw0104.c 去 include 你的 header file，你應該準備好你的 Makefile

◎注意

1) 此題目為 **rational** number，**虛數並不包括在有理數內**
2) 若該運算結果或分數不合法，會提示你的分母為零因此無法計算

◎輸出格式

```
┌─yuchen0515@NTNUMATHLIN ~/pra/40771131H_hw01 ‹master*›
└─$ ./hw0104
(9,5)
(-1,1)
(14,25)
(2,7)
```

編譯後，執行"$ ./hw0104"
依序為加、減、乘、除的結果 (此為範例 hw0104.c 之執行結果)

◎程式設計思路

1) 加減法部分需要考慮通分，而通分即找兩個分母的最小公倍數即可
2) 為了找最小公倍數，我們必須實作 gcd，並且要小心負數的情況
3) 乘法、除法最為重要的在於檢查分母是否變為零
4) 化為最簡分數，需要做的是就是去找分子、分母的 gcd，找到後同除以他便能化為最簡分數
5) 要注意當分子為 0，分母必須化為 1，當有兩個負號要記得變成正號，而負號一律擺在分子的位置
6) 因此只要 rational_set 寫好，包括驗證的函式，就可以很簡單的完成此題

說明

5 Rational Number Arithmetic Part 2 (20 pts)

Now, please write a program to calculate the result of a given equation. For example, if a user wants to get the result of the following equation:

$$\frac{1}{2}+\frac{5}{6}\times\frac{3}{10}$$

```
$ ./hw0105
Please enter the equation: (1,2) + (5, 6)*(3,10)
(3,4)
```

Of course, you must follow arithmetic operation precedence. For simplicity, you do not need to consider parentheses.

◎**題意說明**→請寫一個程式去計算如下列方程式的結果，而分數的輸入形態為(a,b)，為了你們方便，你**不需要考慮**括號的優先順位問題

◎輸入&輸出方式

編譯後，執行"$ ./hw0105"

→一個合法的輸入為 (a,b)，且兩個分數間必定有運算符號

→數字並不包含虛數、字母等合法整數以外的東西

→最終輸出(a,b) [運算結果]

◎注意

1) 你輸入的合法方程式，不得超過 99999 個分數、運算子
2) 此題目為"rational number"，因此**並不會考慮虛數**的情況，他不是有理數

◎程式設計思路

1) 最主要就是「要如何去驗證使用者輸入的字串為合法的？」
2) 當能解決此問題，並正確使用 struct 存取輸入、array 存取運算子基本上就完成了
3) 因此對於輸入，我設置 readend, readspace, readsign, readfraction 的函式，分別讀掉所有輸入、讀掉空白、讀取符號並存進 array 內、讀取分數並存進 array structure 內
4) 對於 readfraction，讀取到左括號，之後正確讀取到所有的整數(包含負號)，之後必須還要一個逗號，再持續以上動作，最終讀取到右括號，以上皆無誤才叫合法輸入，若為合法輸入就丟進 array 中，不合法就停止並提示輸入錯誤！
3) 當輸入沒問題以後，我只要處理加減乘除即可，現在我有一條分數的陣列、一條符號陣列，我用 index 一個一個去跑，當讀取到乘法或除法時，我就將該 index 的分數和下一個做乘法或除法，並存到目前 index 中，然而 index+1 的分數我就直接用後面的所有分數去覆蓋掉，符號當然也一樣
4) 當我進行這樣的運算後，我必須將 index 扣回去，然後繼續運算，當結束後，此時運算子已經沒有乘法和除法，接著再重覆同樣的步驟跑加法、減法就完成了

◎各情形範例

INPUT #1： `(1,5)*(10,7)+(9,14)/(9,5)`

OUTPUT #1: `(9,14)`

```
┌─yuchen0515@NTNUMATHLIN ~/pra/40771131H_hw01 ‹master*›
└─$ ./hw0105
Please enter the equation: (1,5)*(10,7)+(9,14)/(9,5)
(9,14)
```

INPUT #2: `(7,4)-(3,5)*(3,8)`

OUTPUT #2: `(61,40)`

```
Please enter the equation: (7,4)-(3,5)*(3,8)
(61,40)
```

INPUT #3: `(29,8)*(16,29)/(5,3)`

OUTPUT #3: `(6,5)`

```
Please enter the equation: (29,8)*(16,29)/(5,3)
(6,5)
```

INPUT #4: `(95,8)-(7,4)-(11,6)`

OUTPUT #4: `(199,24)`

```
Please enter the equation: (95,8)-(7,4)-(11,6)
(199,24)
```

INPUT #5: `(95,8)-(11,4)-(11,6)`

OUTPUT #5: `(175,24)`

```
Please enter the equation: (95,8)-(11,4)-(11,6)
(175,24)
```

- **INPUT & OUTPUT #6**

→有很多空白字元必須能正確計算

```
Please enter the equation: (     95 ,  8  ) -  ( 1 1  , 4 ) -   ( 1   1 ,  6 )
(175,24)
```

- **INPUT & OUTPUT #7**

→尾巴雖有運算元，但其左右必須皆有分數才可運算

```
Please enter the equation: (      95 ,  8  ) -  ( 1 1  , 4 ) -   ( 1   1 ,  6 ) +
input error!
```

- **INPUT & OUTPUT #8**

```
Please enter the equation: (      95 ,  8  ) -  ( 1 1  , 4 ) -   ( 1   1 ,  6 )  i
input error!
```

→尾巴有不合法輸入

- **INPUT & OUTPUT #9**

```
Please enter the equation: (      95 ,  8  ) -  ( 1 1  , 4 ) -   ( 1   1 ,  6i )
input error!
```

→不處理虛數的情況，題目是有理數的四則運算

- **INPUT & OUTPUT #10**

→超級不合法的輸入

```
Please enter the equation: ()Sdsd0djfieflfklflnl2fjrke jfelfj ekjf lefj iefh legrji ile
input error!
```

- **EXAMPLE TEST**

→作業 hw01 中所附測資

```
Please enter the equation: (1,2) + (5,6) * (3,10)
(3,4)
```

說明

6 fgets (5 pts)

fgets needs the programmer to give the buffer size. The problem is that the programmer does not know the user input size. How to solve this problem?Please write **Wildcard Matching** again without the string size limitation. You also need to describe your idea in your readme file.

◎題意說明→ fgets 必須知道 buffer 的大小，在本題中，程式設計師並不知道使用者輸入的大小，那麼要如何解決這個問題呢？請再寫一次 hw0103 的題目，但沒有字串大小的上限，並且需要在你的 readme 內說明你的想法

◎**注意**：

1) 上課時，紀老師說 只有輸入字串是沒有限制使用者輸入大小的，**pattern 輸入過大的情況可以不用管他**，其餘輸入的注意事項同 hw0103 之注意狀況

◎**解釋**

在讀取時，透過 calloc 動態規畫一個記憶體位置，我使用 getchar 去檢查是否有新的字元，若有新的字元時，我就使用 realloc 去重新規畫記憶體空間，每次都多 1 byte 的大小空間，只要將 realloc 回傳的指標位置重新丟回原始指標，跑迴圈直到 getchar == EOF 或\n 即可

Realloc 所做的事情是，再規畫一個記憶體空間，並且將原本的指標位置的所有內容複製到該空間中，只要使用**動態分配記憶體空間**，就不會受限於不知道使用者輸入大小的問題。

您可以直接執行 hw0106 的檔案，觀摩是否和 hw0103 的輸出結果一致，另外，在 hw0106.c 的檔案內有詳細的註解！