**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**

**第一題**──────────────────P.2-4

第二題──────────────────**P.5-7**

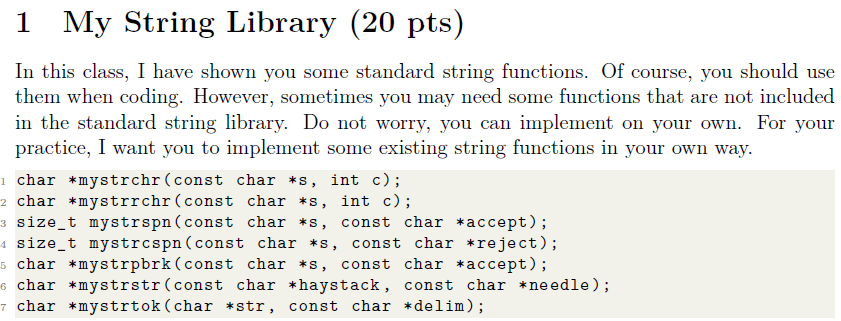
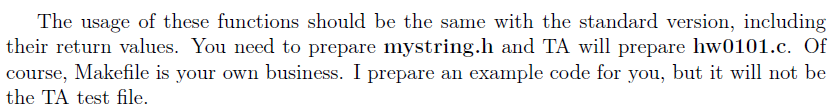
**第三題**──────────────────P.7-9

第四題──────────────────**P.10-11**

**第五題**──────────────────P.11-14

第六題──────────────────**P.14-15**

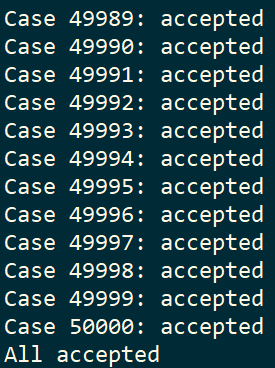
**說明**

**◎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



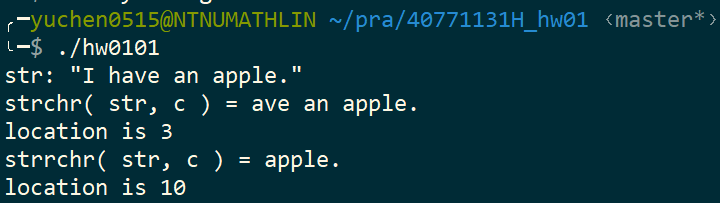
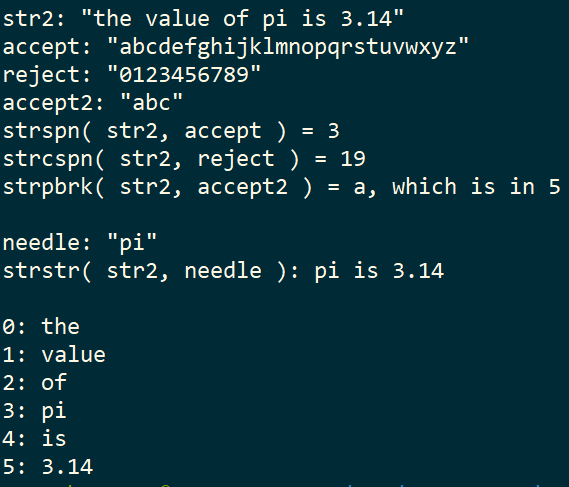
2) 有額外撰寫除以上7個函式庫以外的函式，例如mystrcmp，此僅為撰寫後面的標準函式庫方便而寫，**和標準函式庫的strcmp無關**

**◎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：

**◎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

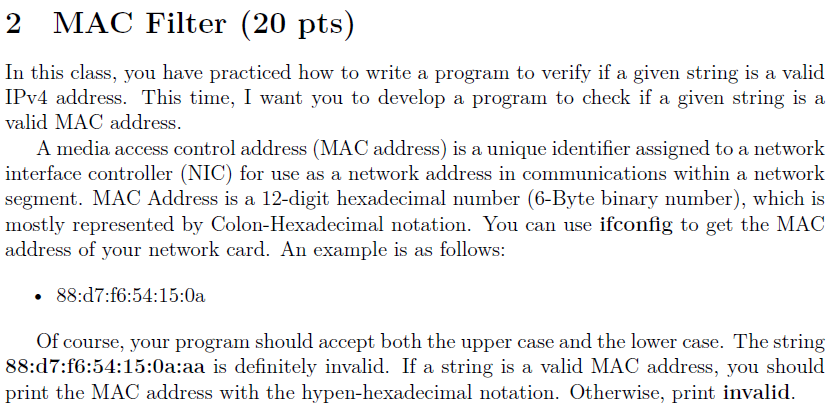
4) mystrcspn：和第3個函式庫剛好相反，改成!=即可

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…

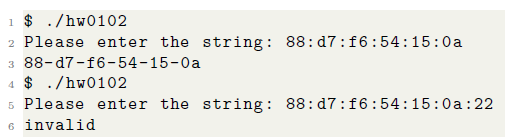
7) mystrtok：比較特別就是最重大的技巧應該使用動態變數去存上一個tok的位置，接著就是善用strpbrk找到分割字元，然後將此位置設為0再往後一格

**說明**



**◎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**



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

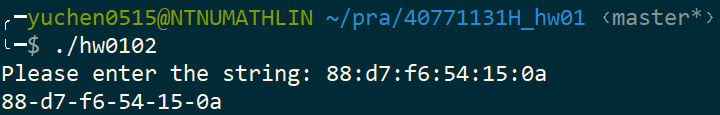
3) 如果已經是不合法的輸入，必須將後面全部讀掉

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)

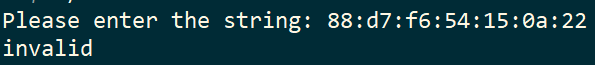
5) 因我們只用20格陣列去儲存答案，其餘皆以getchar先檢查是否為合法再行處理，如此無論使用者輸入有多大，皆能占用最少記憶體的情況下處理完成

**◎各情形範例**

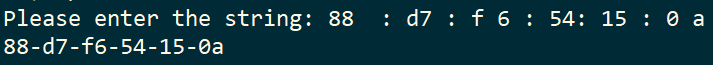
1) Tip: test case #1



2) Tip: test case #2



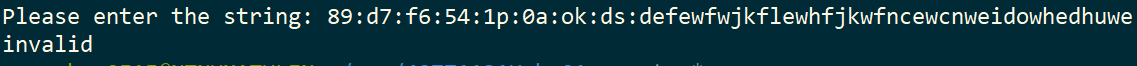
3)Tip: 包含空白的情況，但合法



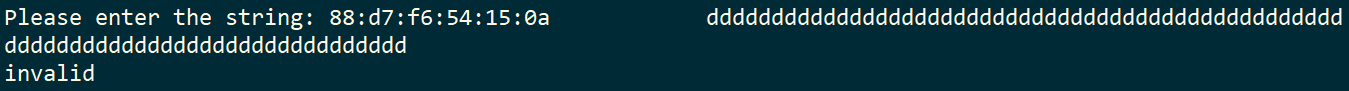
4) Tip: 有非十六進制的數



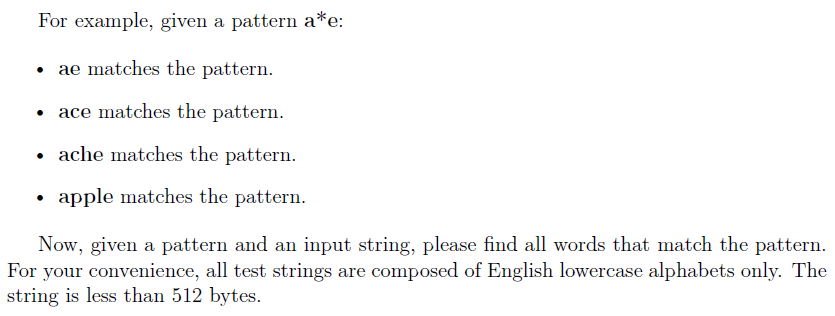
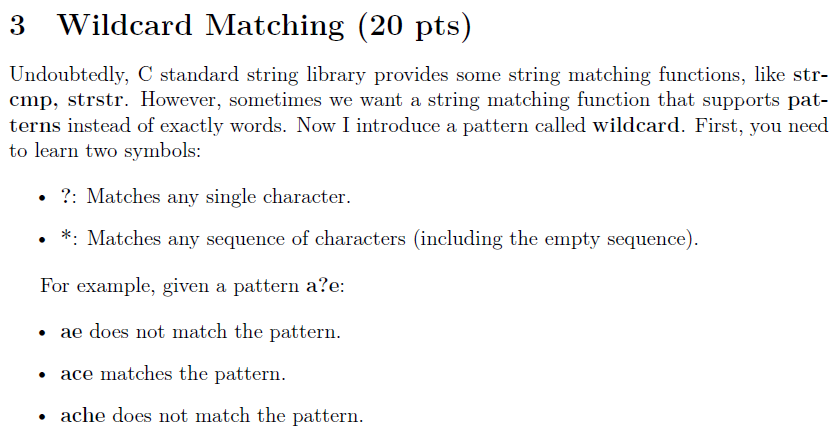
5) Tip: 嚴重不符合格式



6) Tip: 無論輸入有多大，都要能讀掉



**說明**



**◎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**

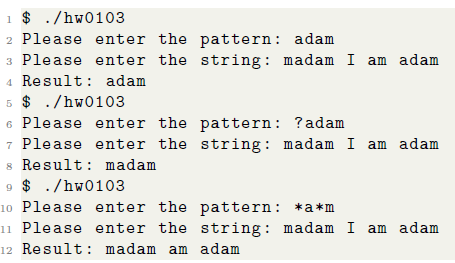
1) pattern不得輸入空白，如a k p ddd如此字串，但前後空白是可以的，例如「

\*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**



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

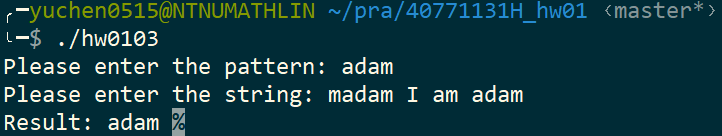
3) 若不符合直接到END，進行下一匹字串的比較

4) 最後利用strtok去分割字串，一個一個比較就能夠完成

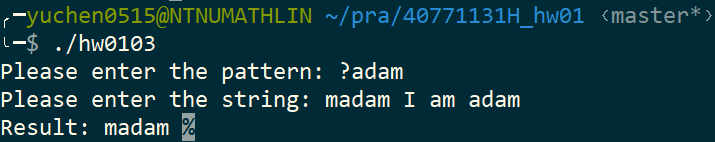
**◎各情形範例**

1) Tip: test case #1

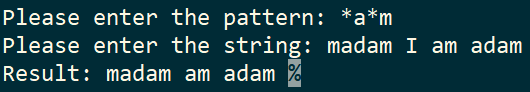
備註：%為cmder模擬終端機有時會出現的字元，並不是原本的輸出



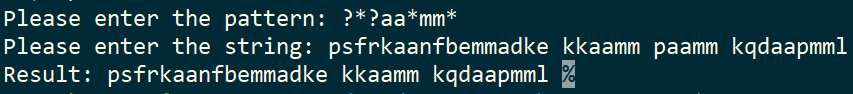
2) Tip: test case #2



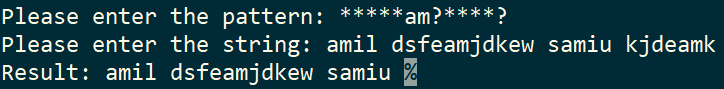
3) Tip: test case #3



4) Tip: 連續多重\*?連環的情況

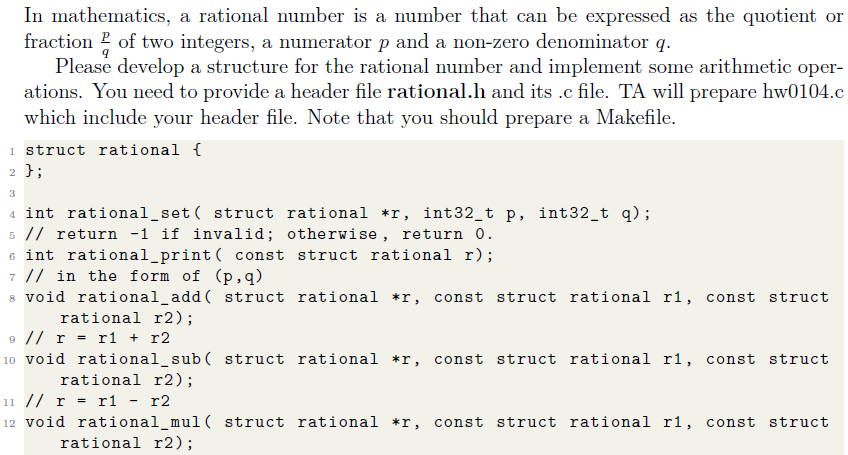
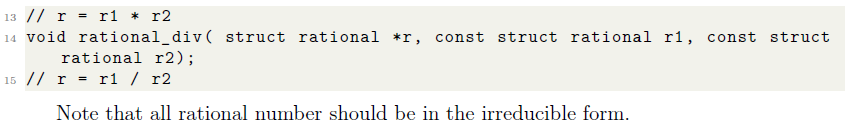


5) Tip: \*連環的情況



**說明**



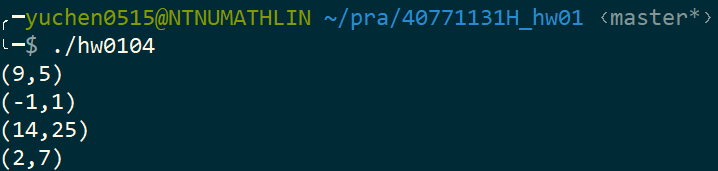
**◎題意說明**→數學上，有理數可被表示成分數的型態，請使用structure去實作分數的加減乘除，並且準備rational.h的標頭檔，和他的.c檔，助教會準備hw0104.c去include你的header file，你應該準備好你的Makefile

**◎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**



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

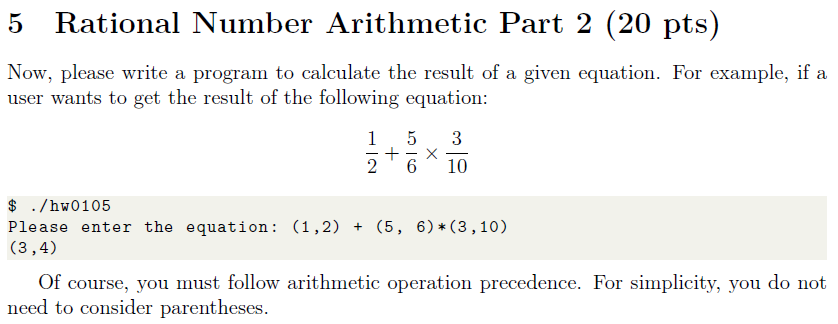
3) 乘法、除法最為重要的在於檢查分母是否變為零

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**



**◎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)，

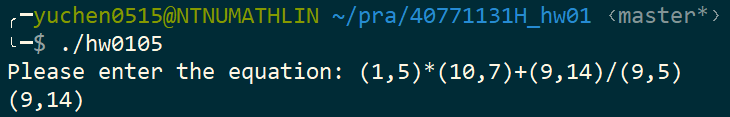
之後必須還要一個逗號，再持續以上動作，最終讀取到右括號，以上皆無誤才叫合法輸入，若為合法輸入就丟進array中，不合法就停止並提示輸入錯誤！

1. 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
2. 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**

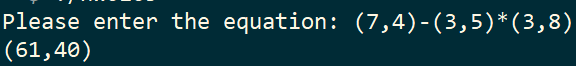
**INPUT #1：**(1,5)\*(10,7)+(9,14)/(9,5)

**OUTPUT #1:** (9,14)



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

**OUTPUT #2:** (61,40)



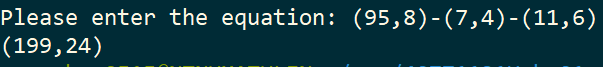
**INPUT #3:**  (29,8)\*(16,29)/(5,3)

**OUTPUT #3:** (6,5)



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

**OUTPUT #4:** (199,24)



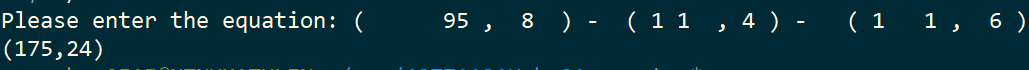
**INPUT #5:** (95,8)-(11,4)-(11,6)

**OUTPUT #5:** (175,24)



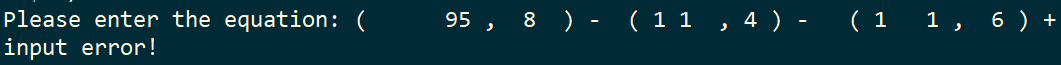
* **INPUT & OUTPUT #6**

**→有很多空白字元必須能正確計算**



* **INPUT & OUTPUT #7**

**→尾巴雖有運算元，但其左右必須皆有分數才可運算**



* **INPUT & OUTPUT #8**



**→尾巴有不合法輸入**

* **INPUT & OUTPUT #9**



**→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**

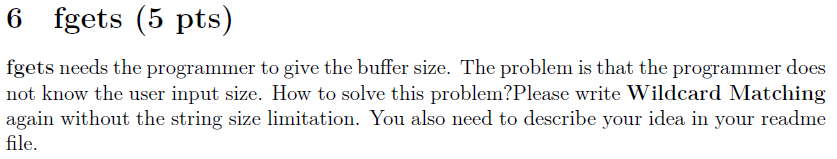


* **EXAMPLE TEST**

**→作業hw01中所附測資**



**說明**



**◎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！