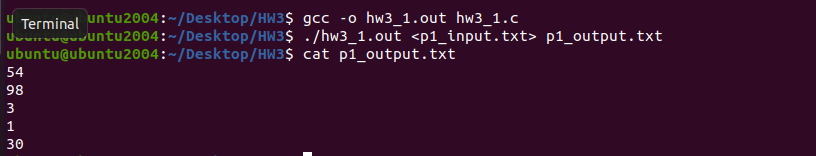
HW3\_1

Result screenshot



Program Architecture

先用函式讀入資料，將盤子建立成堆疊，並在enqueue的時候將POP的盤子的索引值分別建立出A B兩個佇列，並按照題目規則，根據四個不同的指令去執行。詳細的方式會在下面的部份說明

Program function

1. 自定義函式:int power(int a,int b)

這個自定義函式式用來處理PUSH指令後面的random index的，將字元轉換成int型態去儲存的時候，由於不同位數，相當於該位數的數字乘上10的不同次方，為了縮短程式碼並簡化，使其能寫在同一個for回圈內就解決，所以自定義了這個function。a代表底數，b代表指數，以下是這個副程式的內容:

int power(int a,int b)

{

int ans=1;

if(b<1)

{

return ans;

}else

{

for(int i=1;i<=b;i++)

{

ans\*=a;

}

return ans;

}

}

1. Platepush(int index)

這個函式是在PUSH指令的時候，用linked list實作stack，將盤子索引值建立出一個堆疊

1. Platepop()

這是在POP指令的時候，用串列走訪的方式，取出最上方的盤子索引值的函式

4.Aenqueue(int num) Benqueue(int num)

這是把剛剛的Platepop()取出的盤子值以鏈結串列實作queue的方式，丟進A B裡面並建立queue的函式

5.Adequeue() Bdequeue()

這是將A B裡面的開頭盤子取出並印出的函式

2.系統函式

1. char \*strchr(const char \*str, int c)

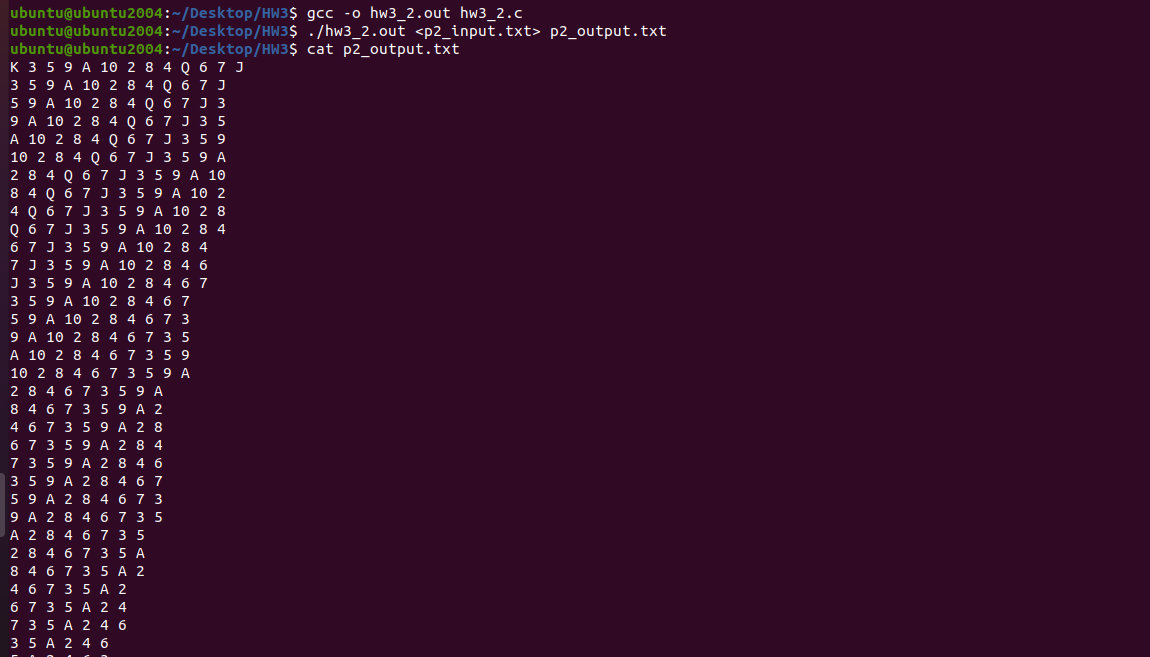
這個函式式我為了處理Unix與windows與MacOs三種不同的OS底下，換行符號的不同而使用的函式，先傳入某個字元陣列的開頭位址，然後再寫入我需要搜尋的目標字元，這個函式就可以幫我找到字串中的目標字元的位址，並回傳那個位址給我，我就可以對目標字元進行修改或處理。

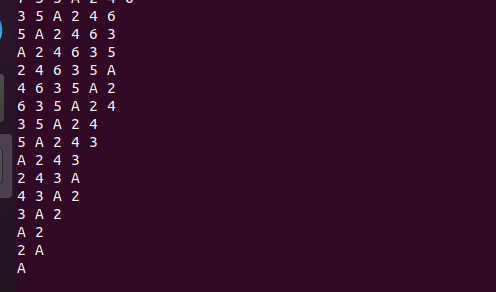
how you design your program

這次作業與上次不同的是，這次要使用linked list去模擬stack和queue，我根據我自己買的一本屬:圖解資料結構(使用C語言)裡面教的，將許多功能，比如POP PUSH DEQUEUE ENQUEUE都寫成副程式，這樣到時候main函式裡面就只需要直接呼叫，非常方便。這次使用到了串列走訪，front與rear指標等。因為堆疊是先進後出，所以只要管開頭那個節點的問題就好，而佇列是先進先出，所以必須設兩個指標去做頭尾的管理。

HW3\_2

Result screenshot





Program architecture

這次是有13張牌，分別含有A~K 13種大小，我先將所有牌都讀入，建立成一個queue，再根據題目規則去判斷front指向的第一個數字是否為我要取的目標數字，如果是就將其取出，不是的話就將其丟到queue尾端。

Program function

自定義函式:

Enqueue(int num)

這是用來一開始建立撲克牌queue，與將不為draw out目標的牌丟到queue尾端的函式。

Dequeue()

這是用來將如果第一張牌就是目標的牌，則將其draw out出來的函式

Int show()

這是利用串列走訪，在每次檢查時均將整個queue印出來的函式

Int check(int num,int mover)

這是用來比對queue第一張牌是否為當前的目標的函式，若是的話救回傳1 不是的話就回傳0

2.系統函式

1. char \*strchr(const char \*str, int c)

這個函式式我為了處理Unix與windows與MacOs三種不同的OS底下，換行符號的不同而使用的函式，先傳入某個字元陣列的開頭位址，然後再寫入我需要搜尋的目標字元，這個函式就可以幫我找到字串中的目標字元的位址，並回傳那個位址給我，我就可以對目標字元進行修改或處理。

How you design your program

第一次看到題目的時候，由於有看到第一張牌丟到最尾端的這個說明，我一開始以為是環狀串列的實作，但是後來發現，我可以利用第一題的enqueue dequeue來完成，並且只需要單向串列就足以實作。我只要利用check函式先去判斷是否第一張牌就是我的目標，如果是的話，直接呼叫dequeue函式，把這張牌從queue中去除就好，如果第一張牌不是我要的目標，則先將這張牌利用enqueue丟入queue末尾(此時queue的頭跟尾都是同一張牌)，接著使用dequeue()將front指標指向的那張牌丟棄，這樣就完成了將牌丟到末尾去重新排隊的動作。這次作業也是利用linked list來實作queue，並且在show()裡面也用到了串列走訪的動作。