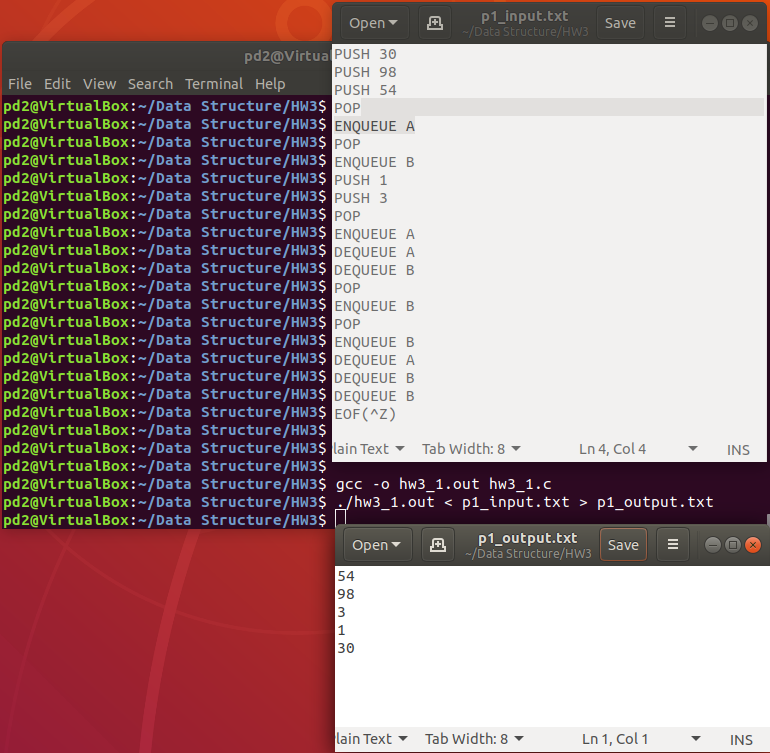
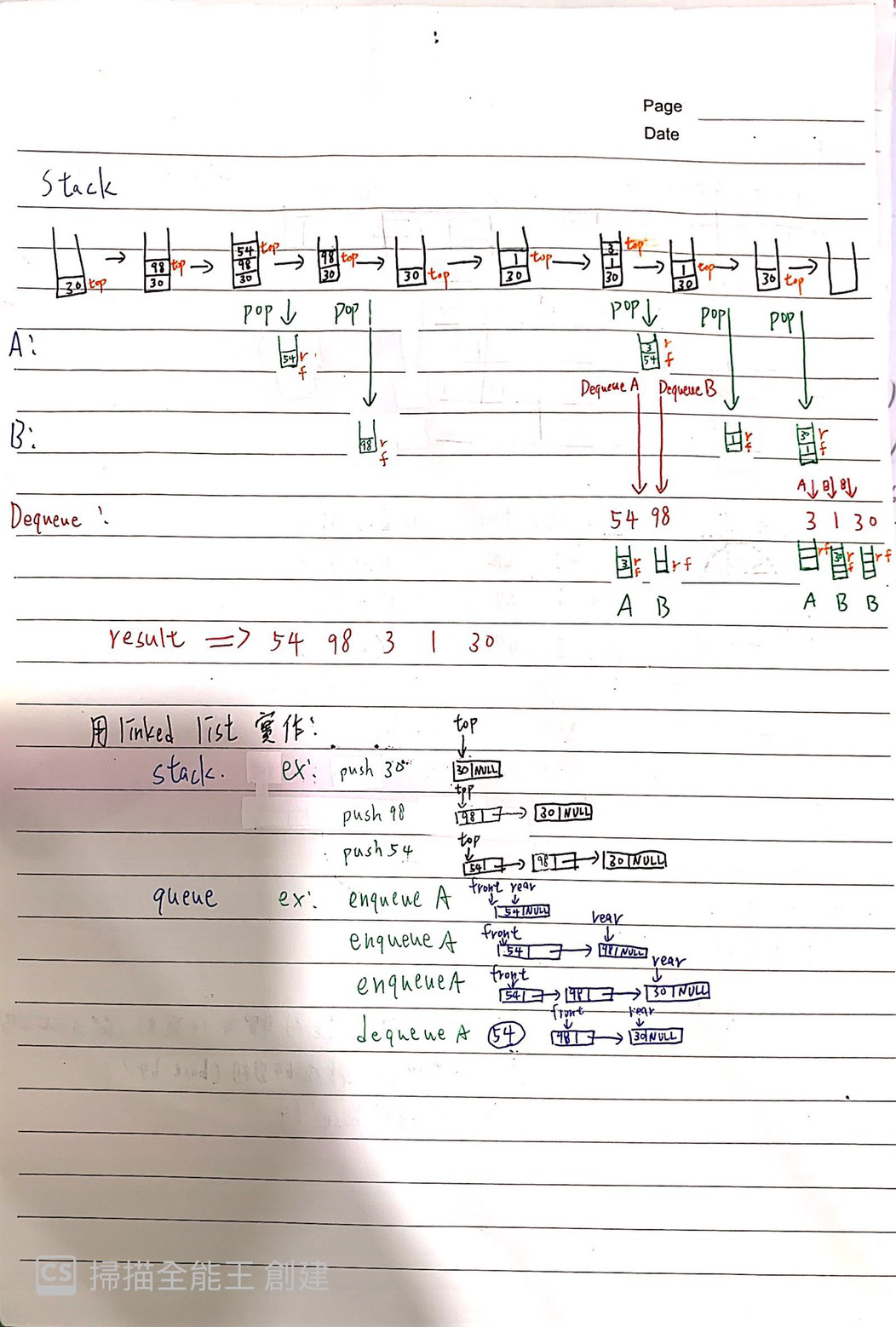
**HW3\_1**

1. Result screenshot



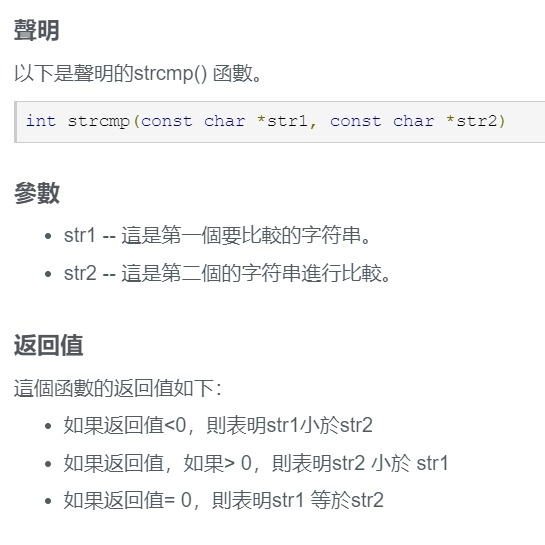
1. Program architecture



3. Program functions

(1)

strcmp():用來比對字串



(2)

void add\_plate(stack\_pointer \*top, int plate):把plate放入stack中

top – 指向stack中最上面那一筆資料的pointer

plate – 要放入的plate的號碼

(3)

int delete\_plate(stack\_pointer \*top):用來取出stack中的plate

top – 指向stack中最上面那一筆資料的pointer

return – 取出的plate的號碼

(4)

void enqueue(queue\_pointer \*front, queue\_pointer \*rear, int data):把客人列入隊伍

\*front – 指向該隊伍(A或B)第一個進來(隊伍最前面)那一筆資料的pointer

\*rear – 指向該隊伍(A或B)最後一個進來(隊伍最前面)那一筆資料的pointer

data – 客人手上拿的plate的號碼

(5)

int dequeue(queue\_pointer \*front):用來取得離開隊伍的客人的plate的號碼

\*front – 指向該隊伍(A或B)第一個進來(隊伍最前面)那一筆資料的pointer

return – 客人放回的plate的號碼

1. How I design my program

利用typedef和struct來建立stack\_pointe、queue\_pointer，來建立能夠存放資料和指向下一筆資料的pointer。

設立stack\_pointer陣列top\_plate來建立一個stack，設立queue\_pointer陣列front\_A、front\_B、rear\_A、rear\_B，

設一個陣列operation來接從input.txt接收到的字串，用strcmp加上if、else if，用類似switch case的方式來判斷現在是要PUSH、POP、ENQUEUE、或是DEQUEUE。

若接收到PUSH，則設一個int變數N來接收後面的號碼，也就是plate的號碼，並且呼叫add\_plate()這個函式，宣告一個新節點temp，把N輸入到temp的data中(temp->data = plate)，之後再透過add\_plate中的操作，把top指向temp。

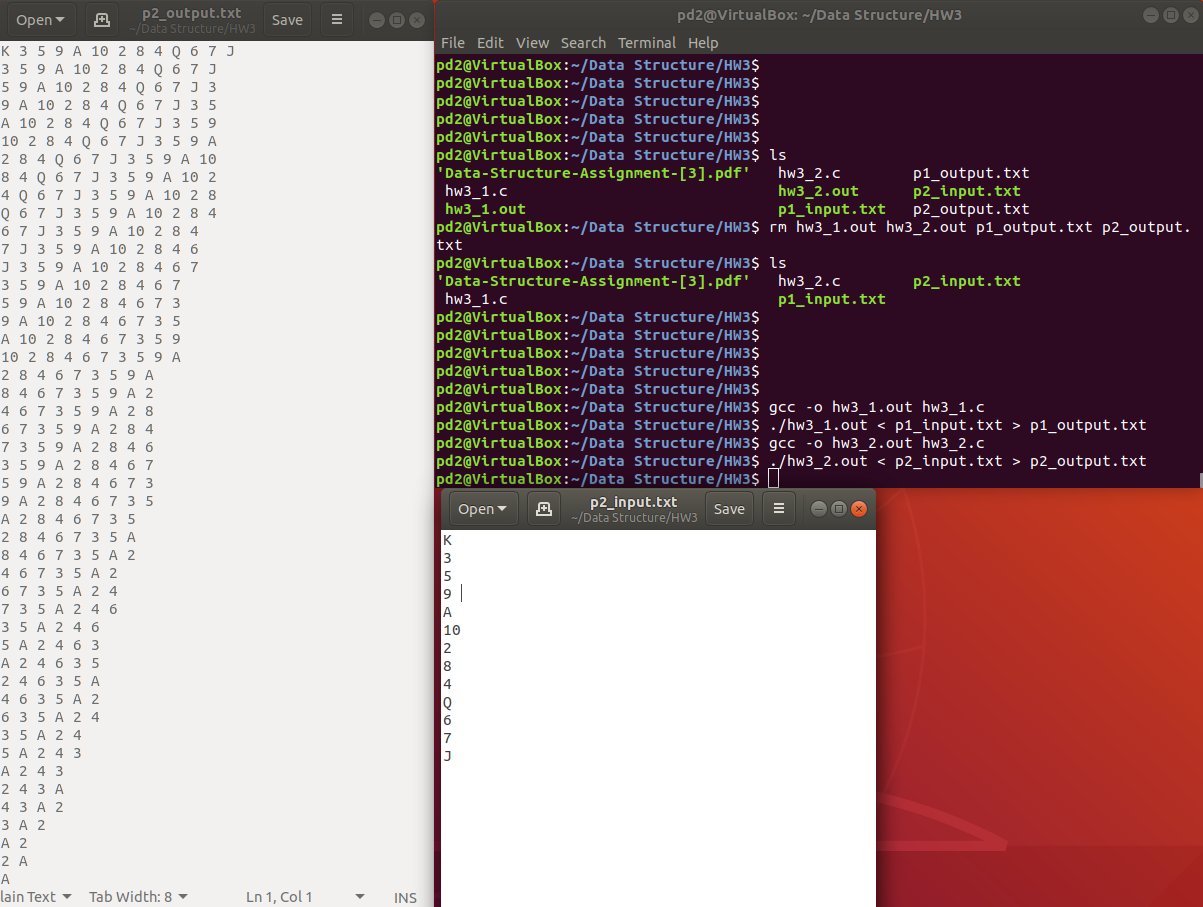
若接收到POP則直接呼叫continue，因為POP後面是接ENQUEUE A或ENQUEUE B，也就是將stack中的N pop進隊伍中，而這一步我直接放到ENQUEUE的地方做，因此POP這邊我直接continue。

若接收到ENQUEUE，先設立一個陣列AorB，並把ENQUEUE後面的字串放入其中，也就是A或是B，並一樣透過strcmp()來判斷是A或B，接著呼叫enqueue()來將stack中位於top所指向的node中的data(也就是N)用來建立一個新的節點，然後把top指向top->next，並把原本top的地方free掉，以免造成memory leak，然後透過enqueue()中的操作，新的節點接在rear所指向的地方，並把front跟rear這兩個pointer指向該指的地方(front指在第一個進來的資料、rear指在最後一個進來的資料)。

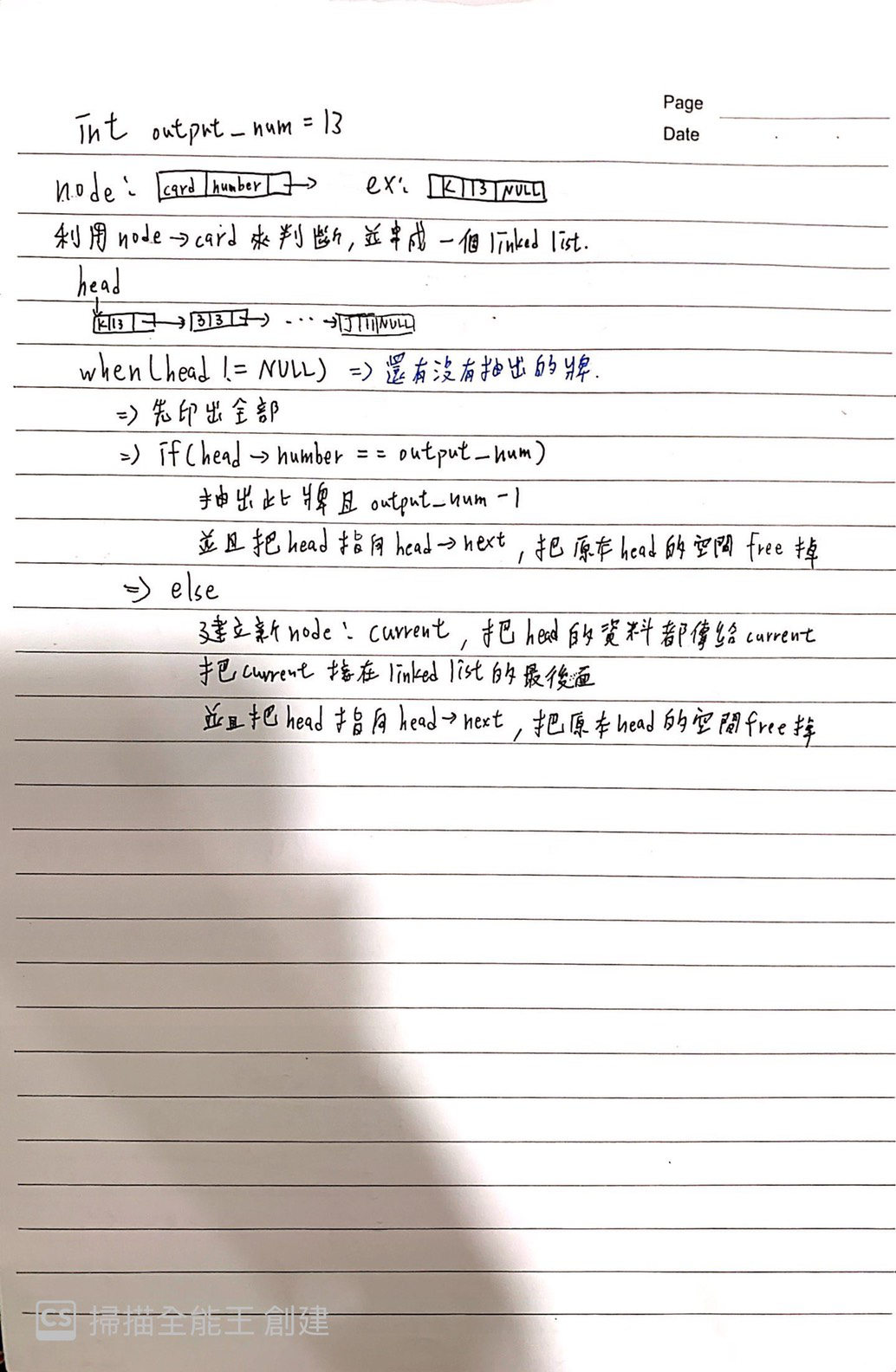
若接收到DEQUEUE，做法和ENQUEUE一樣，先判斷A或B，然後呼叫dequeue()，將返回的結果直接輸出在output.txt中，並且將透過dequeue()中的操作，使front指向front->next，並將原本front指向的那個節點free掉，以免造成memory leak。

**HW3\_2**

1. Result screenshot

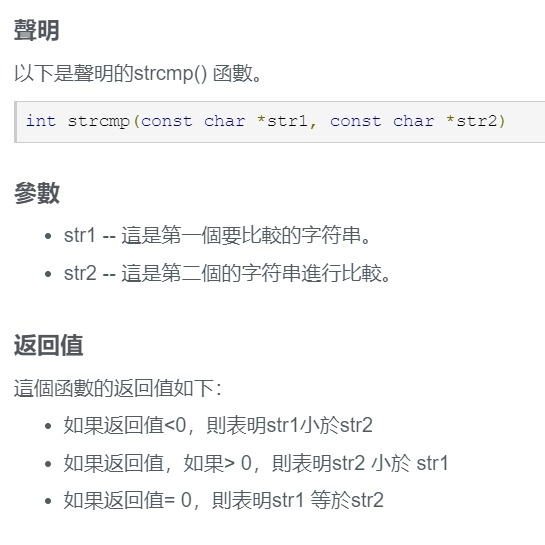


1. Program architecture



1. Program functions

(1) strcmp():用來比對字串



(2) strcpy():用來複製字串



1. How I design my program

首先利用typedef和struct建立節點，一個節點包含一組字串(card)一個數字(number)、和指向下一節點的pointer(next)。

再來先宣告一個integer output\_num=13，用來對之後判斷是否要抽出此牌做準備，並宣告兩個solitaire\_pointer，head和current，head永遠指向第一個節點，而current則是現在要用來判斷讀進來的是哪一張卡片。

利用strcmp()將傳進來的字母(K、Q、J、A)或是數字(2~10)建立成一個一個的節點，利用strcpy()將其字母傳入node->card，而代表的數字則直接傳入node->number(例如K，則strcpy(current->card, “K”)、current->number = 13)，而要注意的是，當head還是空的時候，則head=current，其他則透過while迴圈將current放入linked list的最後面。

再來用一個do while迴圈，先把所有節點中的字串card印出來，然後開始判斷，利用前面的output\_num，如果head指向的資料的數字number == output\_um，則把head指向head->next，並且把原本head的位置free掉，以免造成memory leak，如果number != output\_num，則建立一個新節點current，把head中的card、number都傳給他，然後接在linked list的最後面，之後把head指向head->next，再把head原本的位置free掉，以免造成memory leak。