# Data Structure Assignment 3

|  |  |  |
| --- | --- | --- |
| **ID:** E14066282 | **Name:** 溫梓傑 | **Department:** ME 110 |

## Result Screenshots

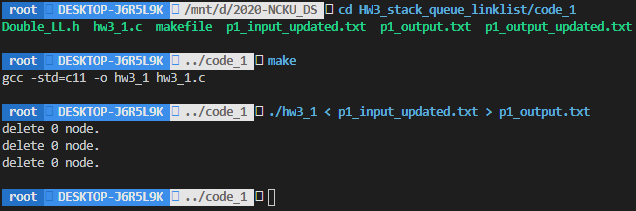


Figure 1 Screenshot of command line (hw3\_1)

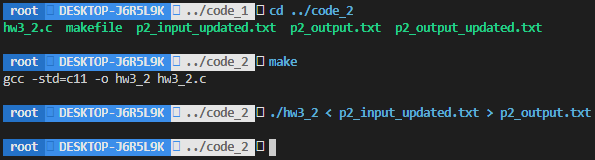


Figure 2 Screenshot of command line (hw3\_2)

|  |  |
| --- | --- |
| Figure 3 p1\_output.txt | Figure 4 p2\_output.txt |

## Program Architecture

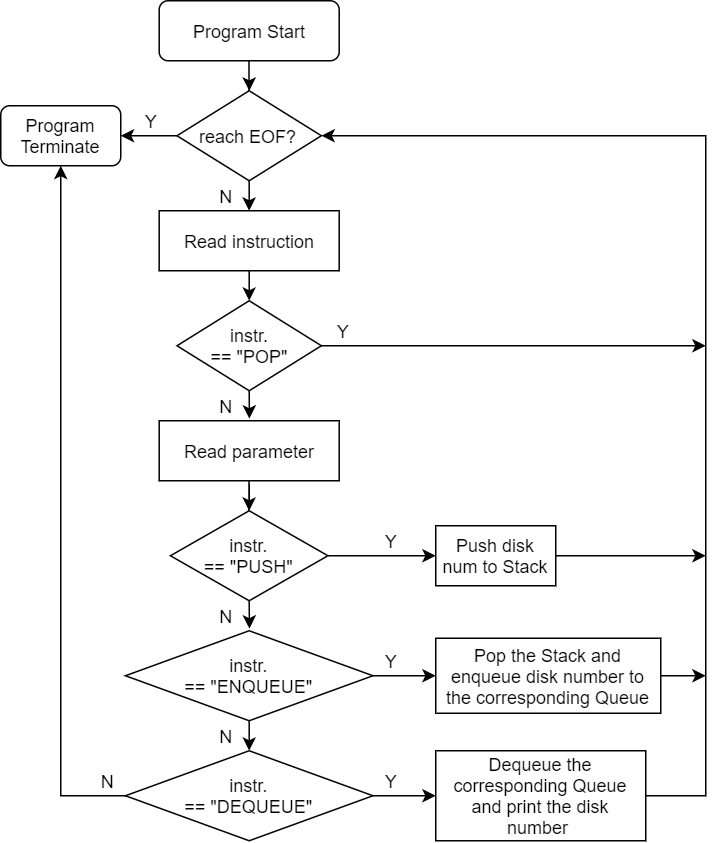


Figure 5 Flow chart of hw3\_1

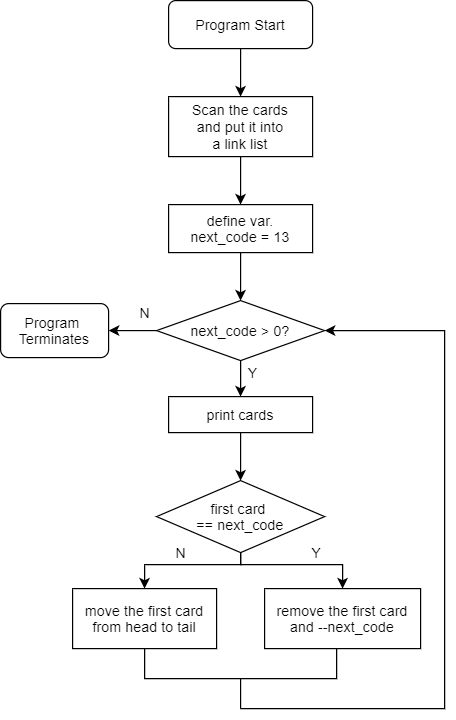


Figure 6 Flow chart of hw3\_2

## Program Functions

📒 Double\_LL.h

LinkList \*create\_ll();

Constructs a link list.

### 📐Parameters

None.

### ↩Return Value

Returns the new pointer of the link list.

* If construction fails, returns NULL.

Node \*create\_node(int val);

Constructs a node.

### 📐Parameters

val

The element that would be initialized in the constructed node.

### ↩Return Value

Returns the new pointer of the node.

* If construction fails, returns NULL.

**void** push\_node(LinkList \*lp, Node \*np);

Inserts the node on the back of the link list.

### 📐Parameters

lp

The pointer of the link list.

np

The pointer of the node.

### ↩Return Value

None.

**int** pop\_node(LinkList \*lp);

Removes the node on the back of the link list.

### 📐Parameters

lp

The pointer of the link list.

### ↩Return Value

Returns the back element before removal.

* If the link list is empty, program terminates.

**void** push\_front\_node(LinkList \*lp, Node \*np)

Inserts the node at the front of the link list.

### 📐Parameters

lp

The pointer of the link list.

np

The pointer of the node.

### ↩Return Value

None.

**int** pop\_front\_node(LinkList \*lp);

Removes the node at the front of the link list.

### 📐Parameters

lp

The pointer of the link list.

### ↩Return Value

Returns the front element before removal.

* If the link list is empty, program terminates.

**void** free\_LL(LinkList \*lp);

Free all nodes in the link list.

### 📐Parameters

lp

The pointer of the link list.

### ↩Return Value

None.

## Program Design

本作業承接上個作業的精神—物件導向來實作，為了實作doubly link list，我使用兩個 struct來實現，一個是struct Node另一個是struct LinkList，其宣告細節如下所示：

**typedef** **struct** Node

{

**struct** Node \*next;

**struct** Node \*prev;

**int** val;

} Node;

**typedef** **struct** LinkList

{

    Node \*head;

    Node \*tail;

} LinkList;

使用doubly link list的好處是，可以在任意節點往前或往後移動。

另外，我在LinkList中，另外定義了末端節點指標，如此便可以直接存取最後一個Node，節省push以及pop的時間。

## Operating System

Ubuntu 20.04.1 LTS (Focal Fossa)

## Compiler

gcc (Ubuntu 9.3.0-10ubuntu2) 9.3.0

## Compile

make

* ⚠️Notice

hw3\_2.c會使用到../code\_1/Double\_LL.h，編譯hw3\_2.c前，請確認code\_1目錄存在！

## Run

### 📄hw3\_1:

./hw3\_1 < input.txt > output.txt

### 📄hw3\_2:

./hw3\_2 < input.txt > output.txt