Lab 5: Linked list

1. Write a program where you have to create a linked list with 6 nodes.
   a. Get 6 integers from user and insert it into the linked list.
   b. Then display the list.
   c. After displaying the original list, delete the first node and display the list again.
   d. Prompt the user to enter a data to search in the list. If the data is in the list, display the message ***"DATA FOUND!".*** If not, display the message ***"DATA NOT FOUND!"*** Below is the sample screen of the output: -

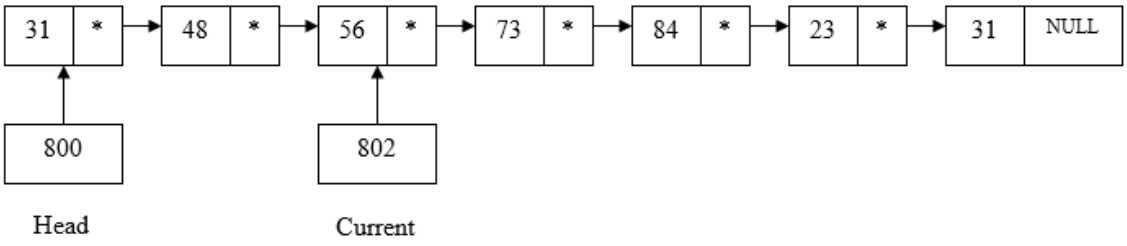
Sample Output:

```
Enter data 1:99
Enter data 2:66
Enter data 3:44
Enter data 4:55
Enter data 5:77
The Current List:
99 66 44 55 77

Deleting the first node
The list after deletion:
66 44 55 77
Enter a data to search:44
DATA FOUND!
```

ANS: REFER CODE Lab 5 Q1

2. Refer to figure below:



Complete the table below by writing the value for each of the following statements:

	Value
Head	800
Head→Next	801
Head→Next→Next→Next→Data	84
Current→Next→Data	84
Current→Next→Next→Next→Next	805
Current→Next→Next→Data	23
Current→Next→Next→Next→Next→Data	31

3. Table below represents an array implementation of a linked list.

Index	Data	Link
0	65	2
1	78	4
2	33	5
3	44	99
4	55	3
5	39	6
6	52	1

i) Taking 0 as the start of the list and 99 as a dummy representing the end of the file, fill in the link for all the elements in table to maintain a list of the following order.

65 33 39 52 78 55 44

Solution

- Index 0 contains 65, link to 33 (2), so move to index **2**.
- Index 2 contains 33, link to 39 (5), so move to index **5**.
- Index 5 contains 39, link to 52 (6), so move to index **6**.
- Index 6 contains 52, link to 78 (1), so move to index **1**.
- Index 1 contains 78, link to 55 (4), so move to index **4.**
- Index 4 contains 55, linked to 44(3), so move to index **3**.
- Index 3 contains 44, linked to **99**, it's the end of the order.

ii) Based on the answer in question b (i), draw an updated table after data 40 is added between data 39 and 52.

Index	Data	Link
0	65	2
1	78	4
2	33	5
3	44	99
4	55	3
5	39	7
6	52	1
7	40	6

65 33 39 **40(added)** 52 78 55 44

Solution

- Index 0 contains 65, link to 33 (2), so move to index **2**.
- Index 2 contains 33, link to 39 (5), so move to index **5**.
- Index 5 contains 39, link to 40 (7), so move to index 7.
- Index 8 contains 40, link to 52(6), so move to index 6.
- Index 6 contains 52, link to 78 (1), so move to index 1.
- Index 1 contains 78, link to 55(4), so move to index 4.
- Index 4 contains 55, link to 44(3), so move to index 3.
- Index 3 contains 44, linked to **99**, it's the end of the order.

iii) Based on answers in question b(ii), draw an updated table after data **78 is deleted.**

Index	Data	Link
0	65	2
1	78	[DELETED]
2	33	5
3	44	99
4	55	3
5	39	7
6	52	4
7	40	6

65 33 39 40 52 55 44

Solution

- Index 0 contains 65, link to 33 (2), so move to index **2**.
- Index 2 contains 33, link to 39 (5), so move to index **5**.
- Index 5 contains 39, link to 40 (7), so move to index 7.
- Index 7 contains 40, link to 52(6), so move to index 6.
- Index 6 contains 52, link to 55 (3), so move to index 4.
- Index 4 contains 55, link to 44(3), so move to index 3.
- Index 3 contains 44, linked to **99**, it's the end of the order.
- Index 1 is deleted at 78 is deleted

Submission question

Show and draw the trace diagram from the following C++ codes. Assume that the node consists of two members, data and next with the data of the type int (list and ptr are pointers of the type node).

i) `ptr=new node;`

- **ptr - - > [|NULL]**

ii) `ptr->data=28;`

- **ptr - - > [28 |NULL]**

iii) `ptr->next=NULL;`

- **ptr - - > [28 | NULL]**

iv) `list = new node;`

- **ptr - - > [28 | NULL]**
- **list - - > [|NULL]**

v) `list->data=56;`

- **ptr - - > [28 | NULL]**
- **list - - > [56 | NULL]**

vi) `list->next=ptr;`

- **ptr - - > [28 | NULL]**
- **list - - > [56 | ptr] -- > [28 | NULL]**

vii) `ptr=new node;`

- **ptr - - > [|NULL]**
- **list - - > [56 | ptr] -- > [28 | NULL]**

viii) ptr->data=68;

- **ptr - - > [68 | NULL]**
- **list - - > [56 | ptr] -- > [28 | NULL]**
-

ix) ptr->next=list;

- **ptr --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

x) list=ptr;

- **ptr --> [62 | NULL]**
- **list --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xi) ptr=new node;

- **ptr --> [62 | ptr->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**
- **list --> [62 | ptr->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xii) ptr->data=62;

- **ptr --> [62 | ptr->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**
- **list --> [62 | ptr->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xiii) ptr->next=list->next;

- **ptr --> [70 | NULL]**
- **list --> [62 | ptr->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xiv) list->next=ptr;

- **ptr --> [70 | NULL] --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr**

xv) ptr=list;

- **ptr --> [70 | ptr->next] --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr**

xvi) ptr=new node;

- **ptr --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**
- **list --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xvii) ptr->data=70;

- **ptr --> [70 | ptr->next] --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr]**

xviii) ptr->next=list->next;

- **ptr --> [70 | ptr->next] --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**
- **list --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xix) list->next=ptr;

- **ptr --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**
- **list --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xx)
```
ptr=list;
```
- **ptr --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**
- **list --> [62 | ptr->next->next] --> [68 | list] --> [56 | ptr] --> [28 | NULL]**

xxi)
```
while(ptr!=NULL)
{
  cout<<ptr->data<<endl;
  ptr=ptr->next;
}
```
- **62**
- **68**
- **56**
- **28**