UNIVERSITY OF ENGINEERING AND TECHNOLOGY MARDAN

Name

Abdul Haseeb

Reg. No

22MDSWE197

Class. No

4

DSA Lab.

6

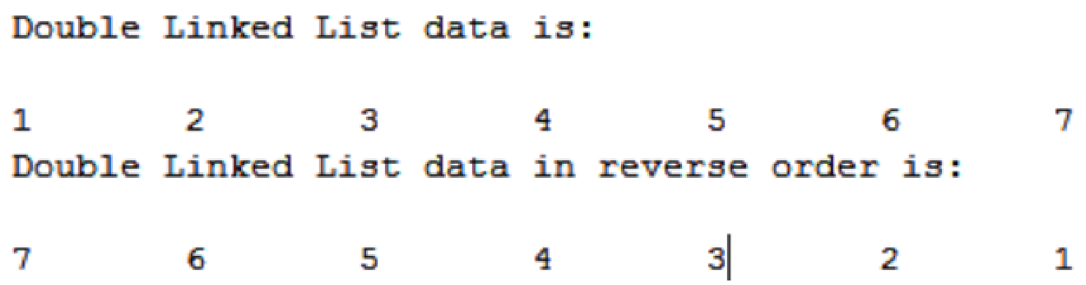
Submitted To.

Engr. Sohail

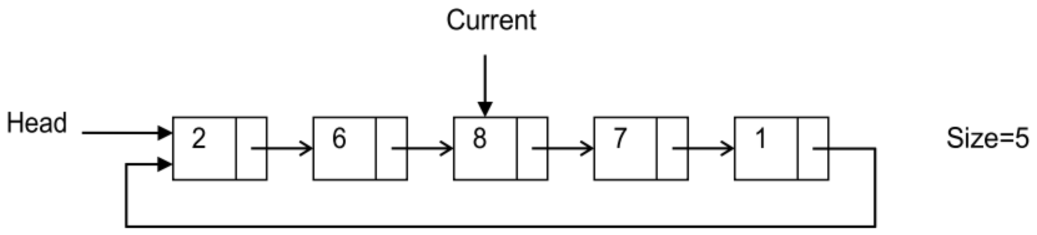
1. Modify the Doubly linked list to create a Doubly Circular Linked List. Update all the functions accordingly for Insertion, Deletion and Display.

2. Write a function to display size of doubly linked list in task 1 and display all the elements in reverse order

3. Add Search function to the Doubly Circular Linked List while will receive a parameter containing the value that will be searched in the list. Do implement a stopping mechanism as there will be no NULL link in the circular list

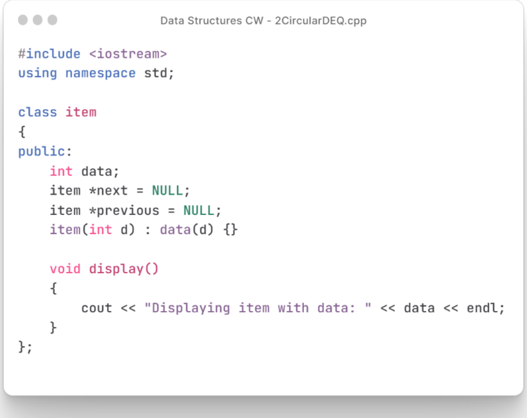


```
Double Linked List data is:

1       2       3       4       5       6       7
Double Linked List data in reverse order is:

7       6       5       4       3       2       1
```

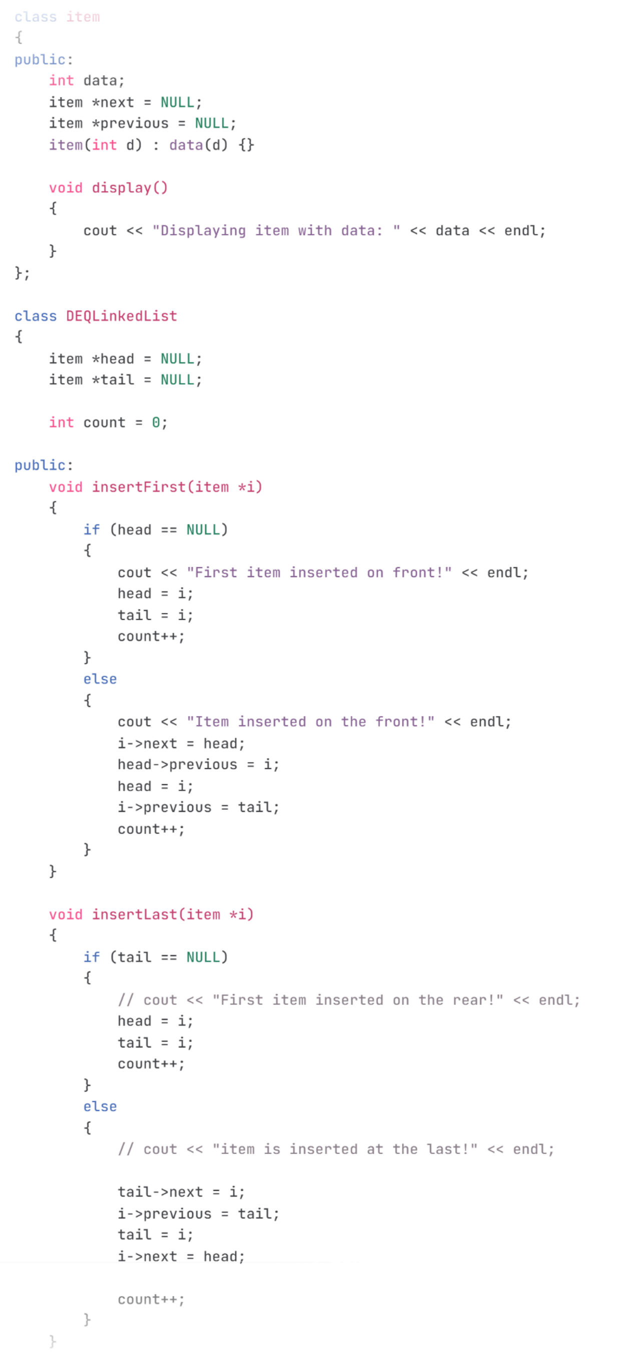


1. Modify the Doubly linked list to create a Doubly Circular Linked List. Update all the functions accordingly for Insertion, Deletion and Display.

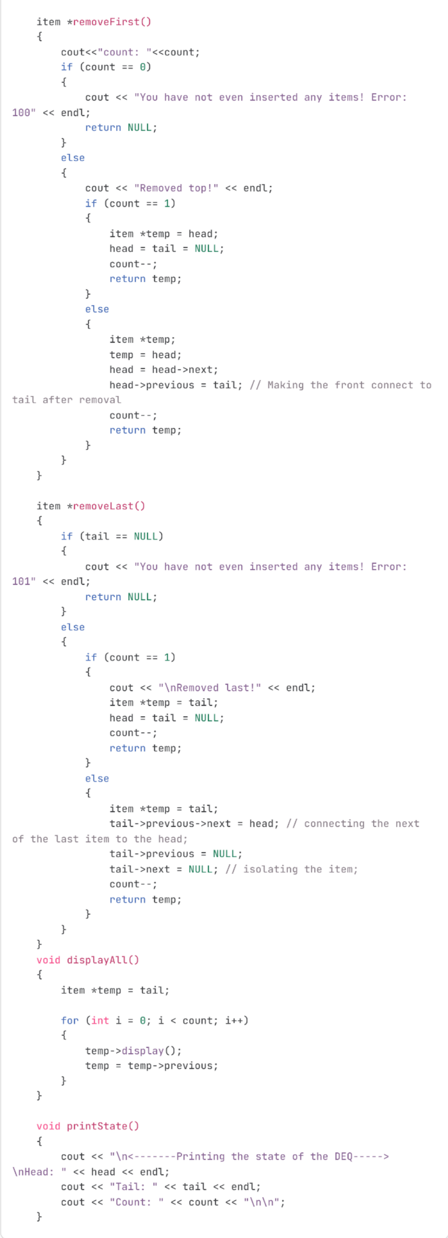
Doubly Circular Linked List :



```cpp
#include <iostream>
using namespace std;

class item
{
public:
    int data;
    item *next = NULL;
    item *previous = NULL;
    item(int d) : data(d) {}

    void display()
    {
        cout << "Displaying item with data: " << data << endl;
    }
};
```

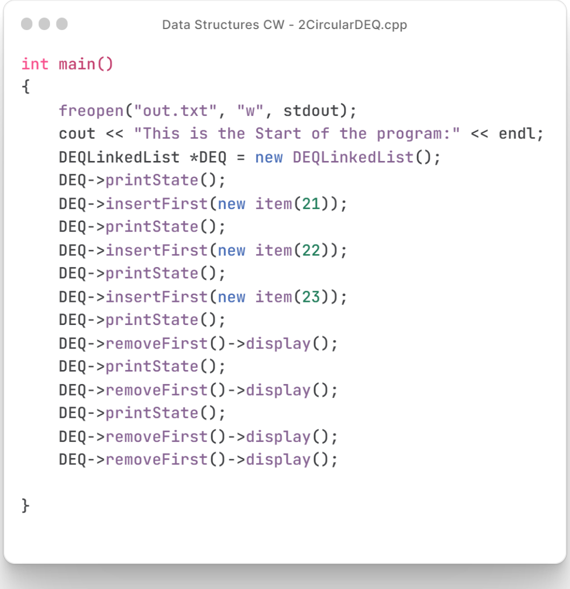


```cpp
class item
{
public:
    int data;
    item *next = NULL;
    item *previous = NULL;
    item(int d) : data(d) {}

    void display()
    {
        cout << "Displaying item with data: " << data << endl;
    }
};

class DEQLinkedList
{
    item *head = NULL;
    item *tail = NULL;

    int count = 0;

public:
    void insertFirst(item *i)
    {
        if (head == NULL)
        {
            cout << "First item inserted on front!" << endl;
            head = i;
            tail = i;
            count++;
        }
        else
        {
            cout << "Item inserted on the front!" << endl;
            i->next = head;
            head->previous = i;
            head = i;
            i->previous = tail;
            count++;
        }
    }

    void insertLast(item *i)
    {
        if (tail == NULL)
        {
            // cout << "First item inserted on the rear!" << endl;
            head = i;
            tail = i;
            count++;
        }
        else
        {
            // cout << "item is inserted at the last!" << endl;

            tail->next = i;
            i->previous = tail;
            tail = i;
            i->next = head;

            count++;
        }
    }
```

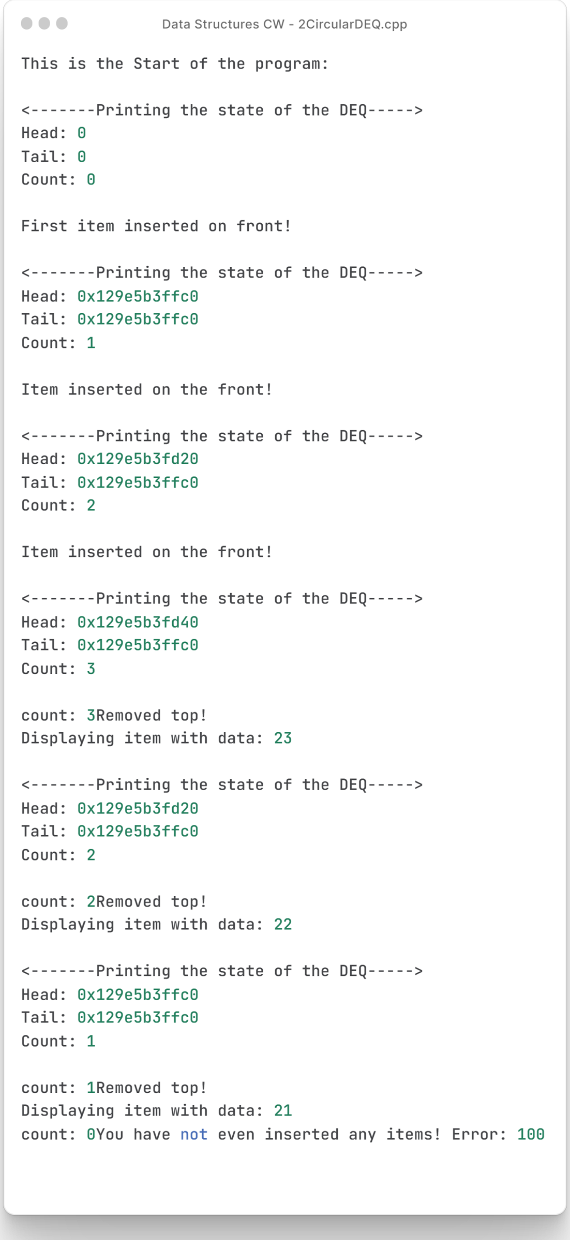


```cpp
    item *removeFirst()
    {
        cout<<"count: "<<count;
        if (count == 0)
        {
            cout << "You have not even inserted any items! Error:
100" << endl;
            return NULL;
        }
        else
        {
            cout << "Removed top!" << endl;
            if (count == 1)
            {
                item *temp = head;
                head = tail = NULL;
                count--;
                return temp;
            }
            else
            {
                item *temp;
                temp = head;
                head = head->next;
                head->previous = tail; // Making the front connect to
tail after removal
                count--;
                return temp;
            }
        }
    }

    item *removeLast()
    {
        if (tail == NULL)
        {
            cout << "You have not even inserted any items! Error:
101" << endl;
            return NULL;
        }
        else
        {
            if (count == 1)
            {
                cout << "\nRemoved last!" << endl;
                item *temp = tail;
                head = tail = NULL;
                count--;
                return temp;
            }
            else
            {
                item *temp = tail;
                tail->previous->next = head; // connecting the next
of the last item to the head;
                tail->previous = NULL;
                tail->next = NULL; // isolating the item;
                count--;
                return temp;
            }
        }
    }
    void displayAll()
    {
        item *temp = tail;

        for (int i = 0; i < count; i++)
        {
            temp->display();
            temp = temp->previous;
        }
    }

    void printState()
    {
        cout << "\n<-------Printing the state of the DEQ----->
\nHead: " << head << endl;
        cout << "Tail: " << tail << endl;
        cout << "Count: " << count << "\n\n";
    }
```

Output



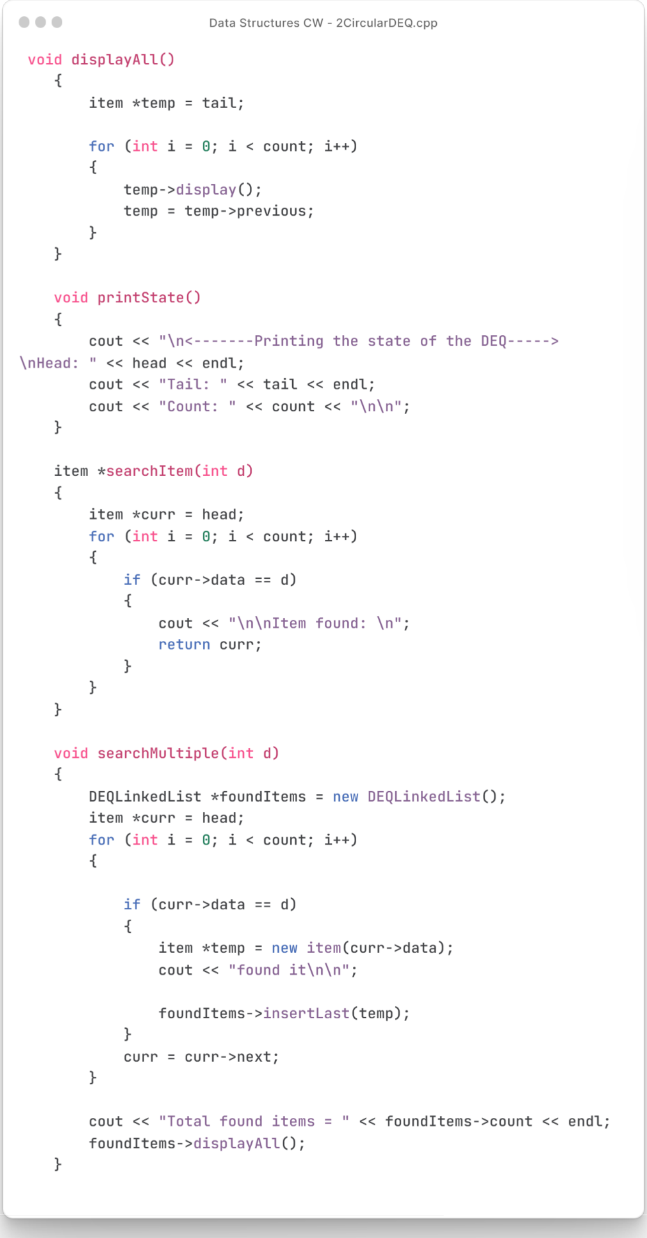
```
int main()
{
    freopen("out.txt", "w", stdout);
    cout << "This is the Start of the program:" << endl;
    DEQLinkedList *DEQ = new DEQLinkedList();
    DEQ->printState();
    DEQ->insertFirst(new item(21));
    DEQ->printState();
    DEQ->insertFirst(new item(22));
    DEQ->printState();
    DEQ->insertFirst(new item(23));
    DEQ->printState();
    DEQ->removeFirst()->display();
    DEQ->printState();
    DEQ->removeFirst()->display();
    DEQ->printState();
    DEQ->removeFirst()->display();
    DEQ->removeFirst()->display();

}
```



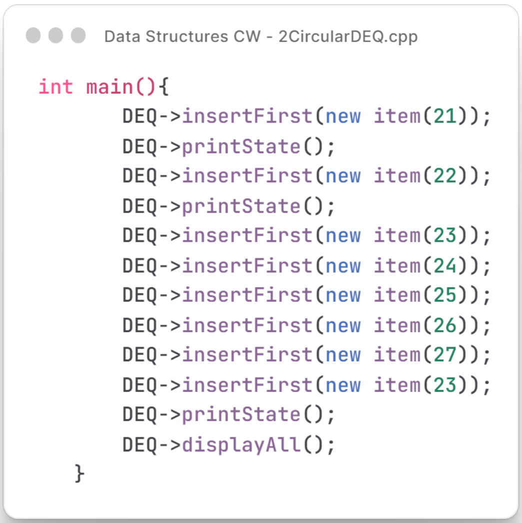
```
This is the Start of the program:

<-------Printing the state of the DEQ----->
Head: 0
Tail: 0
Count: 0

First item inserted on front!

<-------Printing the state of the DEQ----->
Head: 0x129e5b3ffc0
Tail: 0x129e5b3ffc0
Count: 1

Item inserted on the front!

<-------Printing the state of the DEQ----->
Head: 0x129e5b3fd20
Tail: 0x129e5b3ffc0
Count: 2

Item inserted on the front!

<-------Printing the state of the DEQ----->
Head: 0x129e5b3fd40
Tail: 0x129e5b3ffc0
Count: 3

count: 3Removed top!
Displaying item with data: 23

<-------Printing the state of the DEQ----->
Head: 0x129e5b3fd20
Tail: 0x129e5b3ffc0
Count: 2

count: 2Removed top!
Displaying item with data: 22

<-------Printing the state of the DEQ----->
Head: 0x129e5b3ffc0
Tail: 0x129e5b3ffc0
Count: 1

count: 1Removed top!
Displaying item with data: 21
count: 0You have not even inserted any items! Error: 100
```

2. Write a function to display size of doubly linked list in task 1 and display all the elements in reverse order

Displaying size and Elements :

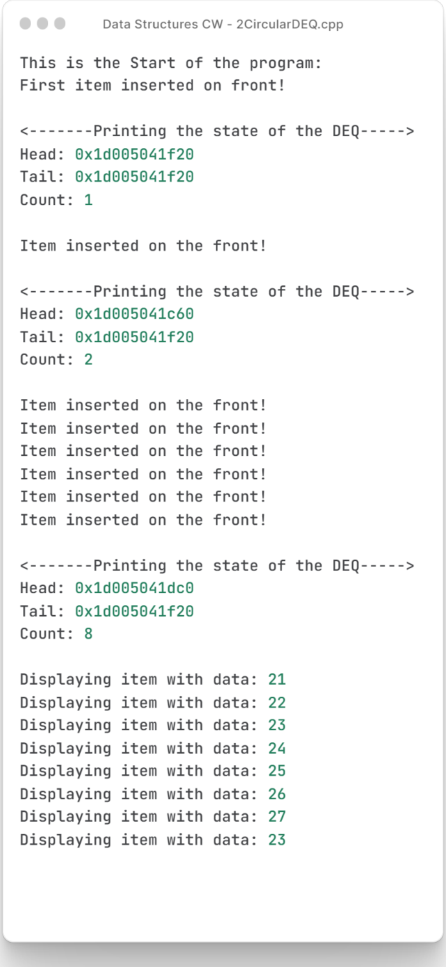


```
 void displayAll()
    {
        item *temp = tail;

        for (int i = 0; i < count; i++)
        {
            temp->display();
            temp = temp->previous;
        }
    }

    void printState()
    {
        cout << "\n<-------Printing the state of the DEQ----->
\nHead: " << head << endl;
        cout << "Tail: " << tail << endl;
        cout << "Count: " << count << "\n\n";
    }

    item *searchItem(int d)
    {
        item *curr = head;
        for (int i = 0; i < count; i++)
        {
            if (curr->data == d)
            {
                cout << "\n\nItem found: \n";
                return curr;
            }
        }
    }

    void searchMultiple(int d)
    {
        DEQLinkedList *foundItems = new DEQLinkedList();
        item *curr = head;
        for (int i = 0; i < count; i++)
        {

            if (curr->data == d)
            {
                item *temp = new item(curr->data);
                cout << "found it\n\n";

                foundItems->insertLast(temp);
            }
            curr = curr->next;
        }

        cout << "Total found items = " << foundItems->count << endl;
        foundItems->displayAll();
    }
```



```
int main(){
        DEQ->insertFirst(new item(21));
        DEQ->printState();
        DEQ->insertFirst(new item(22));
        DEQ->printState();
        DEQ->insertFirst(new item(23));
        DEQ->insertFirst(new item(24));
        DEQ->insertFirst(new item(25));
        DEQ->insertFirst(new item(26));
        DEQ->insertFirst(new item(27));
        DEQ->insertFirst(new item(23));
        DEQ->printState();
        DEQ->displayAll();
    }
```

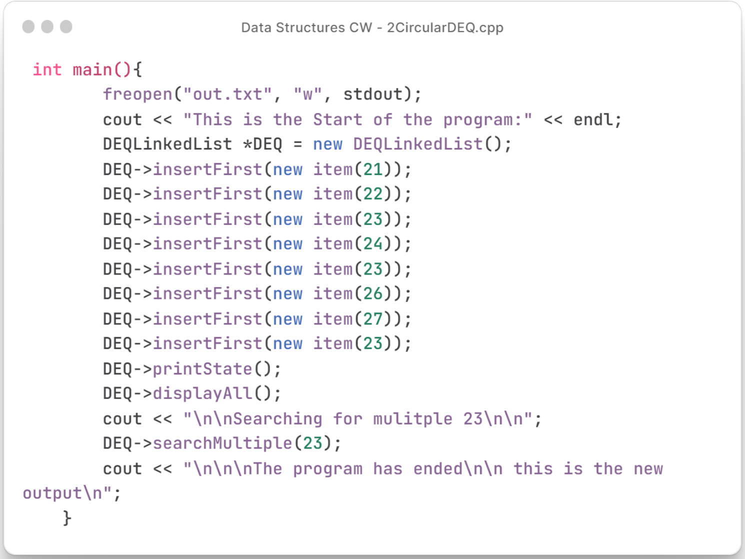
Output:



```
This is the Start of the program:
First item inserted on front!

<-------Printing the state of the DEQ----->
Head: 0x1d005041f20
Tail: 0x1d005041f20
Count: 1

Item inserted on the front!

<-------Printing the state of the DEQ----->
Head: 0x1d005041c60
Tail: 0x1d005041f20
Count: 2

Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!

<-------Printing the state of the DEQ----->
Head: 0x1d005041dc0
Tail: 0x1d005041f20
Count: 8

Displaying item with data: 21
Displaying item with data: 22
Displaying item with data: 23
Displaying item with data: 24
Displaying item with data: 25
Displaying item with data: 26
Displaying item with data: 27
Displaying item with data: 23
```

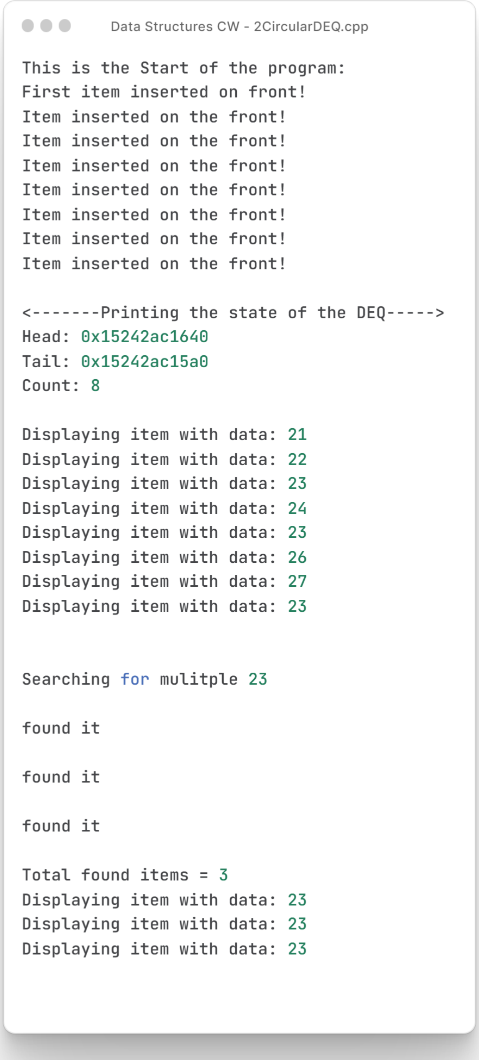
3. Add Search function to the Doubly Circular Linked List while will receive a parameter containing the value that will be searched in the list. Do implement a stopping mechanism as there will be no NULL link in the circular list

Searching for Elements:



Data Structures CW - 2CircularDEQ.cpp

```
 int main(){
        freopen("out.txt", "w", stdout);
        cout << "This is the Start of the program:" << endl;
        DEQLinkedList *DEQ = new DEQLinkedList();
        DEQ->insertFirst(new item(21));
        DEQ->insertFirst(new item(22));
        DEQ->insertFirst(new item(23));
        DEQ->insertFirst(new item(24));
        DEQ->insertFirst(new item(23));
        DEQ->insertFirst(new item(26));
        DEQ->insertFirst(new item(27));
        DEQ->insertFirst(new item(23));
        DEQ->printState();
        DEQ->displayAll();
        cout << "\n\nSearching for mulitple 23\n\n";
        DEQ->searchMultiple(23);
        cout << "\n\n\nThe program has ended\n\n this is the new
output\n";
    }
```



Data Structures CW - 2CircularDEQ.cpp

```
This is the Start of the program:
First item inserted on front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!
Item inserted on the front!

<-------Printing the state of the DEQ----->
Head: 0x15242ac1640
Tail: 0x15242ac15a0
Count: 8

Displaying item with data: 21
Displaying item with data: 22
Displaying item with data: 23
Displaying item with data: 24
Displaying item with data: 23
Displaying item with data: 26
Displaying item with data: 27
Displaying item with data: 23


Searching for mulitple 23

found it

found it

found it

Total found items = 3
Displaying item with data: 23
Displaying item with data: 23
Displaying item with data: 23
```

UNIVERSITY OF ENGINEERING AND TECHNOLOGY MARDAN

ABDUL HASEEB

REGISTRATION NO : 22MDSWE197

DEPARTMENT : COMPUTER SOFTWARE ENGINEERING

BATCH : BATCH_05

SEMESTER : 3RD SEMESTER (MID)

DATE : 19 - 10 – 2023

DSA [LAB 6]

SUBJECT : DATA STRUCTURES AND ALGORITHMS

TITLE : DOUBLY CIRCULAR LINKED LIST

SUBMITTED TO : ENGR. SOHAIL SIR