PIMPRI CHINCHWAD EDUCATION TRUST's.

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

(An Autonomous Institute)

Class : SY BTech **Acad. Yr. 2025-26** **Semester :** I

Name of the student: Samruddhi Ramswami Bansode. **PRN :** 124B1B019

Department: Computer Engineering **Division :** A

Course Name : Data Structures

Course Code: BCE23PC02

Completion Date : 25/08/2025

Assignment No. 7

Problem Statement:

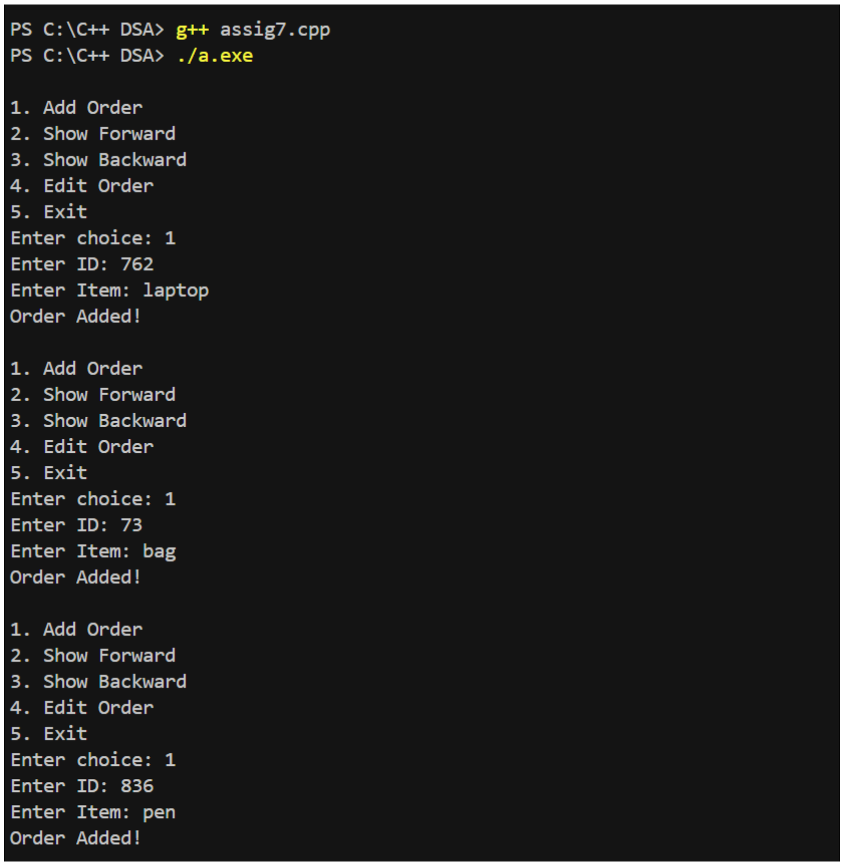
circular order history CDLL

create an order history system where the orders are stored in circular structure allowing navigation from the latest back to the earliest and vice versa.
key operation :
- add orders
- travers forward backward
- edit the orders

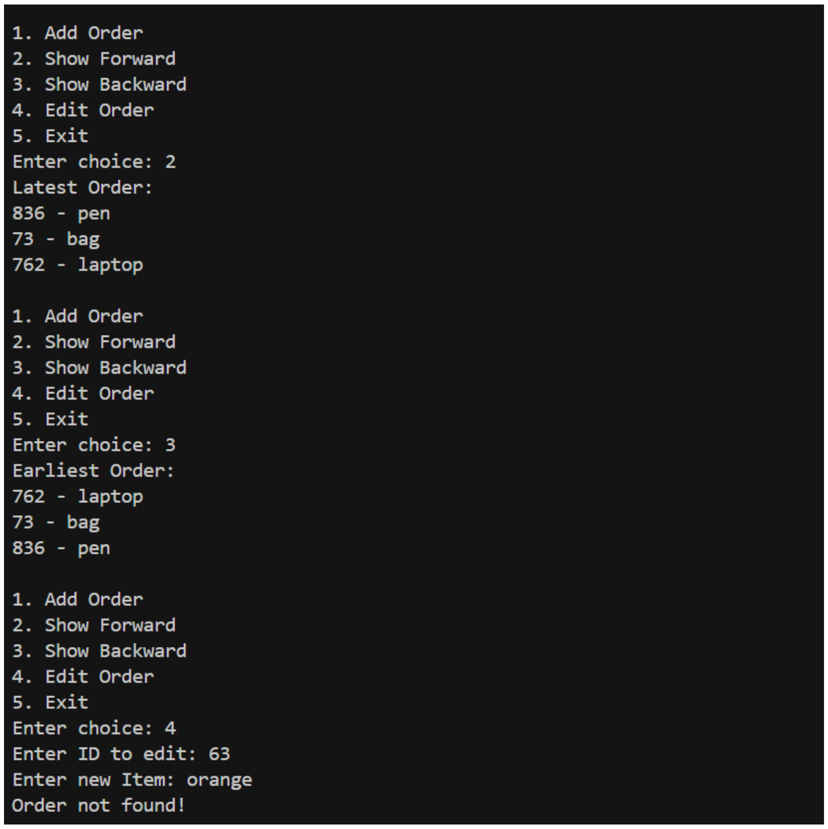
Source Code :

```
#include<iostream>

#include<string>

using namespace std;

struct Node {

    int id;

    string item;

    Node* next;
    Node* prev;

};


class OrderHistory {

    Node* head;

public:

    OrderHistory() { head = NULL; }

    // Add order

    void addOrder(int id, string item) {

        Node* n = new Node;

        n->id = id;

        n->item = item;

        if (head == NULL) {

            head = n;

            n->next = n;

            n->prev = n;

        } else {

            Node* last = head->prev;

            n->next = head;

            n->prev = last;

            last->next = n;

            head->prev = n;

            head = n;

        }

        cout << "Order Added!\n";

    }

    // Show orders forward
    void showForward() {
        if (head == NULL) { cout << "No orders!\n"; return; }
        Node* temp = head;
        cout << "Latest Order:\n";
        do {
            cout << temp->id << " - " << temp->item << endl;
            temp = temp->next;
        } while (temp != head);
    }
    // Show orders backward
    void showBackward() {
        if (head == NULL) { cout << "No orders!\n"; return; }
        Node* temp = head->prev;
        cout << "Earliest Order:\n";
        do {
            cout << temp->id << " - " << temp->item << endl;
            temp = temp->prev;
        } while (temp != head->prev);
    }
    // Edit order
    void editOrder(int id, string newItem) {
        if (head == NULL) { cout << "No orders!\n"; return; }
        Node* temp = head;
        do {
            if (temp->id == id) {
                temp->item = newItem;
                cout << "Order Updated!\n";
                return;

            }

            temp = temp->next;

        } while (temp != head);

        cout << "Order not found!\n";

    }

};


int main() {

    OrderHistory oh;

    int ch, id;

    string item;


    while (1) {

           cout << "\n1. Add Order\n2. Show Forward\n3. Show Backward\n4. Edit
Order\n5. Exit\nEnter choice: ";

        cin >> ch;

        switch (ch) {

            case 1:

                cout << "Enter ID: "; cin >> id;

                cout << "Enter Item: "; cin >> item;

                oh.addOrder(id, item);

                break;

            case 2: oh.showForward(); break;

            case 3: oh.showBackward(); break;

            case 4:

                cout << "Enter ID to edit: "; cin >> id;
                cout << "Enter new Item: "; cin >> item;

                oh.editOrder(id, item);

                break;

            case 5: return 0;

            default: cout << "Invalid choice!\n";

        }

    }

    return 0;

}
```

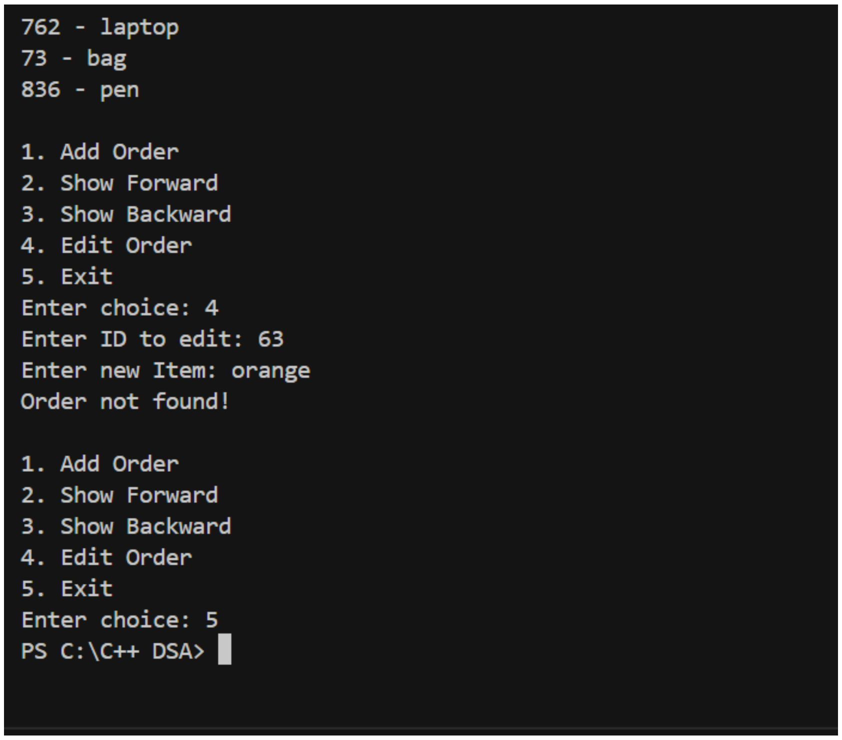
Screen Shot of Output :



```
PS C:\C++ DSA> g++ assig7.cpp
PS C:\C++ DSA> ./a.exe

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 1
Enter ID: 762
Enter Item: laptop
Order Added!

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 1
Enter ID: 73
Enter Item: bag
Order Added!

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 1
Enter ID: 836
Enter Item: pen
Order Added!
```



```
1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 2
Latest Order:
836 - pen
73 - bag
762 - laptop

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 3
Earliest Order:
762 - laptop
73 - bag
836 - pen

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 4
Enter ID to edit: 63
Enter new Item: orange
Order not found!
```



```
762 - laptop
73 - bag
836 - pen

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 4
Enter ID to edit: 63
Enter new Item: orange
Order not found!

1. Add Order
2. Show Forward
3. Show Backward
4. Edit Order
5. Exit
Enter choice: 5
PS C:\C++ DSA>
```

Conclusion:

In this assignment, we built a simple order history system using a **Circular Doubly Linked List**. It lets us move through orders both forward and backward, just like scrolling through order history in real apps. We can easily add new orders and even update existing ones. This shows how CDLL can be a practical way to manage data where continuous navigation is needed, such as in shopping or browsing history.