Lab Report - 07

Problem statement : Write a menu program to create and manipulate Stack using stack class and perform the following operations using the specified method.

Source Code :

```
#include<bits/stdc++.h>
using namespace std;

class Menu{
private :
    stack<int> st;
public :
    void Push(){
        int x;
        cout << "Enter a value to push : ";
        cin >> x;
        st.push(x);
        cout << "Value " << x << " push to the stack.\n";
    }
    void Pop(){
        if (!st.empty()){
            st.pop();
            cout << "The last value of the stack is poped.\n";
        }
        else cout << "The stack is already empty.\n";
    }
    void Display(){
        cout << "Stack = ";
        if (st.empty()){
            cout << "Empty.\n";
            return;
        }
        while(!st.empty()){
            cout << st.top() << ' ';
            st.pop();
        }
        cout << '\n';
    }
};
```

```cpp
int main(void){
    Menu mu;

    int option;
    while (4){
        cout << "\t*** Stack ***\n";
        cout << "1. Push.\n";
        cout << "2. Pop.\n";
        cout << "3. Display.\n";
        cout << "4. Exit.\n";

        cout << "\nEnter a option = ";
        cin >> option;
        while (option < 1 || option > 4){
            cout << "Invalied option.\n";
            cout << "\nEnter a option = ";
            cin >> option;
        }

        if (option == 1) mu.Push();
        else if (option == 2) mu.Pop();
        else if (option == 3) mu.Display();
        else if (option == 4){
            cout << "\nThe menu is closed.\n";
            break;
        }
        cout << '\n';
    }
}
```

Input & Output :

```
        *** Stack ***
1. Push.
2. Pop.
3. Display.
4. Exit.

Enter a option = 1
Enter a value to push : 10
Value 10 push to the stack.
```

```
        *** Stack ***
1. Push.
2. Pop.
3. Display.
4. Exit.

Enter a option = 1
Enter a value to push : 20
Value 20 push to the stack.

        *** Stack ***
1. Push.
2. Pop.
3. Display.
4. Exit.

Enter a option = 2
The last value of the stack is poped.

        *** Stack ***
1. Push.
2. Pop.
3. Display.
4. Exit.

Enter a option = 3
Stack = 10

        *** Stack ***
1. Push.
2. Pop.
3. Display.
4. Exit.

Enter a option = 4

The menu is closed.
```

Problem Statement : Write a menu program to create and manipulate Queue using queue class. Perform the following operations using the specified method.

Source Code :

```
#include<bits/stdc++.h>
using namespace std;

class Menu{
private :
    queue<int> q;
public :
    void Enqueue(){
        int x;
        cout << "Enter a value to push : ";
        cin >> x;
        q.push(x);
        cout << "Value " << x << " is enqueued.\n";
    }
    void Dequeue(){
        if (q.empty()) cout << "The queue is already empty.\n";
        else{
            q.pop();
            cout << "The last element is dequeued.\n";
        }
    }
    void Display_Front(){
        cout << "Front element : " << q.front() << ".\n";
    }
    void Display_Back(){
        cout << "Last element : " << q.back() << ".\n";
    }
};

int main(void){
    Menu mu;

    int option;
    while (4){
        cout << "\t*** Queue ***\n";
        cout << "1. Enqueue.\n";
        cout << "2. Dequeue.\n";
        cout << "3. Display Front.\n";
        cout << "4. Display Rear.\n";
        cout << "5. Exit.\n";

        cout << "\nEnter a option = ";
        cin >> option;
        while (option < 1 || option > 5){
            cout << "Invalied option.\n";
            cout << "\nEnter a option = ";
            cin >> option;
        }

        if (option == 1) mu.Enqueue();
        else if (option == 2) mu.Dequeue();
        else if (option == 3) mu.Display_Front();
        else if (option == 4) mu.Display_Back();
        else if (option == 5){
            cout << "\nThe menu is closed.\n";
            break;
        }
        cout << '\n';
    }
}
```

Input & Output :

```
        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 1
Enter a value to push : 10
Value 10 is enqueued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
```

```
5. Exit.

Enter a option = 1
Enter a value to push : 20
Value 20 is enqueued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 1
Enter a value to push : 30
Value 30 is enqueued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 2
The last element is dequeued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 3
Front element : 20.

        *** Queue ***
```

```
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 4
Last element : 30.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 5

The menu is closed.
```

Problem Statement : Write a menu program to create and manipulate linked list using vector class and use the following methods.

Source Code :

```
#include<bits/stdc++.h>
using namespace std;

class Menu{
private :
    vector<int> v;
public :
    void Insert(){
        int x;
        cout << "Enter a value to push : ";
        cin >> x;
        v.push_back(x);
    }
    void Delete(){
        int ind;
        cout << "Enter the index of the value : ";
```

```cpp
        cin >> ind;
        if (ind < 0 || ind >= v.size()) cout << "Invaild 
Index.\n";
        else{
            v.erase(v.begin() + ind);
            cout << "The value of " << ind << "-th index is 
deleted.\n";
        }
    }
    void Update(){
        int ind, value;
        cout << "Enter the index and the value : ";
        cin >> ind >> value;
        if (ind < 0 || ind >= v.size()) cout << "Invaild 
Index.\n";
        else v[ind] = value;
    }
    void Search(){
        int value;
        cout << "Enter the value : ";
        cin >> value;
        auto it = find(v.begin(), v.end(), value);
        if (it != v.end()) cout << "The index of the " << value 
<< " is " << it - v.begin() + 1 << ".\n";
        else cout << "The value is not found.\n";
    }
};

int main(void){
    Menu mu;

    int option;
    while (4){
        cout << "\t*** Vector ***\n";
        cout << "1. Insert.\n";
        cout << "2. Delete.\n";
        cout << "3. Update.\n";
        cout << "4. Search.\n";
        cout << "5. Exit.\n";

        cout << "\nEnter a option = ";
        cin >> option;
        while (option < 1 || option > 5){
            cout << "Invalied option.\n";
            cout << "\nEnter a option = ";
            cin >> option;
        }

        if (option == 1) mu.Insert();
        else if (option == 2) mu.Delete();
        else if (option == 3) mu.Update();
        else if (option == 4) mu.Search();
        else if (option == 5){
            cout << "\nThe menu is closed.\n";
            break;
        }
        cout << '\n';
    }
}
```

Intput & Output :

```
        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 1
Enter a value to push : 10
Value 10 is enqueued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 1
Enter a value to push : 20
Value 20 is enqueued.
```

```
        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 1
Enter a value to push : 30
Value 30 is enqueued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 2
The last element is dequeued.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 3
Front element : 20.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.
```

```
Enter a option = 4
Last element : 30.

        *** Queue ***
1. Enqueue.
2. Dequeue.
3. Display Front.
4. Display Rear.
5. Exit.

Enter a option = 5

The menu is closed.
```