Practical No. 12 (Group E)

Name : Atharva B. Iparkar

Roll no : S211045

Class : S.E.

Div : A

Batch : A-2

Problem Statement :

A double-ended queue (deque) is a linear list in which additions and deletions may be

made at either end. Obtain a data representation mapping a deque into a

onedimensional array. Write C++ program to simulate deque with functions to add

and delete elements from either end of the deque.

Code :

```
#include <iostream>

using namespace std;


#define SIZE 5 // Consistent size


class Dequeue {

    int a[SIZE], front, rear, count;
```

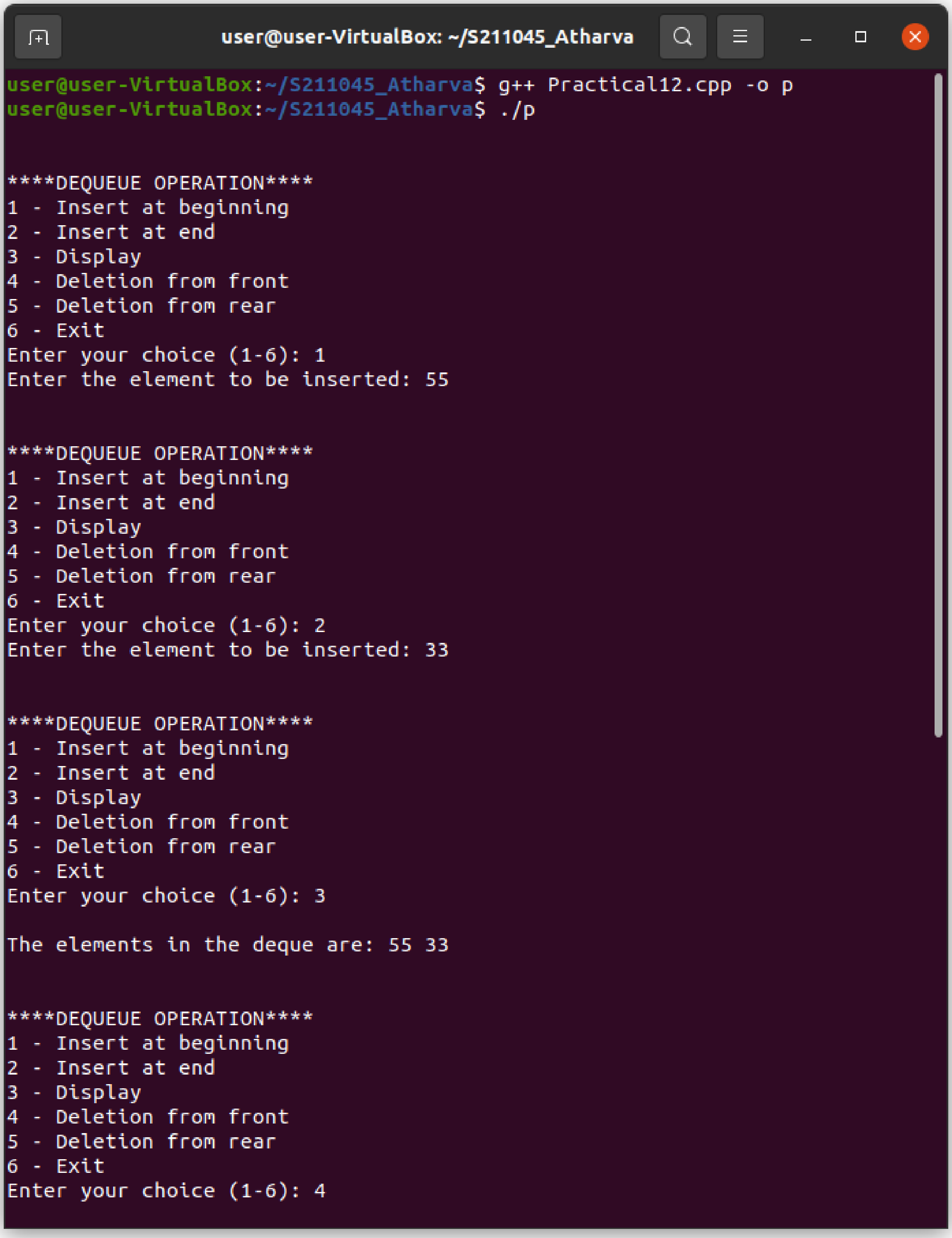
```
public:
    Dequeue();
    void add_at_beg(int);
    void add_at_end(int);
    void delete_fr_front();
    void delete_fr_rear();
    void display();
};

Dequeue::Dequeue() {
    front = -1;
    rear = -1;
    count = 0;
}

void Dequeue::add_at_beg(int item) {
    if (count == SIZE) {
        cout << "\nInsertion is not possible, overflow!!!";
        return;
    }
    if (front == -1) {
        front = rear = 0;
        a[front] = item;
    } else {
        if (front == 0) {
            cout << "\nInsertion is not possible, overflow!!!";
            return;
        }
        for (int i = rear; i >= front; i--) {
            a[i + 1] = a[i];
        }
        a[front - 1] = item; // Place the new item at front - 1
        front--;
    }
    count++;
}

void Dequeue::add_at_end(int item) {
    if (count == SIZE) {
        cout << "\nInsertion is not possible, overflow!!!";
        return;
    }
    if (front == -1) {
        front = rear = 0;
        a[rear] = item;
    } else {
        rear++;
        a[rear] = item; // Insert item at the end
```

```
  }
  count++;
}

void Dequeue::delete_fr_front() {
  if (front == -1) {
    cout << "\nDeletion is not possible: Dequeue is empty";
    return;
  }
  cout << "\nThe deleted element is " << a[front];
  if (front == rear) {
    front = rear = -1; // Queue becomes empty
  } else {
    front++;
  }
  count--;
}

void Dequeue::delete_fr_rear() {
  if (front == -1) {
    cout << "\nDeletion is not possible: Dequeue is empty";
    return;
  }
  cout << "\nThe deleted element is " << a[rear];
```

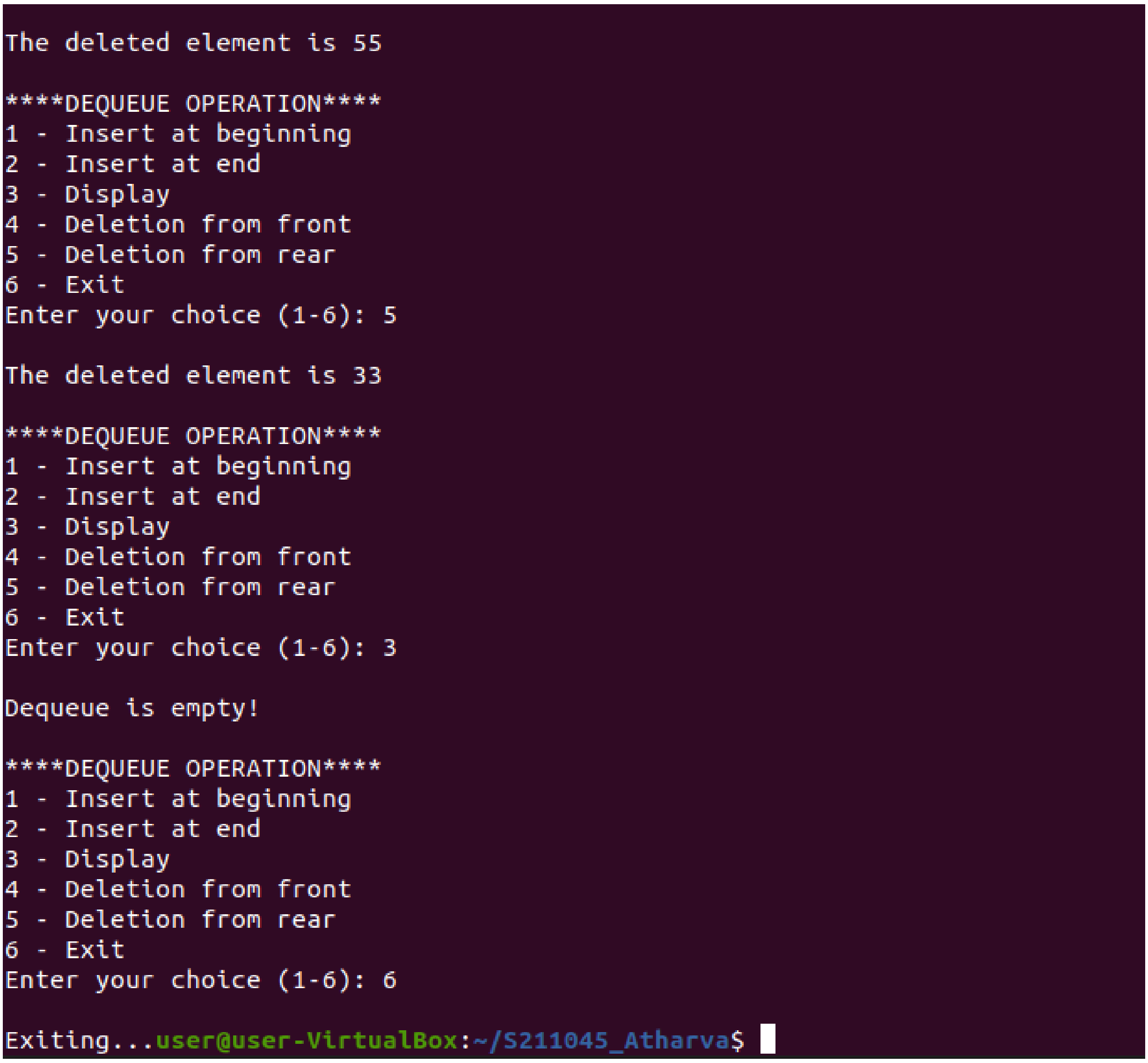
```
    if (front == rear) {
        front = rear = -1; // Queue becomes empty
    } else {
        rear--;
    }
    count--;
}

void Dequeue::display() {
    if (front == -1) {
        cout << "\nDequeue is empty!";
        return;
    }
    cout << "\nThe elements in the deque are: ";
    for (int i = front; i <= rear; i++) {
        cout << a[i] << " ";
    }
    cout << endl;
}

int main() {
    int c, item;
    Dequeue d1;
```

```
do {
    cout << "\n\n****DEQUEUE OPERATION****\n";
    cout << "1 - Insert at beginning\n";
    cout << "2 - Insert at end\n";
    cout << "3 - Display\n";
    cout << "4 - Deletion from front\n";
    cout << "5 - Deletion from rear\n";
    cout << "6 - Exit\n";
    cout << "Enter your choice (1-6): ";
    cin >> c;

    switch (c) {
        case 1:
            cout << "Enter the element to be inserted: ";
            cin >> item;
            d1.add_at_beg(item);
            break;

        case 2:
            cout << "Enter the element to be inserted: ";
            cin >> item;
            d1.add_at_end(item);
            break;
            case 3:
                d1.display();
                break;

            case 4:
                d1.delete_fr_front();
                break;

            case 5:
                d1.delete_fr_rear();
                break;

            case 6:
                cout << "\nExiting...";
                break;

            default:
                cout << "Invalid choice!";
                break;
        }
    } while (c != 6);

    return 0;
}
```

Output :



```
user@user-VirtualBox:~/S211045_Atharva$ g++ Practical12.cpp -o p
user@user-VirtualBox:~/S211045_Atharva$ ./p


****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 1
Enter the element to be inserted: 55


****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 2
Enter the element to be inserted: 33


****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 3

The elements in the deque are: 55 33


****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 4
```



```
The deleted element is 55

****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 5

The deleted element is 33

****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 3

Dequeue is empty!

****DEQUEUE OPERATION****
1 - Insert at beginning
2 - Insert at end
3 - Display
4 - Deletion from front
5 - Deletion from rear
6 - Exit
Enter your choice (1-6): 6

Exiting...user@user-VirtualBox:~/S211045_Atharva$ 
```