DEQUEUE

AIM:

To demonstate the functions of Dequeue in C++.

ALGORITHM:

- Declare an integer deque and required variables for choice and value.

- Display the menu of deque operations to the user.

- Repeat the menu until the user selects the exit option.

- Read the user’s choice.

- If the choice is **Insert at Back**, read a value and insert it using push_back().

- If the choice is **Insert at Front**, read a value and insert it using push_front().

- If the choice is **Delete from Back**, remove the last element using pop_back() if deque is not empty.

- If the choice is **Delete from Front**, remove the first element using pop_front() if deque is not empty.

- If the choice is **Erase**, remove the first element using erase(begin()) if deque is not empty.

- If the choice is **Display**, traverse and display elements using an iterator.

- If the choice is **Exit**, terminate the loop.

PROGRAM:

```
/*
*  Program to demonstrate demostrate iterator in deque
*  Author   : MUTHUGANESH S
*  Date     : 05/2/2026
*  Filename: Iterator.cpp
*  retval   : void
*/

#include <iostream>
#include <deque>
using namespace std;

int main() {
    deque<int>d;
    int Choice, value;

    cout<<"1. Insert at back\n2. Insert at front\n3.Delete from back\n";
    cout<<"4.Delete from front\n5.Erase\n6.Display\n7.Exit\n";
    while(Choice!=7){

        cout<<"\nEnter your choice: ";
```

```
cin>>Choice;

switch(Choice){

    // Inserting elements at back
    case 1:
        cout<<"Enter value to insert at back: ";
        cin>>value;
        d.push_back(value);
        break;

    // Inserting elements at front
    case 2:
        cout<<"Enter value to insert at front: ";
        cin>>value;
        d.push_front(value);
        break;

    // Deleting elements from back
    case 3:
        if(!d.empty()){
            d.pop_back();
            cout<<"Deleted from back\n";
        } else {
            cout<<"Deque is empty\n";
        }
        break;

    // Deleting elements from front
    case 4:
        if(!d.empty()){
            d.pop_front();
            cout<<"Deleted from front\n";
        } else {
            cout<<"Deque is empty\n";
        }
        break;

    // Erasing elements
    case 5:
        if(!d.empty()){
            d.erase(d.begin());
            cout<<"Erased first element\n";
        } else {
            cout<<"Deque is empty\n";
        }
        break;

    // Displaying elements
    case 6:
        cout<<"Deque elements: ";
        for(auto it = d.begin(); it != d.end(); ++it){
            cout<<*it<<" ";
        }
```

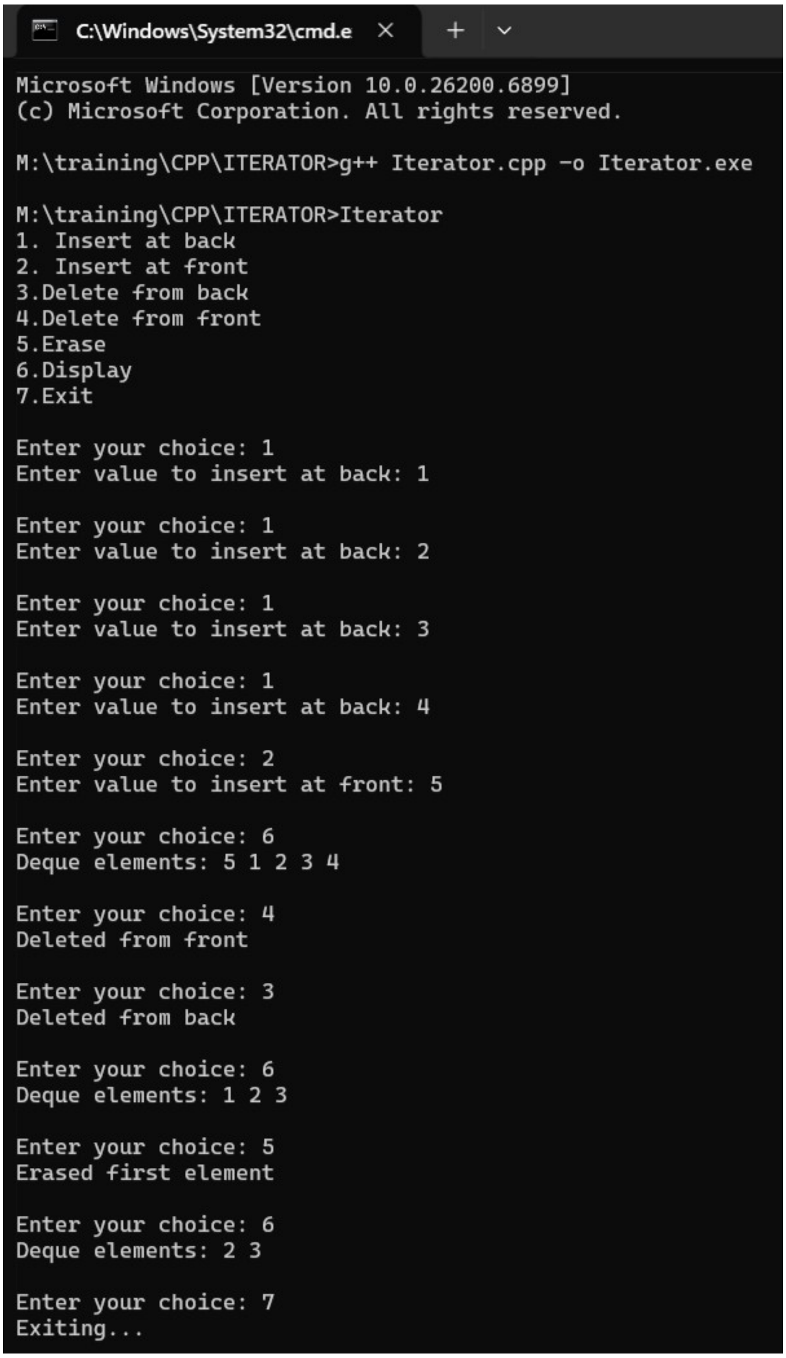
```cpp
                    cout<<endl;
                    break;

                // Exiting
                case 7:
                    cout<<"Exiting...\n";
                    break;

                default:
                    cout<<"Invalid choice. Please try again.\n";
            }
        }

        return 0;
    }
```

OUTPUT:



```
Microsoft Windows [Version 10.0.26200.6899]
(c) Microsoft Corporation. All rights reserved.

M:\training\CPP\ITERATOR>g++ Iterator.cpp -o Iterator.exe

M:\training\CPP\ITERATOR>Iterator
1. Insert at back
2. Insert at front
3.Delete from back
4.Delete from front
5.Erase
6.Display
7.Exit

Enter your choice: 1
Enter value to insert at back: 1

Enter your choice: 1
Enter value to insert at back: 2

Enter your choice: 1
Enter value to insert at back: 3

Enter your choice: 1
Enter value to insert at back: 4

Enter your choice: 2
Enter value to insert at front: 5

Enter your choice: 6
Deque elements: 5 1 2 3 4

Enter your choice: 4
Deleted from front

Enter your choice: 3
Deleted from back

Enter your choice: 6
Deque elements: 1 2 3

Enter your choice: 5
Erased first element

Enter your choice: 6
Deque elements: 2 3

Enter your choice: 7
Exiting...
```