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

#define MAX 10

// class representing a node
struct Node {
    int data;
    struct Node *next;
    struct Node *prev;

    Node () {
        next = NULL;
        prev = NULL;
    }
};


// Deuqe class
class Deque {
    struct Node *head, *tail, *temp;
    int size, front, rear;

    public:
        Deque ();
        bool isFull();
        bool isEmpty();
        void pushfront(int);
        void pushrear(int);
        int popfront();
        int poprear();
        void display();

};


Deque::Deque () {
    head = tail = temp = NULL;
    size = 0;
    front = rear = -1;
}


bool Deque::isFull() {
    return ((front + rear == MAX - 1) || (front == MAX - 1) || (rear == MAX - 1));
}


bool Deque::isEmpty() {
    return ((front + rear == -1) || (head == NULL && tail == NULL));
}
```

```cpp
void Deque::pushfront(int x) {
    if (isFull()) {
        cout << "\nDeque is Full. OVERFLOW!!!!";
        return;
    }

    // First node
    if (front == -1) {
        head = new (struct Node);
        head->data = x;
        tail = head;
        front++;
        rear++;
    }

    // Other node
    else {
        temp = new (struct Node);
        temp->data = x;
        temp->prev = tail;
        tail->next = temp;
        tail = temp;
        front++;
    }

    size++;
}


void Deque::pushrear(int x) {
    if (isFull()) {
        cout << "\nDeque is Full. OVERFLOW!!!!";
        return;
    }

    // First node
    if (front == -1) {
        head = new (struct Node);
        head->data = x;
        tail = head;
        front++;
        rear++;
    }

    // Other node
    else {
        rear++;
        temp = new (struct Node);
        temp->data = x;
        temp->next = head;
        head->prev = temp;
        head = temp;
    }

    size++;
}


int Deque::popfront() {
    int data;

    if (isEmpty()) {
        cout << "\nDeque is empty. UNDERFLOW!!!!";
        return -1;
    }
    data = tail->data;

    // Last node
    if (size == 1) {
        head = tail = NULL;
    }

    // Other node
    else {
        tail = tail->prev;
        tail->next = NULL;
    }

    front--;

    size--;
    return data;
}


int Deque::poprear() {
    int data;

    if (isEmpty()) {
        cout << "\nDeque is empty. UNDERFLOW!!!!";
        return -1;
    }
    data = head->data;

    // Last node
    if (size == 1) {
        head = tail = NULL;
    }

    // Other node
    else {
        head = head->next;
        head->prev = NULL;
    }

    rear--;

    size--;
    return data;
}


void Deque::display() {
    temp = head;
    cout << '\n';
    while (temp) {
        cout << temp->data << " --> ";
        temp = temp->next;
    }
    cout << "\b\b\b\b    ";
}


int main() {
    Deque deque;
    int ch;

    while (1) {
        cout<< "\n\n1. Insert at front\n"
            << "2. Insert at rear\n"
            << "3. Pop from front\n"
            << "4. Pop from rear\n"
            << "5. Display Deque\n"
            << "6. Exit\n\n"
            << "Choose your option <1-6> : ";
        cin >>ch;

        switch(ch) {
            case 1:
                cout <<"Enter the Element: ";
                cin >> ch;
                deque.pushfront(ch);
                break;
            case 2:
                cout <<"Enter the Element: ";
                cin >> ch;
                deque.pushrear(ch);
                break;
            case 3:
                cout << "\nPopped element is : " << deque.popfront();
                break;
            case 4:
                cout << "\nPopped element is : " << deque.poprear();
                break;
            case 5:
                cout << "\nDeque is :\n";
                deque.display();
                break;
            case 6:
                exit(1);
            }
    }
}
```

```
/*

-------------------- OUTPUT ----------------------

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 1


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 2


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 3


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 4


1. Insert at front
2. Insert at rear
3. Pop from front
```

```
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 5


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 6


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 1
Enter the Element: 7


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 2
Enter the Element: 8


1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :
```

```
8 ⟶ 6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3 ⟶ 5 ⟶ 7

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 3

Popped element is : 7

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

8 ⟶ 6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3 ⟶ 5

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 4

Popped element is : 8

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3 ⟶ 5
```

```
1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 3

Popped element is : 5

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

6 ⟶ 4 ⟶ 2 ⟶ 1 ⟶ 3

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 4

Popped element is : 6

1. Insert at front
2. Insert at rear
3. Pop from front
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 5

Deque is :

4 ⟶ 2 ⟶ 1 ⟶ 3

1. Insert at front
2. Insert at rear
3. Pop from front
```

```
4. Pop from rear
5. Display Deque
6. Exit

Choose your option <1-6> : 6

*/
```