**LETSUPGRADE- DATA STRUCTURES AND ALGORITHMS- ASSIGNMENT DAY 7**

**(Srishti Rakesh Pandey)**

**Question**

Write a program implementing insert, delete and display operation of Circular Queue.

**Program Code**

#include<bits/stdc++.h>

using namespace std;

struct Queue

{

// Initialize front and rear

int rear, front;

// Circular Queue

int size;

int \*arr;

Queue(int s)

{

front = rear = -1;

size = s;

arr = new int[s];

}

void enQueue(int value);

int deQueue();

void displayQueue();

};   
/\* Function to create Circular queue \*/

void Queue::enQueue(int value)

{

    if ((front == 0 && rear == size-1) ||

            (rear == (front-1)%(size-1)))

    {

        printf("\nQueue is Full");

        return;

    }

    else if (front == -1) /\* Insert First Element \*/

    {

        front = rear = 0;

        arr[rear] = value;

    }

    else if (rear == size-1 && front != 0)

    {

        rear = 0;

        arr[rear] = value;

    }

    else

    {

        rear++;

        arr[rear] = value;

    }

}

// Function to delete element from Circular Queue

int Queue::deQueue()

{

    if (front == -1)

    {

        printf("\nQueue is Empty");

        return INT\_MIN;

    }

    int data = arr[front];

    arr[front] = -1;

    if (front == rear)

    {

        front = -1;

        rear = -1;

    }

    else if (front == size-1)

        front = 0;

    else

        front++;

    return data;

}

// Function displaying the elements

// of Circular Queue

void Queue::displayQueue()

{

    if (front == -1)

    {

        printf("\nQueue is Empty");

        return;

    }

    printf("\nElements in Circular Queue are: ");

    if (rear >= front)

    {

        for (int i = front; i <= rear; i++)

            printf("%d ",arr[i]);

    }

    else

    {

        for (int i = front; i < size; i++)

            printf("%d ", arr[i]);

        for (int i = 0; i <= rear; i++)

            printf("%d ", arr[i]);

    }

}

/\* Driver of the program \*/

int main()

{

    Queue q(5);

    // Inserting elements in Circular Queue

    q.enQueue(14);

    q.enQueue(22);

    q.enQueue(13);

    q.enQueue(-6);

    // Display elements present in Circular Queue

    q.displayQueue();

    // Deleting elements from Circular Queue

    printf("\nDeleted value = %d", q.deQueue());

    printf("\nDeleted value = %d", q.deQueue());

    q.displayQueue();

    q.enQueue(9);

    q.enQueue(20);

    q.enQueue(5);

    q.displayQueue();

    q.enQueue(20);

    return 0;

}

