Circular Queue Using Linked List-

Enqueue And Dequeue -

```
// Online C++ compiler to run C++ program online
#include <iostream>
using namespace std;

class Node{
    public:
    int data;
    Node* next;

    Node(int data){
        this->data=data;
        this->next=NULL;
    }
};
class cercularQueue{
    public:
    Node* front;
    Node* rear;

    public:
    cercularQueue(){
        front=nullptr;
        rear=nullptr;
    }

    bool isEmpty(){
        return(front==nullptr);
    }

    void Enqueue(int data){
        Node* newqueue=new Node(data);
        if(isEmpty()){
```

```
        front=rear=newqueue;
        rear->next=front;
    }else{
        rear->next=newqueue;
        rear=newqueue;
        rear->next=front;
    }
    cout<<"INSERTED"<<data<<endl;
}

void Display(){
    if(isEmpty()){
        cout<<"Queue is Empty";
    }
    Node* temp=front;
    do{
        cout<<temp->data<<" ";
        temp=temp->next;
    }while(temp!=front);
    cout<<endl;
}

void Dequeue() {
    if (isEmpty()) {
        cout << "Queue is Empty! Cannot Dequeue." << endl;
        return;
    }

    Node* temp = front;
    int removedData = temp->data;

    if (front == rear) {
        front = rear = nullptr;
    } else {
        front = front->next;
        rear->next = front;
```

```
        }

        delete temp;
        cout << "Removed: " << removedData << endl;
    }
};


int main() {
    cercularQueue cq;
    cq.Enqueue(20);
    cq.Enqueue(90);
    cq.Display();
    cq.Dequeue();
    cq.Display();
     return 0;
}
```