Implementation of queue using linked list

make a node class and declare data and next pointer. Maintain head and tail pointer at every point.

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

For implementing enqueue - insert at tail

Maintain two pointers for queue - one head and one tail pointer.

tail for enqueue and head for dequeue.

check corner cases - whether underflow or overflow occurs.

```
#include <bits/stdc++.h>
class Node
{
    public:
    int data;
    Node* next;
    Node(int data)
    {
        this->data=data;
        next=NULL;
    }
};
class Queue {
public:
    Node* head;
    Node* tail;
    Queue() {
        // Implement the Constructor
        head=NULL;
        tail=NULL;
    }

    /*----------------- Public Functions of Queue -----------------*/

    bool isEmpty() {
        // Implement the isEmpty() function
        if(head==NULL)
        {
            return 1;
        }
        return 0;
    }

    void enqueue(int data) {
        // Implement the enqueue() function
```

```
        Node* newNode=new Node(data);
        if(head==NULL)
        {
            head=newNode;
            tail=newNode;
            return;
        }
        tail->next=newNode;
        tail=newNode;
    }

    int dequeue() {
        // Implement the dequeue() function
        if(head==NULL)
            return -1;
        int ans=head->data;
        Node* temp=head;
        head=head->next;
        if(head==NULL)
        {
            tail=NULL;
        }
        delete temp;
        return ans;
    }

    int front() {
        // Implement the front() function
        if(head==NULL)
            return -1;
        else
            return head->data;
    }
};
```