Assignment-3

Implement a generic Queue data structure in Java using a linked list. The Queue should support the basic operations of enqueue, dequeue, peek, and check if the queue is empty.

```
package datastructures.linear;

class Queue{
        static private int front,rear,capacity;
        static private int queue[];

        Queue(int c){
                front=rear=0;
                capacity=c;
                queue=new int[capacity];
        }
        void queueEnque(int data) {
                if(rear==capacity) {
                        System.out.println("Queue is empty");
                }
                else {
                        queue[rear]=data;
                        rear++;
                        System.out.println("Inserted..\n");
                }
                return;
        }
         void queueDeque() {
                if(front==rear) {
                        System.out.println("\n Queue is empty");
                return;
            }
            else {
                for(int i=0;i<rear-1;i++) {
                    queue[i]=queue[i+1];
                }
                if(rear<capacity) {
                    queue[rear]=0;
                }
                rear--;
            }
        }
         void queueDisplay(){
            if(front==rear) {
                System.out.println("\nQueue is Empty");
                return;
            }
            for(int i=front;i<rear;i++) {
                System.out.printf("%d <-- ",queue[i]);
            }
            return;
        }

         void queueFront() {
             if(front==rear) {
                  System.out.println("Queue is empty");

             }
             else
             {
```

```
                        System.out.printf("\nfront element is
:%d",queue[front]);

                }
                }
}




public class QueueOperations {
        public static void main(String[] args) {
                Queue q=new Queue(5);
                q.queueEnque(10);
                q.queueEnque(20);
                q.queueEnque(30);
                q.queueEnque(40);
                q.queueEnque(50);

                System.out.println("Queue elements :");
                q.queueDisplay();

                q.queueDeque();
                System.out.println("Queue elements after delete");
                q.queueDisplay();
                q.queueFront();

        }
}
```