File - C:\Users\niall\IdeaProjects\B00787301_COM498_Assignment2\src\Queue.java

```java
import java.util.EmptyStackException;

public class Queue <T> implements QueueInterface<T>
{
    private linkNode<T> front; // defines front node variable
    private linkNode<T> rear; // defines rear node variable
    private int numberOfEntries;

    public Queue()
    {
        front = null; // instantiates front node to null
        rear = null; //instantiates rear node to null
    }

    public void enqueue(T newEntry)
    {
        linkNode<T> newNode = new linkNode<T>(newEntry); // e.g. Jim as a parameter

        if (front == null) // if front variable is null
        {
            front = newNode; //set front variable to first in list
            rear = newNode; //set rear variable to front in list
        }
        else
        {
            rear.setNext(newNode); // use current rear node variable (currently null) and set its next to newNode variable e.g. set null's next node to Jim
            rear = newNode; // then set the newNode variable to rear e.g. null is now set to jim
        }
        /*
        if (front == null)
        front = newNode;
            newNode.setNext(rear); //for newly created node, sets its next as what is stored in the topNode variable, e.g. new node (2), passes current topNode (currently 1) as next node to create link
            rear = newNode;
        */
    }

    public T dequeue() {
        if (front == null)
            {
            return null;
            }
        else
            {
            T dataToReturn = front.getData(); //returns current front node
            front = front.getNext();

            if (front == null)
                rear = null;
            return dataToReturn;
            }
        /*
         T dataToReturn = front.getData();
        front = front.getNext(); //sets the next in stack as current topNode (2)
        return dataToReturn;
         */
    }

    public T getFront()
    {
        if (front == null)
        {
            throw new EmptyStackException();
        }
        else
            return front.getData(); //gets data of current top node (3)
    }

    public boolean isEmpty()
    {
        return (rear == null);
    }

    public void clear()
    {
```

```
75          front = null;
76          rear = null;
77      }
78
79  }
80
```