Error Log for Task 1: QueueList

Error 1: Incorrect Handling of Empty Queue in `dequeue()`

- **Description**:
  The `dequeue()` method did not check if the queue was empty before attempting to remove an element. This led to a potential `NullPointerException` when trying to access or remove an element from an empty queue.

  Fix:
  I added a condition in the `dequeue()` method to check if the queue is empty using the `isEmpty()` method. If the queue is empty, the method throws an `IllegalStateException` to prevent further operations.

  Code Fix Example:

```
public T dequeue() {

    if (isEmpty()) {

        throw new IllegalStateException("Cannot dequeue from an empty queue.");

    }

    // Existing dequeue logic

}
```

-

Error 2: Incorrect Update of the `rear` Pointer

- **Description**:
  When the queue became empty after a `dequeue()` operation, the `rear` pointer was not set to `null`. This caused the `rear` pointer to reference an outdated node that no longer existed in the queue, leading to inconsistencies.

  Fix:
  After updating the `front` pointer during the `dequeue()` operation, I added a

check to set the `rear` pointer to `null` if the queue becomes empty. This ensures a consistent state for the queue.

Code Fix Example:

```
public T dequeue() {

   if (isEmpty()) {

       throw new IllegalStateException("Cannot dequeue from an empty queue.");

   }

   T data = front.data;

   front = front.next;

   if (front == null) { // Queue is now empty

       rear = null;

   }

   return data;

}
```

-

Error 3: No Check for Queue Size During Operations

- **Description**:
  The `size` field was not updated during `enqueue()` and `dequeue()` operations, leading to incorrect values being returned by the `size()` method. This created inconsistencies when querying the number of elements in the queue.

Fix:
I updated the `enqueue()` and `dequeue()` methods to increment or decrement the `size` field after each operation. This ensures that the `size()` method always returns the correct number of elements in the queue.

Code Fix Example:

```
 public void enqueue(T data) {
    Node newNode = new Node(data);
    if (isEmpty()) {
        front = rear = newNode;
    } else {
        rear.next = newNode;
        rear = newNode;
    }
    size++; // Increment size after enqueue
}

public T dequeue() {
    if (isEmpty()) {
        throw new IllegalStateException("Cannot dequeue from an empty queue.");
    }
    T data = front.data;
    front = front.next;
    if (front == null) {
        rear = null;
    }
    size--; // Decrement size after dequeue
    return data;
}
```

```
public int size() {
    return size;
}
```

-