Data Structure and Algorithm Practicum
Quiz 2

Name
Dicha Zelianivan Arkana

NIM
2241720002

Class
1i

Department
Information Technology

Study Program
D4 Informatics Engineering

1 Questions

1. Complete the `addLast()` and `deleteLast()` method inside the class `SingleLinkedList`

   - `addLast()`

     ```java
     void addLast(int data) {
         // create a new node
         Node newNode = new Node(data);
         // checks if it's empty, meaning that we don't have any node yet
         // so just assign the head and tail to the new node
         if (isEmpty()) {
             head = tail = newNode;
         } else {
             // otherwise, let's insert the new node into the next pointer of the tail
             tail.next = newNode;
             // and then replace the tail with our node
             tail = newNode;
         }
         // increment the size
         size++;
     }
     ```

   - `deleteLast()`

     ```java
     void deleteLast() {
         Node tmp = head;
         // traverse through the entire list until second to last
         while (tmp.next.next != null) {
             tmp = tmp.next;
         }
         // replace the tail with the second to last node
         tail = tmp;
         // remove the last node using the next pointer from the second to last node
         tail.next = null;
         // decrement the size
         size--;
     }
     ```

2. Complete the method `merge()` and `split()` inside the method `Main`

- `merge()`

```java
public static void merge(SingleLinkedList l1, SingleLinkedList l2) {
    SingleLinkedList mergedList = new SingleLinkedList();

    // insert every nodes from the first list into the merged list
    Node current = l1.head;
    while (current != null) {
        // use addLast to insert it consecutively in order
        // instead of in reverse order
        mergedList.addLast(current.data);
        // traverse to the next node
        current = current.next;
    }

    // do the same thing with the second node
    current = l2.head;
    while (current != null) {
        mergedList.addLast(current.data);
        current = current.next;
    }

    // just to proof the list has been merged
    System.out.print("Merged List: ");
    mergedList.print();
}
```

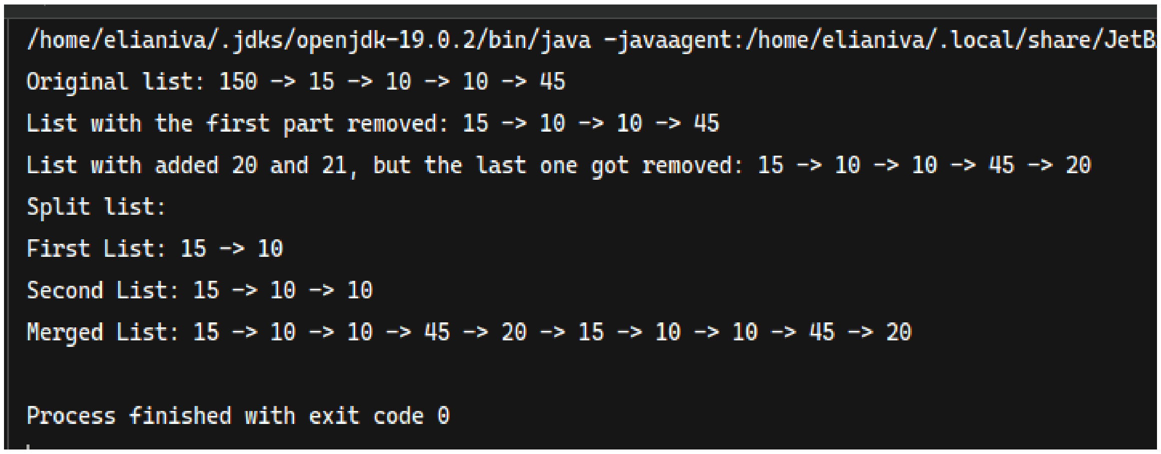
- `split()`

```java
public static void split(SingleLinkedList list) {
    // prepare 2 lists that will be used to contain our split list
    SingleLinkedList firstList = new SingleLinkedList();
    SingleLinkedList secondList = new SingleLinkedList();

    // get the middle point since we're going to split it by half
    int middle = list.size / 2;

    // split the first portion into the first list
    Node current = list.head;
    // loop until we reach the half point
    for (int i = 0; i < middle; i++) {
        // again, we want to insert it in order so we'll use
        // addLast instead of addFirst
        firstList.addLast(current.data);
        current = current.next;
    }

    // do the same thing but for the last portion of the list
    current = list.head;
    // we start from the middle until the last part of the list
    // which is equal to the list.size
    for (int i = middle; i < list.size; i++) {
        secondList.addLast(current.data);
        current = current.next;
    }

    System.out.println("Split list: ");

    // just to proof that the list has been split
    System.out.print("First List: ");
    firstList.print();

    // do the same for the second list
    System.out.print("Second List: ");
    secondList.print();
}
```

2 Solution Source Code

See the attached files

3 Screenshots



```
/home/elianiva/.jdks/openjdk-19.0.2/bin/java -javaagent:/home/elianiva/.local/share/JetB
Original list: 150 -> 15 -> 10 -> 10 -> 45
List with the first part removed: 15 -> 10 -> 10 -> 45
List with added 20 and 21, but the last one got removed: 15 -> 10 -> 10 -> 45 -> 20
Split list:
First List: 15 -> 10
Second List: 15 -> 10 -> 10
Merged List: 15 -> 10 -> 10 -> 45 -> 20 -> 15 -> 10 -> 10 -> 45 -> 20

Process finished with exit code 0
```

Figure 1: The output of the program when ran from `Main.java`