Homework 04 (Linked List 02)

You are given all classes for coding Linked List.

An incomplete ZoomaList class is also given. This class represents a linked list used to store data in the game "Zooma". It contains variable score to keep track of a player's score. The rule may not be exactly the same as the original Zooma, so you must read the question carefully.

- The Zooma list can start with any number in it, these numbers may contain duplicated and consecutively repeated numbers, such as 1,3,3,6,6,6,6,3,4,4,4,1,8. (This means repeated numbers will not be removed at this stage).
- When we "insert" a number into the list, if the insertion causes 3 or more repeated numbers, those numbers are removed, and the score is incremented according to the removed numbers.
    - With 1,3,3,6,6,6,6,3,4,4,4,1,8 if we insert 6 at position just after the second 3:
        - 1,3,3,(6),6,6,6,6,3,4,4,4,1,8 // the insertion value is in bracket
    - This insertion causes five repeated values of 6s, so they are removed, and the score is incremented by 5. The list that remains is:
        - 1,3,3,3,4,4,4,1,8 //yellow highlight indicates collision points between 2
        - //separate list portions.
    - If the collision points produce three or more repeated numbers, they must be removed and the score must be incremented. For this example, we get:
        - 1,4,4,4,1,8 //the score now incremented by 3
    - If the collision points no longer produce three or more consecutive numbers, stop the "insert". Otherwise, keep removing repeated data caused by collision points.
    - For this example, even though there are three 4s, they are not removed because they are not caused by the collision points. The final list will therefore be:
        - 1,4,4,4,1,8

```
public class ZoomaList extends CDLinkedList {
    int score = 0;

    public ZoomaList() {
        super();
    }

    public ZoomaList(CDLinkedList l) {
        header = l.header;
        size = l.size;
    }
```

1. (6 marks) Write method

```
public void removeBetween(DListIterator left, DListIterator right, int inc)
```

- Put this method as the last method in your class, otherwise the test file won't grade you properly.
- This method removes data between (not including) left and right. It reduces the list size by "inc".
- left is always the beginning of the range.

- score does not play any part in this method.
- If there is no data between left and right (see Figure 1), this method does nothing.
- This method only performs 1 remove, without looking at data in the list.
- Your code must not have any loop. The file TestNoLoop.java tests this and if it fails, your maximum score for this question becomes 3.

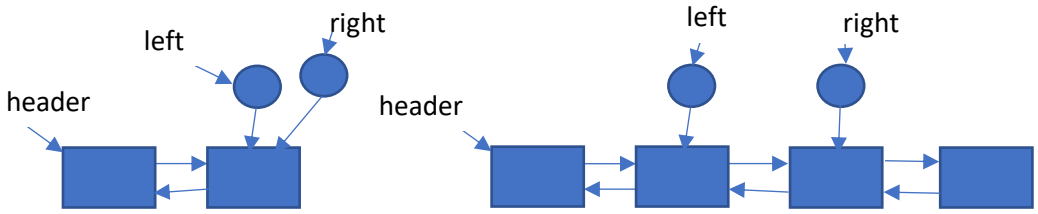

Figure 1 lists that do not have enough remove range.

For general case see Figure 2, Figure 3, and testcases in TestRemoveBetween.java.

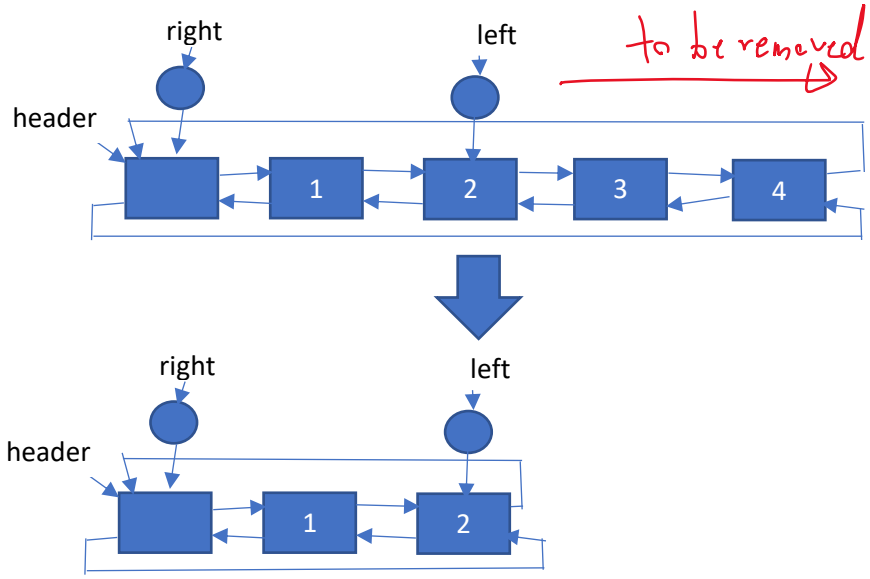

Figure 2 Removing, when left is closer to the end of the list.

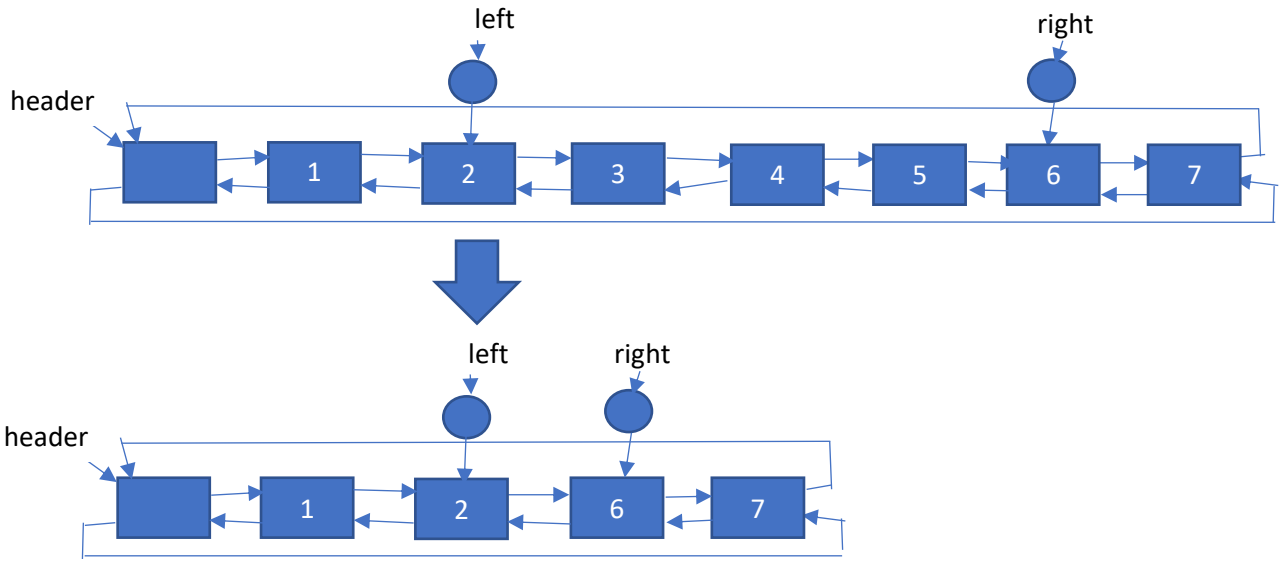

Figure 3 General case of removal

- In TestNoLoop.java, you will have to supply a correct path for your "ZoomaList.java".

2. (8 marks) Write method

```
public void insert(int value, Iterator p) throws Exception {
```

This method performs the "insert" as described in the first page of this assignment.

- Remember that if the list contains zero or one data, this "insert" will surely work like a regular "insert" from CDLinkedList.
- See test cases (they have good comments description) from TestZoomaList.java