@CS
Problem Set 2
Linked Lists

Assuming an IntNode class defined like this:

```
public class IntNode {
   public int data;
   public IntNode next;
   public IntNode() {
       this.data = 0; this.next = null;
   }
   public IntNode(int data, IntNode next) {
       this.data = data; this.next = next;
   }
   public String toString() {
       if(next != null) return data + " " + next.toString();
       return data + "";
   }
```

1. Implement a method that will add a new integer before a target integer in the list. The method should return a pointer/reference to the front node of the resulting list. If the target is not found, it should return front without doing anything:

```
public static IntNode addBefore(IntNode front, int target, int newItem)
{ /* COMPLETE THIS METHOD */ }
```

2. Implement a method that will add a new integer before the last item in a linked list. (In other words, the added integer will become the second-to-last item in the resulting linked list.) The method should return a pointer/reference to the front node of the resulting linked list. If the input linked list is empty, the method should return null, without doing anything.

```
public static IntNode addBeforeLast(IntNode front, int item)
{ /* COMPLETE THIS METHOD */ }
```

3. Implement a method that will search a given linked list for a target int, and return the number of occurrences of the target:

```
public static int numberOfOccurrences(IntNode front, int target)
{ /* COMPLETE THIS METHOD */ }
```

4. Implement a method to delete EVERY OTHER item from an integer linked list. For example:

```
before: 3->9->12->15->21    after: 3->12->21
before: 3->9->12->15        after: 3->12
before: 3->9                after: 3
before: 3                   after: 3
```

If the list is empty, the method should do nothing.

```
public static void deleteEveryOther(IntNode front)
{ /* COMPLETE THIS METHOD */ }
```

5. Implement a method that will delete all occurrences of a given target int from a linked list, and return a pointer to the first node of the resulting linked list:

```
public static IntNode deleteAllOccurrences(IntNode front, int target)
{ /* COMPLETE THIS METHOD */ }
```

6. Implement a method to find the common elements in two sorted linked lists, and return the common elements in sorted order in a NEW linked list. The original linked lists should not be modified. So, for instance,

   `l1 = 3->9->12->15->21` `l2 = 2->3->6->12->19`

   should produce a new linked list: `3->12`

You may assume that the original lists do not have any duplicate items. Return null if there are no common items.

```
public static IntNode commonElements(IntNode frontL1, IntNode frontL2)
{ /* COMPLETE THIS METHOD */ }
```

Submit your .java file in a zipped folder titled with your LastnameFirstname