CMSC 132 Quiz 2 Worksheet

The next quiz for the course will be on Thursday, June 21. The following list provides more information about the quiz:

- The quiz will be a written quiz (no computer).
- Closed book, closed notes quiz.
- Answers must be neat and legible. **You must use pencil.**
- Check the information available at http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours. **We strongly recommend you do not use Eclipse to write the code associated with these exercises.** Try to answer the exercises in a piece of paper and then use Eclipse to verify your solutions. This approach will better prepare you for the quiz. **You cannot use any Java API class (except String) during the implementation of the methods below.**

**Exercises**

Implement the methods below based on the following Java class definitions.

```java
public class LinkedList<T> {
    public class Node {
        private T data;
        private Node next;
        public Node(T data) {
            this.data = data;
            next = null;
        }
    }
    Node head;
}
```

1. Define a constructor for the LinkedList class that creates an empty list.
2. Define a method called `addFirst` that adds an element to the beginning of the list.
3. Define a method called `addLast` that adds an element to the end of the list.
4. Define a method named `size` that returns the number of elements in the list.
5. Implement a method named `toString()` that returns a String with the data component of each node in the list.
6. Implement a method called `getCount` which has the following prototype:
   ```java
   public int getCount(T targetElement)
   ```
   The method returns the number of instances of `targetElement` in the list.
7. Implement a method called `append` which has the following prototype:
   ```java
   public void append(ArrayList<T> data)
   ```
   The method appends the elements from the ArrayList to the end of the LinkedList object. You must handle the case when the list is empty.
8. Implement a method named `removeLastElement` that removes the last element from the list.

9. Implement a method named `elementsInRange` that returns a LinkedList with the elements in the specified range. Notice that the original list must not be modified. Throw an exception for invalid parameters or if the current object list is empty.

   ```java
   public LinkedList<T> elementsInRange(int startIndex, int endIndex)
   ```

10. Define a method called `delete` that has the following prototype:

    ```java
    public boolean delete(T target);
    ```

    The method will delete the first instance of `target` from the list. The method will return true if a `target` instance is found and false otherwise. The list should not be modified if a `target` instance is not found.