CMSC 132 Quiz 3 Worksheet

The next quiz for the course will be on Wed, Oct 24. The following list provides additional information about the quiz:

- **Do not post any solutions to this worksheet in Piazza.**
- The quiz will be a written quiz (no computer).
- The quiz will be in lecture.
- Closed book, closed notes quiz.
- Answers must be neat and legible.
- Quiz instructions can be found at http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html
- Make sure you know your section number and your TA's name.
- **You must take your quiz in your assigned lab/discussion section and not show up to a random lecture section. We will not grade quizzes taken in the incorrect section.**

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 TAs or instructors during office hours. It is recommended that you try these exercises on paper first (without using a computer).

Exercises

1. What is the Java Hash Code Contract?

2. Implement the methods below based on the following Java class definitions. For recursive methods you may only add one auxiliary function, and you may not add any instance nor static variables.

```
public class LinkedList<T extends Comparable<T>> {
   private class Node {
      private T data;
      private Node next;

      private Node(T data) {
         this.data = data;
         next = null;
      }
   }

   private Node head;

   public LinkedList() {
      head = null;
   }
}
```

   a. Define a constructor that takes a **TreeSet<T>** as a parameter and initializes a linked list with the elements in the set. The new list must be sorted in increasing lexicographic order.

   b. Define a **RECURSIVE** method named **size** that returns the number of elements in the list. The prototype for this method is:

```
public int size()
```

   c. Define a **RECURSIVE** method named **inRange** that returns a **HashSet<T>** with the elements in the list that in the specified range. The range includes the lower and upper bound. The prototype for this method is:

```
public HashSet<T> inRange(T lowerBound, T upperBound)
```

   d. Define a **RECURSIVE** method named **remove** that removes all instances in the list that are equal to the target parameter. The prototype for this method is:

```
public void remove(T target)
```

e. Define a **RECURSIVE** method named **positionOfElementInList** that returns a **TreeMap<T, Integer>** that maps each element of the list to its position in the list. The prototype for this method is:

```
public TreeMap<T, Integer> positionOfElementInList()
```

f. Define an equals method for the class. Two lists are equal if the data elements in the two lists are the same (and appear in the same position). Make sure your class satisfies the Java Hash Code Contract.