Homework 3 (due by 4 pm on Nov 14)

Objectives:

- Implement your own Linked List from scratch using recursion.
- Solidify your understanding of recursion.
- Have a better insight into sorting in Java.
- Take a look at an interface and implement it on your own.

In homework 1, you implemented the MyArray class to store words parsed from a text file.

In this homework, you want to accomplish very similar results as you got from the homework 1 such as adding new words, getting the current size of the list, removing words and displaying the current values but you are to implement and use a different Data Structure class.

```
public class SortedLinkedList implements MyListInteface
```

Your SortedLinkedList must have the following properties.

- It implements MyListInterface that is provided to you.
- It does not allow any duplicate at any point. It needs to check duplicates as it adds a new word.
- It has nodes as its elements and each node's data type is String.
- It is sorted in an ascending order. (For example, when you add "d", "c", "a" and "b" sequentially into the SortedLinkedList and then call the display() method of the list, you should see [a, b, c, d] as a result.)

In addition to the properties, **all of the methods in the SortedLinkedList must USE *recursion***. In other words, ***there is no for loop or while loop*** in your SortedLinkedList implementation.

Along with a constructor with no parameters, the SortedLinkedList should have **another constructor that takes unsorted String array as a parameter and build a new SortedLinkedList** as you can see from the example of calling this constructor in the MainDriver.java

You should see the following results when you run the MainDriver with the childrensbible.txt file provided to you. The display method should print all the values even though the example below uses ellipsis to reduce the space.

```
god is found
Number of words in the file is: 1425
[a, abel, abigail, able, abound, about, above, abraham, absalom,
accepted, account,…..]
Number of words in the file is: 1415
[account, acts, adam, advanced, advice, affairs, …..]
```

MainDriver.java

```
import java.io.*;
import java.util.*;
public class MainDriver {
      public static void main(String[] args) {
            MyArray words = new MyArray(10);
            Scanner scanner = null;
            try {
                  scanner = new Scanner(new
File("childrensbible.txt"));
                  while(scanner.hasNextLine()) {
                        String line = scanner.nextLine();
                        String[] wordsFromText = line.split("\\W");
                        for(String word:wordsFromText)
                              words.add(word.toLowerCase());
                  }
            } catch(FileNotFoundException e) {
                  System.err.println("Cannot find the file");
            } finally {
                  if(scanner!= null) scanner.close();
            }
            String[] unsorted = new String[words.size()];
            for(int i=0; i<unsorted.length; i++) {
                  unsorted[i] = words.get(i);
            }
            SortedLinkedList sortedWords = new
SortedLinkedList(unsorted);
            if(sortedWords.contains("god"))
```

```
                System.out.println("god is found");
            else
                System.out.println("not found");
            System.out.println("Number of words in the file is:
"+sortedWords.size());
            sortedWords.display();
            for(int i=0; i<9;i++)
                sortedWords.removeFirst();
            sortedWords.removeAt(0);
            System.out.println("Number of words in the file is:
"+sortedWords.size());
            sortedWords.display();
        }
}
```

Deliverables:

- A few sheets of paper that have your initial code as well as your comments (Submit this in class that is on the due date.)
    - *In your comments, focus on explaining how your recursion works for any method that recursion is used. (For example, what is the base case and what is the recursive case?)*
- Your source code file. (Submit only SortedLinkedList.java file on Blackboard.)

Grading:

Your homework will be graded first by compiling and testing it. After that, we will read your code to determine appropriate methods/classes are used. In addition, we will judge the overall style and modularity of your code. Points will be deducted for poor design decisions, uncommented and unreadable code. Your comments on your paper should be able to demonstrate your proper understanding of the code.

- Working code: 60 points
- Coding style (refer to the guideline): 20 points
- Your paper code's comments: 20 points

Late submission will not be allowed and if you have multiple versions of your code file, make sure you do submit the correct version. Only the version that is submitted before the due will be graded.

MyListInterface interface has been provided to you and you can find about the methods to be implemented and their descriptions in the interface.