CS2110 Homework 8 - Spring 2021
Recursion

You will not need code from a previous assignment to complete this homework. You will develop solutions to several problems using a recursive approach.

First, set up your project:

- Create a new Java project on Eclipse, and call it "Homework 8."

Learning Goals

In this assignment, we will practice:

1. Using a recursive approach to solve problems.

Implementing Recursive Methods

Implement recursive solutions for each of the following problems. Create a class `RecursiveTools` to hold the following utility methods.

Note: For each of these problems, an iterative solution exits. However, you will get no points for an iterative solution. A solution is recursive because at some point, the recursive method *will* call itself. (*This is a clue!* If your method does not call itself at some point, you may have taken an iterative approach. Be sure to collaborate on strategy with your cohort and get feedback.)

1. Write a recursive method that accepts a String as an argument and returns a `String` with the characters in reverse order.
   `public String reverseString(String input) {  }`

2. Write a recursive method that accepts a `List` of `Strings` and returns the number of items in the list.
   `public int sizeOfList(List<String> l) {  }`

3. Write a recursive method that accepts two `Strings` and returns `true` if the two strings are equal, `false` otherwise. You may **not** use `String`'s `.equals()` method.
   `public boolean stringEqualityTest(String s1, String s2) {  }`

4. Write a recursive method that accepts two `ArrayList<String>` arguments and returns `true` if the two lists are equal. Use your `stringEqualityTest` method from problem 3 to test the `Strings`. Note you must compare the lists *without* changing them - you may need to use a helper method. You may **not** use `ArrayList`'s `.equals()`

method.

```
public boolean listEqualityTest(ArrayList<String> l1,
                                          ArrayList<String> l2) {  }
```

5. Using the methods you have already created, write a set of methods that use mutual recursion to return `true` if and only if every other item in the list, beginning with the first item, is the reverse of the item in that position. You may **not** use `ArrayList`'s `.equals()` method. The method should return true if the lists are empty.

```
public boolean listCustomComparison(ArrayList<String> l1,
                                             ArrayList<String> l2) {  }
```

Main Method (or JUnit) Testing

In this homework, we expect you to do your own testing before submitting to Gradescope.

Note: Your main method should have enough testing to provide sufficient evidence to determine that the behavior implemented matches the behavior described above. It is *highly recommended* to write the tests before implementation as it will help you to understand the exact behavior and what is the expected output of different inputs.

Our goal in this assignment is to encourage main method testing so that you know your code works well before submitting to Gradescope. Therefore, you will only be able to submit to Gradescope a total of 10 times for this assignment.

Submission Information

Method and Class Naming: You must match method names, instance variable names, and data types exactly. You must use correctly formatted Java code. Declare fields in the class definition, and create a default constructor for each class that initializes every instance variable.

Coding Style: In real-world software development, it is paramount to create readable and easily maintainable code. That is typically achieved through the use of style and commenting guidelines. Since you will be updating this code over the next few weeks, we have provided a style guide and formatting guide that we strongly encourage you to follow:

- Coding Style Guide (includes installation instructions for Eclipse)
- Eclipse Style File

Submitting: Upload your Eclipse project (the `.java` file) to the "Homework 8 - Recursion" assignment on Gradescope. You should submit `RecursiveTools.java`. This submission utilizes an autograder to check that your code follows these specifications. If it spots an issue, it will alert you, but you should **NOT** use the submission system as your testing. We encourage you to test your code during the implementation phase. Therefore, you may upload your code a **maximum of ten (10) times**.

Note: After the 10th submission, Gradescope will still allow submissions, but they will NOT be graded by the system.

Grading Rubric

The assignment will be worth a total of 100 points:
15 points - Recursive solution for problem 1
15 points - Recursive solution for problem 2
15 points - Recursive solution for problem 3
15 points - Recursive solution for problem 4
25 points - Recursive solution for problem 5
15 points - Code readability (organized, well-indented, readable by others)