Lab 24 (December 4 or December 5)

Instructions: Complete the exercises below. Be sure to show your code to one of the lab TAs before you leave, so that you can receive credit for this lab. You must also upload a copy of all your source code (`.java`) files through Blackboard by 11:59 PM on Tuesday, December 5.

1. Define a method named `reverse()` that uses recursion to reverse the order of the characters in a `String`. For example, the reverse of “camouflage” is “egalfuomac”. Note that an empty or 1-character `String` is automatically its own reverse; otherwise, extract the first and last characters and move them around the reverse of the `String`’s “interior” characters (so, for example, the first step in reversing “cottage” would be to rearrange ‘c’ and ‘e’ around the reverse of “ottag”).

   Your method should have the following header:

   ```
   public static String reverse (String original)
   ```

   Hint: For the sake of space efficiency (by ***not*** creating a potentially-large number of intermediate substrings), use a recursive helper method like the following:

   ```
   private static String reverse (String source, int lowIndex,
   int highIndex)
   ```

   This method does not (partially) duplicate `source` as it works; instead, it updates the indices and extracts substrings as needed. In this implementation, the original `reverse()` method just calls its helper with 0 and (length() – 1) as the initial indices.

2. Define a recursive method named `findLargest()`, which locates and returns the largest value in an array of integers. It may be more efficient to use the `max()` helper method below (compare the current element to the largest element in the rest of the array):

   ```
   public static int max(int a, int b)
   {
       if (a > b) { return a; }
       else { return b; }
   }
   ```

   Use the following method header for `findLargest()`:

   ```
   public static int findLargest (int [ ] list, int currentIndex)
   ```

   Hint: Start at one end of the array, and make the opposite end’s index into your base case (for example, starting at index 0, the base case occurs when you reach index (length – 1)).

Grading Guidelines: This lab is graded on a scale of 0-3 points, assigned as follows:

0 points: Student is absent or does not appear to have completed any work for the lab

1 point: Student has written only one program, but it does not compile or run at all due to errors.

2 points: Student has written (or attempted to write) both programs, but only one compiles and runs without error.

3 points: Student has written both programs, and they both compile and run correctly, without any apparent errors.