Lab 16 (November 1 or November 2)

Instructions: Complete the steps 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 to the link on Blackboard by 11:59 PM on Thursday, November 2.

1. Define a class named `Color` that contains three `private int` instance variables, representing the red, green, and blue components that make up the color. Each variable has a value between 0 and 255 (inclusive), representing the contribution of that color component to the overall color (for example, a red value of 200 would mean a high amount of red in the color, while a blue value of 50 would represent a low amount of blue in the color). The class also has a `private String` instance variable that represents the name of the color (e.g., "salmon" or "teal").

   The `Color` class has a constructor that takes three integer arguments, representing the color's red, green, and blue components; it assigns the value `null` (**NOTE:** this is a literal value, **NOT** the string “null”) to the color name. A second constructor takes three integers and a `String` as arguments, and assigns them to the appropriate instance variables. Finally, the class has `public` accessor methods for all four variables.

2. Next, define a subclass of `Color` named `AlphaChannelColor`. An `AlphaChannelColor` has one additional `private` instance variable: an integer that represents the color’s degree of transparency (0 is fully opaque, and 255 is fully transparent). Define two constructors for this class; one takes four integers representing the red, green, blue, and opacity values, while the second constructor also takes a String argument representing the name of the color (note that each constructor will need to invoke one of the superclass constructors as its first line).

   Finally, override the `getColorName()` accessor method that you inherited from `Color` as follows:

   a. The new version of `getColorName()` should return the name of the color, prefixed by either "opaque", "semi-transparent", or "transparent" (e.g., "opaque teal"). **Hint:** use `Color`'s `getColorName()` method to get the original name.

   b. If the transparency value is less than 100, the prefix should be "opaque". If it is between 100 and 199, the prefix should be "semi-transparent". If the transparency value is 200 or greater, the prefix should be "transparent".

   c. If the color name is `null`, substitute "color" for the color name (e.g., "semi-transparent color").

For example, an `AlphaChannelColor` with a transparency value of 215 and a color name of "salmon" would cause the method to return "transparent salmon". An `AlphaChannelColor` with a transparency value of 55 and a color name of `null` would cause the method to return "opaque color".

3. Finally, implement a driver class that creates at least one `Color` instance and at least one `AlphaChannelColor` instance, and demonstrates that their accessor methods work correctly (i.e., by printing out the result of calling each accessor method on each instance). Your driver can either collect the starting values for each object from the user or use hard-coded values.

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 completed some work, but the program does not compile or run due to errors.

2 points: Student has correctly completed only one of the first two classes (`Color` or `AlphaChannelColor`), plus the driver class.

3 points: Student has correctly completed all three classes, without any apparent errors.