CS 129 HW2 Custom Colors

Write a program with three classes: CustomColor, Palette, and a Driver class. Do not use Java's built-in Color class for this assignment (though we will be using it later on).

CustomColor allows users to specify an RGB color. It has three integer fields: red, green, and blue, and each can have a value between 0 and 255. You should set these in the constructor, and print an error if any of the values are outside the range of 0-255.

You should also have the following methods:

- "blend" takes another CustomColor as an argument, and returns a new CustomColor that's the average of the two color's rgb values.
  - For example if colorA is (100, 0, 100) and colorB is (50, 100, 200), then colorA.blend(colorB) returns (75, 50, 150)
- "lighten" returns a new CustomColor that's brighter, by multiplying R, G, and B values by 1.2.
  - If a value would exceed 255, clamp it at 255 instead.
- "darken" returns a new CustomColor that's darker, by multiplying R, G, and B values by 0.8
- "isLighter" takes a CustomColor as an argument, and returns true if the R+ G+ B values of this color are larger than those of the other color, false otherwise. Ties return false.
  - For example if colorA is (100, 0, 100) and colorB is (50, 100, 200), then colorA.isLighter(colorB) returns false, because 100+0+100 is smaller than 50+100+200
- "toString" returns a String representation of the CustomColor, in the form:
  - "Red: XXX Green: XXX Blue: XXX"
  - Where XXX is the red, green, and blue value respectively.

The Palette class manages an array of five CustomColors. You should have three constructors:

- No arguments, which sets all the CustomColors in the array to white (255, 255, 255)
- Two arguments, a start and end CustomColor, which evenly generates a spectrum of colors between those two. For example Palette(white, black) makes an even greyscale, with (191, 191, 191), (127, 127, 127), and (63, 63, 63) between them.
- One argument, an array CustomColors, which sets the array that the Palette manages to the CustomColors provided. You can assume the array will always be five items long.

You should also have the following methods:

- "printPalette" prints all five colors, each on its own line, with a blank line at the end
- "getAverageColor" returns a new CustomColor that is the average of the red, green, and blue values in the palette.

Some sample code you can use for testing in your Driver should be attached along with this assignment. The expected output is also provided. You should also try out some other values, as I'll use a different main method to test your code.

Extra Credit (up to 5%) for exceptional documentation, especially good method and class headers.

```
	public static void main(String args[]) {
		CustomColor red = new CustomColor(255, 0, 0);
		CustomColor green = new CustomColor(0, 255, 0);
		CustomColor blue = new CustomColor(0, 0, 255);
		CustomColor white = new CustomColor(255, 255, 255);
		CustomColor black = new CustomColor(0, 0, 0);
		CustomColor tealish = new CustomColor(50, 150, 150);

		CustomColor orange = red.blend(green);
		System.out.println(orange.toString());
		System.out.println();

		CustomColor purple = green.blend(blue);
		System.out.println(purple.toString());
		System.out.println();

		CustomColor darkPurple = purple.darken();
		System.out.println(darkPurple);
		System.out.println("Is lighter than purple: " +
darkPurple.isLighter(purple));
		System.out.println();

		Palette lightToDark = new Palette(white, black);
		lightToDark.printPalette();

		Palette variety = new Palette();
		variety.printPalette(); //all white (255, 255, 255)

		variety.colors[1] = variety.colors[1].blend(darkPurple);
		variety.colors[2] = orange.darken();
		variety.colors[3] = purple.lighten();
		variety.colors[4] = black;
		variety.printPalette();

		Palette monochromeTealish = new Palette(
				new CustomColor[] {
					tealish,
					tealish.lighten(),
					tealish.lighten().lighten(),
					tealish.lighten().lighten().lighten(),
					tealish.lighten().lighten().lighten().lighten()
				}
				);
		monochromeTealish.printPalette();
		System.out.println(monochromeTealish.getAverageColor().toString());
	}
```

```
Red: 127 Green: 127 Blue: 0

Red: 0 Green: 127 Blue: 127

Red: 0 Green: 101 Blue: 101
Is lighter than purple: false

Red: 255 Green: 255 Blue: 255
Red: 191 Green: 191 Blue: 191
Red: 127 Green: 127 Blue: 127
Red: 63 Green: 63 Blue: 63
Red: 0 Green: 0 Blue: 0

Red: 255 Green: 255 Blue: 255
Red: 255 Green: 255 Blue: 255
Red: 255 Green: 255 Blue: 255
Red: 255 Green: 255 Blue: 255
Red: 255 Green: 255 Blue: 255

Red: 255 Green: 255 Blue: 255
Red: 127 Green: 178 Blue: 178
Red: 101 Green: 101 Blue: 0
Red: 0 Green: 152 Blue: 152
Red: 0 Green: 0 Blue: 0

Red: 50 Green: 150 Blue: 150
Red: 60 Green: 180 Blue: 180
Red: 72 Green: 216 Blue: 216
Red: 86 Green: 255 Blue: 255
Red: 103 Green: 255 Blue: 255

Red: 74 Green: 211 Blue: 211
```