Deliverable 5

For designing our faults, we implemented faults that would fail for most tests, but not all.

ContrastChecker.java: getConstrastRatio()

Original line: `if (fgLuminosity > bgLuminosity) {`

Fault: `if (fgLuminosity < bgLuminosity) {`

This changes greater than sign to less than, changing the ratio returned. The intended returned ratio is larger/smaller this changes it to smaller/larger. When the colors are the same, the ratio returned will be the same.

Simply comment out the original line and then uncomment the fault to implement.

ColorConverter.java: rgb2Hex()

Original line:

```
return (String.format("#%02x%02x%02x", color.getRed(),
color.getGreen(), color.getBlue())).toUpperCase();
```

Fault:

```
return (String.format("#%02x%02x%02x", color.getBlue(),
color.getGreen(), color.getRed())).toUpperCase();
```

This fault changes the order of values resulting in a different hex value, unless the values are the same.

Simply comment out the original line and then uncomment the fault to implement.

DistanceCalculator.java: calculate()

Original line:

```
return (double)
Math.round(Math.abs((Math.cbrt(Math.pow(Double.valueOf(colorToChange.g
etRed()) - Double.valueOf(colorToKeep.getRed()), CUBIC) +
Math.pow(Double.valueOf(colorToChange.getGreen()) -
Double.valueOf(colorToKeep.getGreen()), CUBIC) +
Math.pow(Double.valueOf(colorToChange.getBlue()) -
Double.valueOf(colorToKeep.getBlue()), CUBIC)))) * ROUND_VALUE) /
ROUND_VALUE;
```

Fault:

```
return (double)
Math.round(Math.abs((Math.sqrt(Math.pow(Double.valueOf(colorToChange.g
etRed()) - Double.valueOf(colorToKeep.getRed()), 2) +
Math.pow(Double.valueOf(colorToChange.getGreen()) -
Double.valueOf(colorToKeep.getGreen()), 2) +
Math.pow(Double.valueOf(colorToChange.getBlue()) -
Double.valueOf(colorToKeep.getBlue()), 2)))) * ROUND_VALUE) /
ROUND_VALUE;
```

For this "fault," because the original code is already incorrect, the code is altered to actually give the correct result as per the euclidean distance formula

Simply comment out the original line and then uncomment the "fault"/fix to implement.

ColorConverter.java: getHue()

Original line: `Color.RGBtoHSB(color.getRed(), color.getGreen(), color.getBlue(), hsbValues);`

Fault: `Color.RGBtoHSB(color.getRed(), color.getBlue(), color.getGreen(), hsbValues);`

It changes the order of the RGB values to return an incorrect answer unless the values are the same.

Simply comment out the original line and then uncomment the fault to implement.

ColorConverter.java: offsetRgbColor()

Original Line: `return new Color(bgColor.getRed() + offsetRed, bgColor.getGreen() + offsetGreen, bgColor.getBlue() + offsetBlue);`

Fault: `return new Color((bgColor.getBlue() + offsetBlue)%256, (bgColor.getGreen() + offsetGreen)%256, (bgColor.getBlue() + offsetBlue)%256);`

It changes which offsets are added to which color value. It also checks to see if the sum of the values goes above 255 and wraps it which could alter the color. The tests will fail unless the modulated sums are equivalent to the solution.

Simply comment out the original line and then uncomment the fault to implement.