Team CargoPants

Isabel Lally Collin Bauer Dylan Evans

The Testing of Tanaguru Contrast Finder

CSCI 362

Software Engineering

Dr. Jim Bowring

Table of Contents

Introduction

Our group was tasked with picking a project from a selected group of HFOSS projects and test their code and inject faults to see how it reacts with errors. We decided to go with the Tanaguru Contrast Finder project which is a program that takes a color and converts it to a contrasting color which allows color blind people to see colors better. These programs are helpful because they allow colorblind people to see websites and their computers better.

Our project involves writing our own test cases and testing specific methods to see if the outputs match what we expect and to inject faults into their code to see how they handle exceptions. By doing this we intend to learn how to work from the command line in Linux and write proper test cases. We are also using bash to run scripts so we are becoming more familiar with that as well.

CHAPTER 1

Our chosen project is Tanaguru Contrast Finder, which is software that finds the best color contrast for people with vision impairment.

We started by cloning our HFOSS project, Tanaguru Contrast Finder, from https://github.com/Tanaguru/Contrast-Finder. The project was already set up to build using Maven. On our virtual machines, we used sudo apt install maven in the terminal. We compiled and tested Contrast Finder by using the command mvn clean && mvn package. The project has the Surefire plugin to run unit tests. The outputs from those tests are below and include which test classes were run and if there were any failures, errors, or skipped tests. None of the unit tests ran failed or had errors and no tests were skipped.

Experiences

We had a lot of trouble getting our virtual machines to work at first. Our chosen distribution was Lubuntu, a lightweight flavor of Ubuntu. However, installing it on VirtualBox didn't work at first, so we switched to VMWare (at least for the time being.) It works, and we were able to get all of the dependencies and the proper version of Java (1.7) to run the project.

Maven test results

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.opens.utils.contrastchecker.ContrastCheckerTest
getConstrastRatio
result :3.6102927852355164
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.026 sec
Running org.opens.utils.colorconvertor.ColorConverterTest
getHue
java.awt.Color[r=128,g=128,b=127]
Hue : java.awt.Color[r=128,g=128,b=127]
Rgb2hexBlack
Rgb2hexWhite
Rgb2hexPink
Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.008 sec
```

```
Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0


-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.opens.contrast.finder.api.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.017 sec
Running org.opens.colorfinder.AbstractColorFinderImplTest
findColorsWithValidContrastWithForegroundTested
findColorsWithInvalidContrastWithBackgroundTested
findColorsWithInvalidContrastWithForegroundTested
findColorsWithValidContrastWithBackgroundTested
Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.058 sec

Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0


-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.opens.color.result.impl.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.02 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0


-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.opens.color.finder.hsv.ColorFinderHsvPsychoTest
FindColorsNearColor
11-09-2019 16:312957 0    INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - SetBounderHue 15.0
11-09-2019 16:312963 6    INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - findColors of
ColorFinderHsvPsycho
11-09-2019 16:312964 7    INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=165,b=0]
11-09-2019 16:313319 362  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=161,b=0]
11-09-2019 16:313444 487  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=157,b=0]
11-09-2019 16:313530 573  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=153,b=0]
11-09-2019 16:313579 622  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=149,b=0]
11-09-2019 16:313633 676  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=145,b=0]
11-09-2019 16:313671 714  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=141,b=0]
11-09-2019 16:313696 739  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=137,b=0]
```

```
11-09-2019 16:313712 755  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=133,b=0]
11-09-2019 16:313727 770  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=129,b=0]
11-09-2019 16:313742 785  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=125,b=0]
11-09-2019 16:313758 801  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=121,b=0]
11-09-2019 16:313774 817  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=117,b=0]
11-09-2019 16:313796 839  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=113,b=0]
11-09-2019 16:313812 855  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=109,b=0]
11-09-2019 16:313827 870  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=105,b=0]
11-09-2019 16:313842 885  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=165,b=0]
11-09-2019 16:313857 900  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=169,b=0]
11-09-2019 16:313872 915  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=173,b=0]
11-09-2019 16:313887 930  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=177,b=0]
11-09-2019 16:313902 945  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=181,b=0]
11-09-2019 16:313918 961  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=185,b=0]
11-09-2019 16:313933 976  INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=189,b=0]
11-09-2019 16:313963 1006 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=193,b=0]
11-09-2019 16:313993 1036 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=197,b=0]
11-09-2019 16:31428 1071 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=201,b=0]
11-09-2019 16:31448 1091 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=205,b=0]
11-09-2019 16:31463 1106 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=209,b=0]
11-09-2019 16:31478 1121 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=213,b=0]
11-09-2019 16:31493 1136 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=217,b=0]
11-09-2019 16:314108 1151 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=221,b=0]
11-09-2019 16:314124 1167 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsycho  - Change
Huejava.awt.Color[r=255,g=225,b=0]
11-09-2019 16:314140 1183 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - 38
11-09-2019 16:314144 1187 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #CC8609
11-09-2019 16:314144 1187 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C6891E
11-09-2019 16:314144 1187 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C38A26
11-09-2019 16:314145 1188 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D28300
11-09-2019 16:314145 1188 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #CC8615
11-09-2019 16:314145 1188 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D2830C
11-09-2019 16:314145 1188 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #CC8621
11-09-2019 16:314146 1189 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C98727
```

```
11-09-2019 16:314146 1189 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C38A2B
11-09-2019 16:314146 1189 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D58118
11-09-2019 16:314146 1189 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #CF8429
11-09-2019 16:314147 1190 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C98730
11-09-2019 16:314147 1190 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C68838
11-09-2019 16:314148 1191 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D58124
11-09-2019 16:314148 1191 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D87F21
11-09-2019 16:314148 1191 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D87F24
11-09-2019 16:314148 1191 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #DB7E03
11-09-2019 16:314149 1192 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D87F27
11-09-2019 16:314149 1192 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D87F2A
11-09-2019 16:314149 1192 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D28236
11-09-2019 16:314149 1192 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #DE7C06
11-09-2019 16:314150 1193 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D87F2C
11-09-2019 16:314150 1193 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #D87F2F
11-09-2019 16:314150 1193 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E47800
11-09-2019 16:314150 1193 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E17A09
11-09-2019 16:314151 1194 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #DE7C12
11-09-2019 16:314151 1194 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E77603
11-09-2019 16:314151 1194 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E4780C
11-09-2019 16:314151 1194 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E17A15
11-09-2019 16:314152 1195 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #EA7406
11-09-2019 16:314152 1195 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E7760F
11-09-2019 16:314152 1195 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #E47818
11-09-2019 16:314153 1196 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #DB7D2B
11-09-2019 16:314153 1196 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #F36D00
11-09-2019 16:314153 1196 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #F06F09
11-09-2019 16:314153 1196 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #F66A00
11-09-2019 16:314154 1197 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C68912
11-09-2019 16:314154 1197 INFO  org.opens.color.finder.hsv.ColorFinderHsvPsychoTest  - Color found #C68906
Key
11-09-2019 16:314157 1200 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.351 sec
Running org.opens.color.finder.hsv.ColorFinderHsvTest
FindColorsWithFgAndBg
11-09-2019 16:314173 1216 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314173 1216 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=127,g=127,b=127]java.awt.Color[r=128,g=128,b=128]0
FindColorsWithFgAndBg2
11-09-2019 16:314184 1227 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314184 1227 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=128,g=128,b=128]java.awt.Color[r=127,g=127,b=127]0
FindColorsBornBlack
11-09-2019 16:314196 1239 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314196 1239 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=1,g=1,b=1]java.awt.Color[r=3,g=3,b=3]0
FindColorsBornBlack2
11-09-2019 16:314197 1240 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314197 1240 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=3,g=3,b=3]java.awt.Color[r=1,g=1,b=1]0
FindColorsBornWhite
```

```
11-09-2019 16:314198 1241 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314198 1241 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=254,g=254,b=254]java.awt.Color[r=255,g=255,b=255]0
FindColorsBornWhite2
11-09-2019 16:314205 1248 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314205 1248 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=255,g=255,b=255]java.awt.Color[r=254,g=254,b=254]0
FindColorsGreenAlert
11-09-2019 16:314221 1264 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314222 1265 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=70,g=136,b=71]java.awt.Color[r=223,g=240,b=216]0
FindColorsGreenAlert2
11-09-2019 16:314268 1311 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314269 1312 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=223,g=240,b=216]java.awt.Color[r=70,g=136,b=71]0
FindColorsNearColor
11-09-2019 16:314288 1331 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314288 1331 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=0,g=0,b=0]java.awt.Color[r=1,g=0,b=1]0
FindColorsNearColor2
11-09-2019 16:314296 1339 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
11-09-2019 16:314297 1340 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - Appel de findColors, couleurs
: java.awt.Color[r=255,g=255,b=255]java.awt.Color[r=255,g=165,b=0]0
Key
11-09-2019 16:314318 1361 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
Tests run: 11, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.147 sec
Running org.opens.color.finder.hsv.ColorFinderRgbTest
FindColorsNearColor
11-09-2019 16:315694 2737 INFO  org.opens.color.finder.hsv.ColorFinderRgbTest  - 0
Key
11-09-2019 16:315695 2738 INFO  org.opens.color.finder.hsv.ColorFinderHsv  - instanciation of
ColorFinderHsv class
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.362 sec

Results :

Tests run: 15, Failures: 0, Errors: 0, Skipped: 0
```

CHAPTER 2

Experiences

We struggled with deliverable 2 for several reasons. We still feel a little unsure about the Test Template as different people on our teams had varied interpretations of what that meant. We were initially searching for a class that had at least five testable methods, which we never found, but we were informed that the methods didn't have to all come from the same file. We finally chose a file to at least base the first five test cases on, but we didn't think through needing a method that takes input. To find methods to test, we chose ColorCombinaisonImpl.java and decided on the method public boolean equals(Object obj).

Update

Since the date of deliverable 2 being due, we were informed that the method we chose was not appropriate for testing since it is a Java method, not specific to the Tanaguru's Contrast Finder. Additionally, we have since needed to update our testTemplate.txt file for ease of use with our script.

Report

Once we chose ColorCombinaisionImpl and equals(obj), it became clear on what we need to test. This method checks that the color combination that is being suggested is valid. It will fail if the object is null, isn't a ColorCombinaisionImpl object, or the colors differ between objects. The method will otherwise pass.

Our five tests on this method should test the three ways to return false and cases to return true. Case 1 includes the comparison object being tested against itself, which should return true. Case 2 will be the correct object type with the same values as the comparison object, which we expect to return true. Case 3 will be the correct object, but with color

values that differ from the comparison object, which will return false. Case 4 will pass null into the method, which will return false. Case 5 will return false, since the input won't even be the correct object type.

Update

For the new first five test cases, we chose the method Rgb2Hex, which takes in a Color, meant to be defined by rgb values and returns the hex value of the color. The first test case, takes a random rgb value and returns the correct hex value. The second and third cases test the ends of the color values, testing black with an rgb value of 0,0,0 and a hex value of 000000 and white with an rgb value of 255,255,255 and a hex value of FFFFFF respectively. The fourth case tests invalid integers out of range of rgb values and will return an error. The fifth case tests strings as an invalid input and will also return an error.

Below is the both the original test plan, followed by the updated test plan we devised after receiving feedback.

Original Test Plan

The Testing Process

We will have five test cases per method and we will have five methods to test.

Requirements Traceability

- Test Case1: The object constructed from itself will equal an input of itself, returning true.
- Test Case2: The input of the ColorCombinaisonImpl(with same parameters as self) will return true.
- TestCase3: The input of the ColorCombinaisonImpl(with different parameters as self) will return false.

- TestCase4: The input null will return false.
- TestCase5: Input of an object other than ColorCombinaisonImpl (String in this test case) will return false.

Tested Items

ColorCombinaisonImpl.java – method: equals(Object obj)

Testing Schedule

- 10/01/19 - Design 5 test cases on one method
- 10/18/19 - Create drivers for first 5 test cases and design 10 more test cases
- 10/25/19 - Create drivers for test cases 6-15 and design 5 more test cases
- 10/29/19 - Design and build test case automation
- 11/01/19 - Create drivers for test cases 16 - 20 and design 5 more test cases
- 11/08/19 - Create drivers for test cases 20 - 25
- 11/12/19 - Design 20 additional test cases
- 11/19/19 - Design and inject 5 faults into code so that at least 5 test cases fail
- Run all tests as they are created and run previous tests.

Test Recording Procedures

The output of the tests will be written to a file that we will use diff against testCaseOracle files to determine if the tests pass or fail.

Constraints

Time is the biggest constraint working against us. Between classes and work, finding time to work together can be difficult.

Original Test Cases

In these test cases, the comparison object will be constructed as ColorCombinaisonImpl(Color(0,0,255), Color(255,0,0), 50).

#T	01	1. Test ID
	02	2. Requirement being tested
	03	3. Component being tested
	04	4. Method being tested
	05	5. Test input(s) including command line argument(s)
	06	6. Expected outputs
#1	01	1. 01
	02	2. The object itself, equals itself.
	03	3. ColorCombinaisonImpl
	04	4. equals(obj)
	05	5. this
	06	6. true
#2	01	1. 02
	02	2. The object must be of type ColorCombinaisonImpl and valid colors.
	03	3. ColorCombinaisonImpl
	04	4. equals(obj)
	05	5. ColorCombinaisonImpl(Color(0,0,255), Color(255,0,0), 50)
	06	6. true
#3	01	1. 03
	02	2. The object must be of type ColorCombinaisonImpl and valid colors.
	03	3. ColorCombinaisonImpl
	04	4. equals(obj)
	05	5. ColorCombinaisonImpl(Color(255,0,0), Color(0,0,255), 50)
	06	6. false
#4	01	1. 04
	02	2. Null is not a valid object.
	03	3. ColorCombinaisonImpl
	04	4. equals(obj)
	05	5. null
	06	6. false
#5	01	1. 05
	02	2. Only the ColorCombinaisonImpl object is valid
	03	3. ColorCombinaisonImpl
	04	4. equals(obj)
	05	5. String "test"
	06	6. false

Updated Test Plan

The Testing Process

We will have five test cases per method and we will have five methods to test.

Requirements Traceability

- Test Case1: The method will take a valid color defined by rgb and change it to hex
- Test Case2: The rgb value equating to black will output a hex value with all zeros.
- Test Case3: The rgb value equating to white will output a hex value with all Fs.
- Test Case4: The method will not allow rgb values out of range.
- Test Case5: This method will not accept strings as valid arguments.

Tested Items

ColorConverter.java – method: rgb2Hex(Color color)

Testing Schedule

- 10/01/19 - Design 5 test cases on one method
- 10/18/19 - Create drivers for first 5 test cases and design 10 more test cases
- 10/25/19 - Create drivers for test cases 6-15 and design 5 more test cases
- 10/29/19 - Design and build test case automation
- 11/01/19 - Create drivers for test cases 16 - 20 and design 5 more test cases
- 11/08/19 - Create drivers for test cases 20 - 25
- 11/12/19 - Design 20 additional test cases
- 11/19/19 - Design and inject 5 faults into code so that at least 5 test cases fail
- Run all tests as they are created and run previous tests.

Test Recording Procedures

- The output of the tests will be written to a file that we will use diff against testCaseOracle files to determine if the tests pass or fail.

Constraints

- Time is the biggest constraint working against us. Between classes and work, finding time to work together can be difficult.

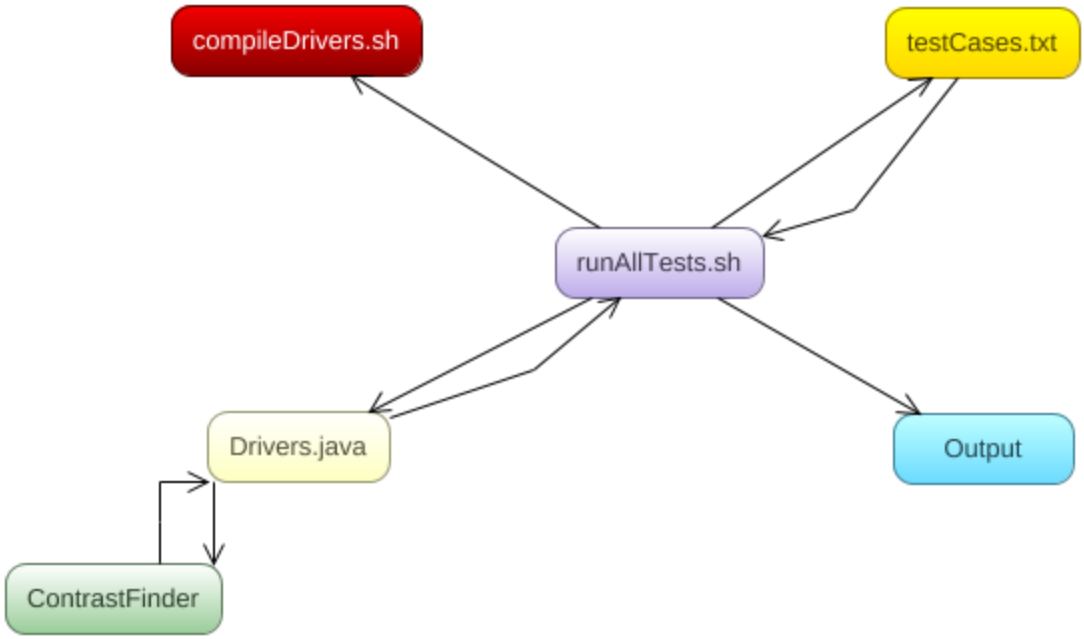
Updated Test Cases

```
#T  01  Test ID ##                                          <-- read by script
    02  Requirement being tested
    03  # Reserved for comments
    04  package.path.to.Component.method()
    05  DriverName                                          <-- read by script
    06  Test input(s) including command line argument(s)    <-- read by script
    07  Expected outputs                                    <-- read by script

#1  01  01
    02  The method will take a valid color defined by rgb and change it to hex
    03
    04  org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()
    05  Rgb2HexDriver
    06  220 136 15
    07  #DC880F

#2  01  02
    02  The rgb value equating to black will output a hex value with all zeros.
    03
    04  org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()
    05  Rgb2HexDriver
    06  0 0 0
    07  #000000

#3  01  03
    02  The rgb value equating to white will output a hex value with all Fs.
    03
    04  org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()
    05  Rgb2HexDriver
    06  255 255 255
    07  #FFFFFF

#4  01  04
    02  The method will not allow rgb values out of range.
    03
    04  org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()
```

```
    05  Rgb2HexDriver
    06  300 -15 2
    07  Exception
```

```
#5  01  05
    02  This method will not accept strings as valid arguments.
    03
    04  org.opens.utils.colorconvertor.ColorConverter.rgb2Hex()
    05  Rgb2HexDriver
    06  "cat" "meow" "dog"
    07  Exception
```

CHAPTER 3



Driver Rationale

- **Rgb2HexDriver.java** - takes three arguments (rgb) and parses as ints, constructs new color object to be inserted into rgb2Hex method and prints output.
- **Hex2RgbDriver.java** - takes one argument (hex value) and uses Color.decode to construct a new color object to be inserted into hex2Rgb method and prints output.
- **ComputeContrastDriver.java** - takes two arguments (doubles) and parses as doubles, inserts the two arguments into the computeContrast method and prints output.
- **OffsetRgbColorDriver.java** - takes six arguments (rgb and offsets for each) and parses all as ints, constructs a new color object from the rgb values and inserts that as well as the three offsets into the offsetRgbColor method and prints output.

- **CalculateDriver.java** - takes either two or six arguments depending on if using hex values or rgb values to construct two new color objects to insert into the method Calculate

Scripts

- **compileDrivers.sh** – reads all of the drivers in our test folder and compiles them using the default Java compiler
- **runSingleTest.sh** – takes a filename as input, reads through the file, and tests a method as specified by our test case template with a detailed pass/fail output
- **runAllTests.sh** – runs compileDrivers.sh, forms an array that represents each defined test case file, and passes each individual filename to runSingleTest.sh

Experiences

Compiling the source code from the command line was difficult to understand, as maven has its package structure defined by various pom.xml files in separate modules. We had to manually combine these modules together for the base Java compiler to read and compile properly.

Once the source code could compile, we needed to figure out how to make our test drivers read from their source code and test the necessary methods. We had trouble finding out how to make our code play nicely with the Java package structure, but eventually we tried something which ended up compiling properly. We were extremely relieved with this accomplishment.

After this, we decided to move into Linux and start building our scripts. Collin studied some Bash scripting tutorials and ended up building several scripts which automate the tasks required to test our drivers. It took time, but was fairly intuitive.

After building a demo driver to run a single test case, we encountered some unexpected behavior when scaling it for different drivers. The test files were not being read

correctly, despite following the template properly. After some research into how Bash works, we discovered this was due to line ending differences between Windows (where our test cases were written) and Linux (where we were running the tests). A workaround was to rewrite these test cases in our virtual machines, but in the future we can use git's built-in setting "autocrlf" which converts between the two conventions for us.

We made some additional errors while working on automation. Originally we had stored all of our testing and automation in a folder called "TeamAutomation" instead of "TestAutomation", so we had to refactor our file structure, causing some merge conflicts. We also had issues understanding where our scripts would run from, but eventually made it so they would only run from the TestAutomation folder, and nowhere else.

Finally, we have been unsure of how exceptions should be handled in our oracles. After some debate, we decided to use try/catch blocks in our drivers, and use Exception.getMessage() as the output to compare against the oracle. This has not been implemented to each driver yet.

CHAPTER 4

Progress and Experiences

We had already made our 25 test cases but not all worked properly and we had significant troubles with the ones that resulted in an error. We debated on how we should handle these exceptions with try catch blocks and still pass in illegal arguments into the method being tested. We decided on isolating the method in its own try block and doing a catch that would handle any exception that would rise. This also caused us to have slightly alter certain test cases so that we would not be injecting errors.

Test cases 21-25 do actually fail. These test the method calculate in distanceCalculator, which calculates the Euclidean distance between two colors. In the comments above the method, this link is given for the formula they use:

http://en.wikipedia.org/wiki/Euclidean_distance#Three_dimensions

This link takes you to the three-dimensional Euclidean distance formula:

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}.$$

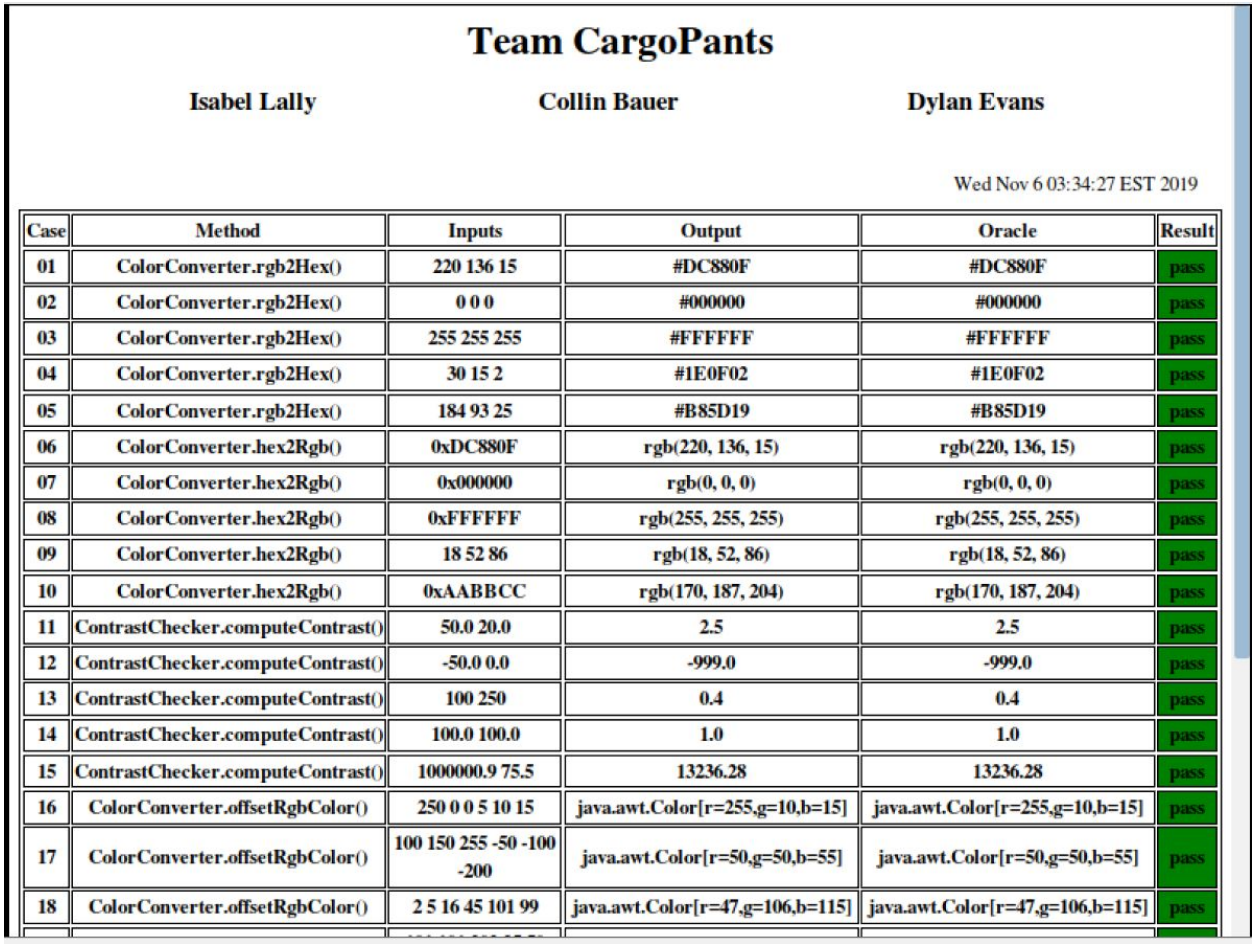
This is how the oracles of the five test cases were decided, however, the output values are different. Our calculations were double checked for any mistakes, just in case. The reason for the failures appears to be that instead of squaring the difference of the rgb values between colors, the developers cubed the values.

HTML output from runAllTests

As is, our scripts did run properly and gave the results from our tests, but they had to be altered so it would output to an HTML file. This required a lot of rewriting and the creation of a template which our scripts would reference and build an output HTML from.

This was done by using sed to read lines from the template and parsing it with regular expressions to replace an internal portion of it with the desired output.

Example output



Team CargoPants

Isabel Lally **Collin Bauer** **Dylan Evans**

Wed Nov 6 03:34:27 EST 2019

Case	Method	Inputs	Output	Oracle	Result
01	ColorConverter.rgb2Hex()	220 136 15	#DC880F	#DC880F	pass
02	ColorConverter.rgb2Hex()	0 0 0	#000000	#000000	pass
03	ColorConverter.rgb2Hex()	255 255 255	#FFFFFF	#FFFFFF	pass
04	ColorConverter.rgb2Hex()	30 15 2	#1E0F02	#1E0F02	pass
05	ColorConverter.rgb2Hex()	184 93 25	#B85D19	#B85D19	pass
06	ColorConverter.hex2Rgb()	0xDC880F	rgb(220, 136, 15)	rgb(220, 136, 15)	pass
07	ColorConverter.hex2Rgb()	0x000000	rgb(0, 0, 0)	rgb(0, 0, 0)	pass
08	ColorConverter.hex2Rgb()	0xFFFFFF	rgb(255, 255, 255)	rgb(255, 255, 255)	pass
09	ColorConverter.hex2Rgb()	18 52 86	rgb(18, 52, 86)	rgb(18, 52, 86)	pass
10	ColorConverter.hex2Rgb()	0xAABBCC	rgb(170, 187, 204)	rgb(170, 187, 204)	pass
11	ContrastChecker.computeContrast()	50.0 20.0	2.5	2.5	pass
12	ContrastChecker.computeContrast()	-50.0 0.0	-999.0	-999.0	pass
13	ContrastChecker.computeContrast()	100 250	0.4	0.4	pass
14	ContrastChecker.computeContrast()	100.0 100.0	1.0	1.0	pass
15	ContrastChecker.computeContrast()	1000000.9 75.5	13236.28	13236.28	pass
16	ColorConverter.offsetRgbColor()	250 0 0 5 10 15	java.awt.Color[r=255,g=10,b=15]	java.awt.Color[r=255,g=10,b=15]	pass
17	ColorConverter.offsetRgbColor()	100 150 255 -50 -100 -200	java.awt.Color[r=50,g=50,b=55]	java.awt.Color[r=50,g=50,b=55]	pass
18	ColorConverter.offsetRgbColor()	2 5 16 45 101 99	java.awt.Color[r=47,g=106,b=115]	java.awt.Color[r=47,g=106,b=115]	pass

CHAPTER 5

Code Changes and Expected Failures

We injected three errors into the color converter class. The first was in the method hex2Rgb(Color color). Within the return strings we changed the order from red, green, blue to blue, red, green. This method is used in test cases 6 to 10 and we expected tests 6, 9, and 10 to fail, but tests 7 and 8 to pass. The reasoning for this is that 7 has a correct output of rgb(0, 0, 0) and 8 has a correct output of rgb(255, 255, 255), so the order does not matter.

The second fault was injected into the method rgb2Hex(Color color). The fault is similar to the one in hex2Rgb, we changed the color order from red, green, blue to green, blue, red. This method is correlated to test cases 1 to 5 and we expected similar results such that 1, 4, and 5 fail and 2 and 3 stay passing as their results are #000000 and #FFFFFF so order does not matter.

The third fault was within the method offsetRgbColor(Color bgcolor, int offsetRed, int offsetGreen, int offsetBlue). Instead of adding each offset color int to the corresponding rgb values of bgcolor, the values are now subtracted. This method is for test cases 16 through 20 and we expected all to fail due to that change.

The final two faults were injected into the contrast checker class. We changed the static variable CONTRAST_FACTOR from 0.05 to 0.5. This change affects the method computeContrast(Double lighter, Double darker). We made a change within the computeContrast method as well. Instead of having the value of the lighter be divided by the darker, they are now multiplied with each other. This method is associated with test cases 11 through 15 and we expected them all to fail after these changes.

We did not want to change the method used in test cases 21 through 25, since several of those test cases failed due to an error already existing within the code.

The results are as expected and shown below.

Case	Requirement	Method	Inputs	Output	Oracle	Resul
01	The method will take a valid color defined by rgb and change it to hex.	ColorConverter .rgb2Hex()	220 136 15	#880FDC	#DC880F	fail
02	The method will take a valid color defined by rgb and change it to hex.	ColorConverter .rgb2Hex()	0 0 0	#000000	#000000	pass
03	The method will take a valid color defined by rgb and change it to hex.	ColorConverter .rgb2Hex()	255 255 255	#FFFFFF	#FFFFFF	pass
04	The method will take a valid color defined by rgb and change it to hex.	ColorConverter .rgb2Hex()	30 15 2	#0F021E	#1E0F02	fail
05	The method will take a valid color defined by rgb and change it to hex.	ColorConverter .rgb2Hex()	184 93 25	#5D19B8	#B85D19	fail
06	The method will take a valid color defined by a hex value and convert to rgb.	ColorConverter .hex2Rgb()	0xDC880F	rgb(15, 220, 136)	rgb(220, 136, 15)	fail
07	The method will take a valid color defined by a hex value and convert to rgb.	ColorConverter .hex2Rgb()	0x000000	rgb(0, 0, 0)	rgb(0, 0, 0)	pass
08	The method will take a valid color defined by a hex value and convert to rgb.	ColorConverter .hex2Rgb()	0xFFFFFF	rgb(255, 255, 255)	rgb(255, 255, 255)	pass
09	The method will take a valid color defined by a hex value and convert to rgb.	ColorConverter .hex2Rgb()	18 52 86	rgb(86, 18, 52)	rgb(18, 52, 86)	fail
10	The method will take a valid color defined by a hex value and convert to rgb.	ColorConverter .hex2Rgb()	0xAABBCC	rgb(204, 170, 187)	rgb(170, 187, 204)	fail
11	This method takes two doubles and computes the contrast between them and outputs in the form of a double.	ContrastChecker .computeContrast()	50.0 20.0	1035.25	2.5	fail
12	This method takes two doubles and computes the contrast between them and outputs in the form of a double.	ContrastChecker .computeContrast()	-50.0 0.0	-24.75	-999.0	fail

	double.					
13	This method takes two doubles and computes the contrast between them and outputs in the form of a double.	ContrastChecker .computeContrast()	100 250	25175.25	0.4	fail
14	This method takes two doubles and computes the contrast between them and outputs in the form of a double.	ContrastChecker .computeContrast()	100.0 100.0	10100.25	1.0	fail
15	This method takes two doubles and computes the contrast between them and outputs in the form of a double.	ContrastChecker .computeContrast()	1000000.9 75.5	7.60001064E7	13236.28	fail
16	This method takes six integers, the first three represent the rgb values of a color	ColorConverter .offsetRgbColor()	250 0 0 5 10 15	Color parameter outside of expected range: Green Blue	java.awt.Color[r=255,g=10,b=15]	fail
17	This method takes six integers, the first three represent the rgb values of a color	ColorConverter .offsetRgbColor()	100 150 255 -50 -100 -200	Color parameter outside of expected range: Blue	java.awt.Color[r=50,g=50,b=55]	fail
18	This method takes six integers, the first three represent the rgb values of a color	ColorConverter .offsetRgbColor()	2 5 16 45 101 99	Color parameter outside of expected range: Red Green Blue	java.awt.Color[r=47,g=106,b=115]	fail
19	This method takes six integers, the first three represent the rgb values of a color	ColorConverter .offsetRgbColor()	101 101 202 25 50 -101	Color parameter outside of expected range: Blue	java.awt.Color[r=126,g=151,b=101]	fail
20	This method takes six integers, the first three represent the rgb values of a color	ColorConverter .offsetRgbColor()	50 50 50 50 50 50	java.awt.Color[r=0,g=0,b=0]	java.awt.Color[r=100,g=100,b=100]	fail
21	This method calculates the distance between two colors using 3-dimensions Euclidean distance.	DistanceCalculator .calculate()	255 255 255 0 0 0	367.77	441.67	fail
22	This method calculates the distance between two colors using 3-dimensions Euclidean distance.	DistanceCalculator .calculate()	200 100 10 200 100 10	0.0	0.0	pass
23	This method calculates the distance between two colors using 3-dimensions Euclidean distance.	DistanceCalculator .calculate()	0x000000 0xFFFFFF	367.77	441.67	fail
24	This method calculates the distance between two colors using 3-dimensions Euclidean distance.	DistanceCalculator .calculate()	0xAABBCC 0xAABBCC	0.0	0.0	pass
25	This method calculates the distance between two colors using 3-dimensions Euclidean distance.	DistanceCalculator .calculate()	20 80 140 10 80 170	29.62	31.62	fail

Experiences and Lessons Learned

The faults were simple enough to create and the results were as expected. We did not have any roadblocks working with this deliverable. Several of the faults we used were changes in the formulas that were used, which is interesting because that is the same type of fault that caused the code to fail in our initial tests. We learned that it is quite simple to mess up code and it is understandable that there are mistakes in the code, but that is why testing is so important.

CHAPTER 6

Overall Experience and Lessons Learned

This project seemed daunting at first: 25 test cases, scripting, drivers, test automation!? Well, we found out early on that 25 test cases aren't actually a lot. Five test cases per method, hardly gives any information and so many errors could still slip through. Tanaguru had some test cases of their own, however, they focused on very few methods. We did learn how to run their tests using Maven.

In designing our own test cases, we expected all tests to pass. To our surprise, when we first ran our tests on the method calculate, they failed. We initially thought our calculations were incorrect, but after checking our math, we took a closer look at their code and discovered that an incorrect formula was used. We reported this error to their GitHub page, which unfortunately isn't being maintained any longer so the error will likely not be fixed. We don't hold that error against them though, because inserting our own faults into their code was easy to do. We're all people so it is understandable how anyone could make a simple error such as that. The simplicity of these errors also show just how important testing is.

We did face several challenges understanding aspects of the project. Maven splits Contrast Finder into different modules, and since none of us had created such a project before, we had to reverse engineer their package structure before we could write our own drivers. After that, we struggled to define our test cases in a way we could read from the file. None of us have scripted in Bash before, so learning its capabilities and limits was a big hurdle. Testing code is an incredibly involved process, and we learned a lot about how it works through lots of research, trial and error.

Self-Evaluation

When working on our project, we had to study and learn several new or unfamiliar technologies, including Linux, Maven, Bash and HTML. Luckily we always had someone with prior knowledge of these technologies as we were learning, which meant we could share our knowledge with each other as our project advanced. There were some occasions when our communication as a team was lacking, which led to disagreements on how to proceed with our project. This often led to misunderstandings, but we were always able to talk through them.

Overall, each team member pulled their weight. We split the project into tasks, so each person advanced their own skill sets in slightly different ways that the other two team members. In the end though, all of us gained knowledge that we didn't have before.

We are proud of our work and proud to be members of Team CargoPants!

Project Evaluation

We enjoyed the aspect of working with an HFOSS, however, it would be nice to have more freedom of choosing an HFOSS that may not be on the list provided. The list provided was not exhaustive and last updated in 2016.

Being thrown to the wolves was an effective way for us to learn, but having more defined requirements may be useful in the future.