Detailed description of unit test strategy Group 13

Since this is a black box testing method, here are the steps we took for each method/function:
1. Understanding requirements. We firstly looked at the Javadoc requirements for the method we wanted to test.
2. Equivalent Testing. We would identify valid and invalid inputs and expected outputs in order to check the SUT.
3. Boundary Value testing. We would identify negative, 0, and positive inputs for most of our boundary Value testing checks for negative, positive and “0” boundary values.
4. Develop Test Cases. We would then develop test cases for the method with equivalent and boundary value testing and cover as much as we could for the method.
5. Compare with outcomes. We would look at the expected outcome and compare it with the program's outcome.
6. Bug documentation. We would comment on the testing code file and describe the reasoning of the bug.

In depth:
Firstly in step 1, we have to understand the requirements of each other method and how our group understood each method was going through the javadocs that was provided and reading the requirements of inputs, and outputs of each function.

Secondly in step 2, we took a look back at how all our functions would behave given some inputs and we would analyze and categorize valid and invalid inputs depending on the parameters of each method. Making sure we have an equivalent partitioned test will ensure that we have good coverage of the function.

Thirdly in step 3, we have boundary value testing. For this lab, there were no constraints for the values, it only specified the having a lower and upper bound as parameters for the method, so our group decided to make our boundary testing look at the negative numbers, “0”, and positive numbers. A combination of all of these boundaries are tested in our program.

Fourthly in step 4, we develop some JUnit tests based on our equivalent and boundary value testing in order to cover the function in an efficient way.

In step 5, we compared the oracle and the actual program output together to determine if the function was working or if the function had defects. We would look at the behavior of the outputs and compare it to what the method should actually do.

Finally step 6, we would document our bugs in Jira as in progress so that in theory a developer would pick up the bug and fix it. Since we did black box testing we just documented the bug we found in the method in Jira.