Armin Rezaiean-Asel – MP1 TASK 9

Test files being tested via JUnit essentially check each method in a class to see if they are working properly. In order to explain this better, I will use the GaussianSolverTest.java file as an example. Within Eclipse, I chose to **Run As → JUnit** this java file. After doing so, JUnit tested each method in the class, checked for any possible errors that may exist, and printed the amount of time it took to complete the full check as well. Another useful quality of JUnit is that individual methods within a class can be checked. This is a particularly useful characteristic, especially when wanting to check and debug individual methods independently of the entire class.

After looking at the GaussianSolverTest code, I started to understand more about JUnit. I realized that JUnit is useful to check for errors in our code, but it is sometimes even more useful when being used to check individual methods within a class to determine which are the fastest running (and therefore most efficient code process). This knowledge is acquired by observing the time it takes for a JUnit check to run and return a green, no errors bar. By observing individual methods, we can deduce which are the most efficient, which methods contain the least/most errors, and obviously which methods are running incorrectly/correctly.

JUnit can be used to test a method by creating a new test suite by which we can test. For example, if I were to use my part of MP0 (MP0b), I could click on my MP0b class file, do **New→JUnit Test Case** and created my test file accordingly. I've added a screenshot below of a test file for this process returned a green bar meaning there were no errors. If I were to change my int expectedvalue to anything but 3, I would receive a red bar because there would now be an error in my code. I ran testTruth to check to see if the expected and resultant values matched. If so, there were no errors and the JUnit check confirmed this. If not, then there was a confirmed error and I had to look into the issue.

```
public class MP0bTest extends MP0b{

    @Test
    public static void testMP0b() {
        //test to see if MP0b works properly

        int testarray[] = {0,1,2,3,4,5};
        int testvalue = 3;
        int expectedvalue = 3;
        int result = recursiveLocater(testvalue, testarray);

        testTruth(expectedvalue == result);
    }
}
```

As I learned in this task, JUnit is extremely useful in checking for errors in a class as well as in individual methods of a class. It is a powerful tool to implement into programming projects to ensure there are no errors/bugs.