**Jacob Alspaw**

**Hierarchy Testing Report**

**JUnit Tests**

“*First, middle, last*” and “*0, 1, many*” are ways to test a program. “*First*” refers to the first part of the tested item. The method should be tested against the first part of the item. “*Middle*” refers to the middle part of the tested item. The method should be tested against the middle part of this item. “*Last*” refers to the last part of the tested item. The method should be tested against the last part of the tested item. “*0*” is to use test cases of values of 0. “*1*” is to use values of 1 and non-zero values around one. “*Many*” is to use values larger than non-zero values around one. If the method is subjected to these two forms of testing, then most test cases have been considered and the method can be thought of as well-checked.

The JUnit tests I have used cover all of these cases. I first tested each class. Each class was tested for the above conditions. I set the values of many numbers and variables. These variables were either assigned a value or assigned an expression that would result in a value. All thirteen classes were successful. For the arithmeticOperations class, we needed to test 5 basic arithmetic operations. I tested these operations in with a variety of values. For the assignment class, we tested assigning variables a name. We produced variables with a variety of names. For the Boolean operations class, a set of tests needed to be conducted for the Booleans. We gathered a set of true or false comparisons, and compared them. For the comparison class, we compared the value of int numbers together. For the compound statement class we tested statements that were paired together. For the condition class, we tested conditions. These conditions are true or false so we tried scenarios that would return true or false. We tested Function and function call to see if the program could handle a variety of variables and do some sort of methodizing with them. For loop, we tried a variety of loops and compared how they ran, and the number of times the ran through the loop body. For number, made a variety of numbers that would be used throughout the code. For return we made conditions of returns and compared them to their expected otucomes.

**\*\*The JUnit testing resulted in thirteen correct classes\*\***

To extra-verify the classes worked, I just played around with my completed code.