SWE437 Assignment 4

Jordan Brown, Allen Fleming, Joseph Seiler

A. **input domain**

roundingTest() : 12.256789, 4

testFahrenAndCelcius() : 45.32

testCentiAndInches() : 200

testFeetAndMeters() : 3

testMilesAndKilometers() : 20

testGallonsAndLiters() : 300

testGramsAndOz() : 52.854

testPoundsToKilograms() : 5.23

testMphAndKph() : 95

getRoundTest() : 1, 2, 3, 4

C. **Observability**

roundingTest() : the arguments in assertEquals(expected, actual) will tell me if the floating point number rounds to decimal places 0-4 as expected.

testXandY() : each of these methods does the same thing to test the values, they just use different numbers for the conversations based on what units they are. Each method has 2 floating point number variables. One gets converted from X to Y, then the converted variable gets converted back its original X. Then, assertEquals compares the unconverted variable to the converted back floating point number. If successful, assertEquals will tell me.

getRoundTest() : the arguments in assertEquals(expected, actual) will tell me if the method returns the correct number for the decimal place the user wants to round to. It handles the cases in which the user inputs an incorrect number (< 0 and > 4).

D. **Controllability**

roundingTest() : for this test I only need to pass in the floating point number I want to round for the input.

testXandY() : for these tests I only need to pass in the floating point number I want to covert from unit X to unit Y. But in the test I have two inputs for assertEquals(), which are num1 and num2. num1 is the float number for unit X. num2 is the converted float number to unit Y then converted back to unit X. This is done to compare num1 and num2 to see if the conversion was correct.

getRoundTest() : for this test I needed to input 2 values, the float number to round and integer numbers 0-4 that specifies how many decimals places to round the float number to.