**Assignment 9 Write Up**

**Additional test cases added to improve code coverage based on code coverage report(s)**

**Additional Test Cases for Shipping Cost**

T001

Input: 30.0f, ShippingMethod.***Standard***, "Illinios" Output: $42.40

T002

Input: 30.0f, ShippingMethod.***NextDay***, "Illinios" Output: $58.29

T003

Input: -30.0f, ShippingMethod.***NextDay***, "WA" Output: Error

T004

Input: 50.01f, ShippingMethod.***Standard***, "WA" Output: $50.01

T005

Input: 49.99f, ShippingMethod.***Standard***, "WA" Output: $59.99

T006

Input: 50.01f, ShippingMethod.***Standard***, "Illinios" Output: $53.01

T007

Input: 50.01f, ShippingMethod.***NextDay***, "Illinios" Output: $79.51

T006

Input: 49.99f, ShippingMethod.***Standard***, "Illinios" Output: $53.01

T007

Input: 49.99f, ShippingMethod.***NextDay***, "Illinios" Output: $79.48

**Additional Test Cases for Pluralization**

T001

Input: “,./<>?;\':” Expected Output: Error Actual: Error

T002

Input: valley Expected Output: valleys Actual: valleies

T003

Input: delay Expected Output: delays Actual: delaies

T004

Input: soliloquy Expected Output: soliloquies Actual: soliloquies

T005

Input: boy Expected Output: boys Actual: boies

T006

Input: pichiy Expected Output: pichys Actual: pichiies

**Defects with the Calculate Shipping Cost Program**

-Does not account for max value of a floating point number for the total

Input: Float.MAX\_VALUE, ShippingMethod.NextDay, "WA" Expected Output: Error

-Does not account for checking if the state input is an actual US State

T001

Input: 30.0f, ShippingMethod.NextDay, "Guam" Expected Output: Error

T002

Input: 30.0f, ShippingMethod.NextDay, "thisshouldnotwork" Expected Output: Error

-Does not account for checking if the state input is empty

T001

Input: 30.0f, ShippingMethod.NextDay, "" Expected Output: Error

T002

Input: 30.0f, ShippingMethod.NextDay, " " Expected Output: Error

-Does not account for checking if the state input is null

Input: 30.0f, ShippingMethod.NextDay, null Expected Output: Error

-Does not account for checking if total is not a number (NaN)

Input: 30.0f/0.0f, ShippingMethod.NextDay, "WA" Expected Output: Error

-Does not account for checking if total is positive infinity

Input: Float.POSITIVE\_INFINITY, ShippingMethod.NextDay, "WA" Expected Output: Error

**-**Does not account for checking for overflow of the total

Input: Float.MAX\_VALUE+10.0f, ShippingMethod.Standard, "WA" Expected Output: Error

**-**The state of Illinois is spelled wrong so it does not account the 6% sales tax

T001

Input: 30.0f, ShippingMethod.Standard, "Illinois" Expected: $42.40 Actual: $40.00

T002

Input: 30.0f, ShippingMethod.NextDay, "Illinois" Expected: $58.29 Actual: $55.00

**Defects with the Pluralization Program**

-Does not account for null string

Input: null Expected Output: Error

-Does not account for empty string

Input: “” Expected Output: Error

Irregular word list is not big enough

T001

Input: Moose Expected Output: Moose Actual Output: Mooses

T002

Input: Nucleus Expected: Nuclei Actual Output: Nucleuses

T003

Input: Phenomenon Expected Output: Phenomena Actual Output: Phenomenons

-Words ending in ay, ey, iy, oy, and uy has flaws in the way the logic is done creating some unintended results

T001

Input: valley Expected Output: valleys Actual Output: valleies

T002

Input: delay Expected Output: delays Actual Output: delaies

T003

Input: buy Expected Output: buys Actual Output: buies

T004

Input: boy Expected Output: boys Actual Output: boies

T005

Input: pichiy Expected Output: pichys Actual Output: pichiies

-Entering a single letter such “I” crashes the program

Input: I Expected Output: We Actual Output: array index out of range exception

-Program thinks box means ox so it pluralizes box as boxen instead boxes

Input: box Expected Output: boxes Actual Output: boxen

-Does not account for invalid input

Input: thisshouldnotwork Expected Output: Error Actual Output: Unexpected error

**Final Coverage Report**

In the Calculate Orders tests, I was able to get 100% branch coverage, but was only to get 93% line coverage. Also, there was an Orders$ShippingMethod binary file with only 50% line coverage, but it was inaccessible by Cobertura to see what was missed. Not really sure if you were supposed to test that binary file also. With the 93% line coverage, the only line that was not covered was the class and package declaration which doesn’t really make sense to me. Everything else though was covered especially the conditionals. While it was 100% branch coverage, it does not mean that the code itself is defect/bug free. As you can see in my defects detected report there were quite a few defects for this particular method. This is my coverage report for the Calculate Shipping cost program.

In the Pluralize Words program, I was only able to get 92% line coverage and 71% branch coverage. With 71% branch coverage, it mostly dealt with not covering the conditions where words end in ay, ey, iy, oy, and uy. The way the logic is set for it has some flaws in it where test cases thrown at it say some parts are not executed and then you get unintended results. Also the report said the main method wasn’t tested in the line coverage or the branch coverage, but I don’t think it should be tested since that is where the program is actually being executed to show that it works with its very specific cases only. When you try to stress it out more, we get a lot of undefined behavior like what I described in the defects with pluralization section.

In all, these are the reasons as to why I was not able to get 100% branch/line coverage in both programs. This is mostly due to the defects in the code when trying to create test cases to satisfy them and also with the lines that are not being covered that don’t really make sense in trying to test them such as class and package declarations. Even you are theoretically able to get 100% on both them for branch/line coverage, it does not mean that the programs are error free. It is a dangerous assumption to have when you are developing and testing software. These are the thoughts that I have gathered when trying to compile a report on my testing for both of these programs.