Project Report: Windows Calculator

Unit Testing with Sufficient Coverage

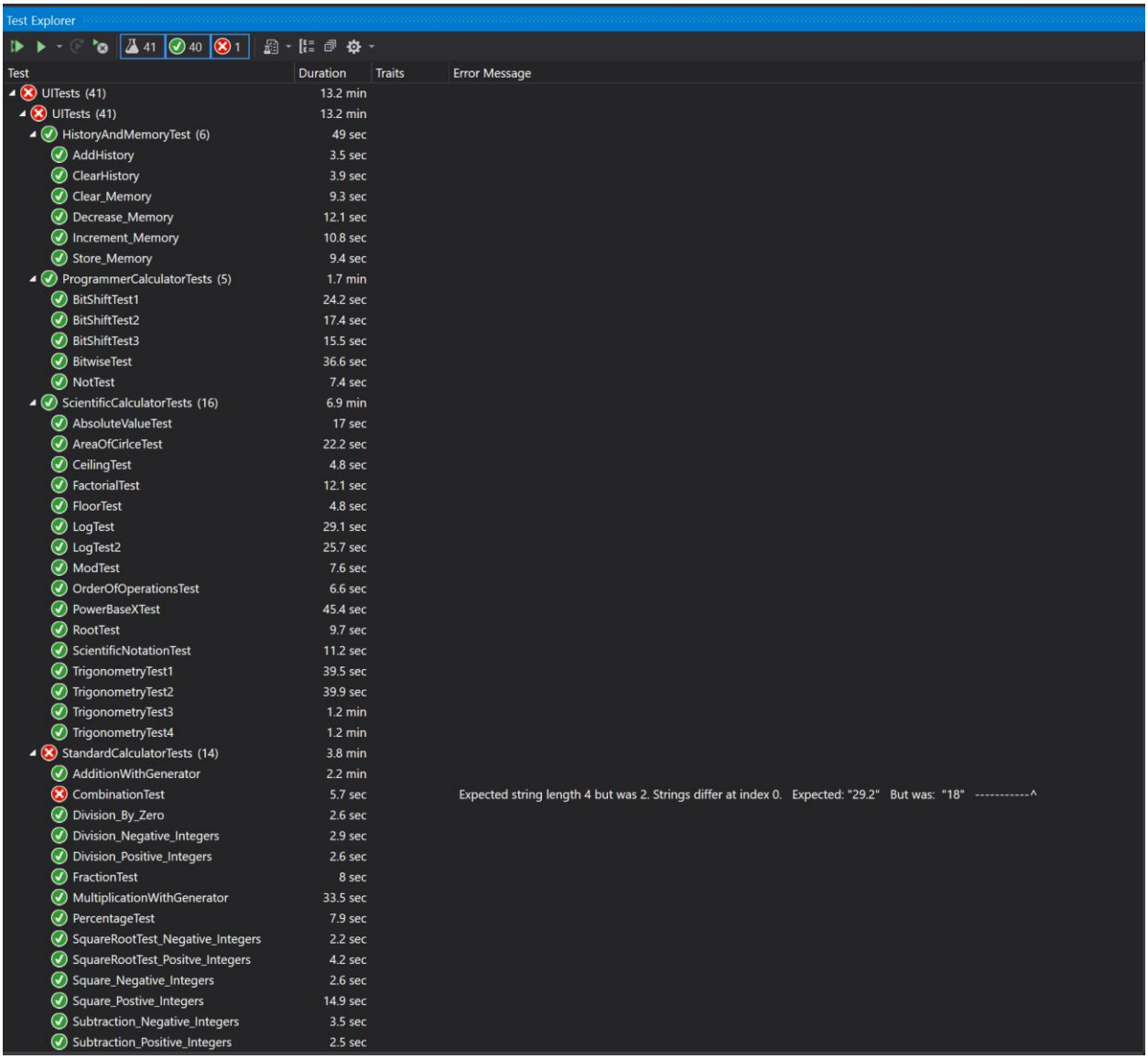
Since I was going to be using WinAppDriver (selenium-like) tool to interactively test the Windows Calculator through the UI and minimizes the need to modify implementation code. During my initial setup and testing I realized that the tests were unable to measure coverage of the SUT and after trying to use multiple tools to instrument that code, I was unable to get code coverage working with WinAppDriver. My emphasis was to create tests for as many features for each Calculator mode, to ensure there every feature and operation was covered.

Automated Testing

In addition, with using the WinAppDriver, I created generators for creating expressions and various input for different operations.

- Standard Calculator
  - Standard Calculator Tests - https://github.com/csun-comp587-s20/calculator/blob/master/src/UITests/StandardCalculatorTests.cs
  - Created tests that include coverage for - Addition, Subtraction, Multiplication, Division, Percentage, Fraction, Square Root, Square, Including use of Generator.
  - Generator - https://github.com/csun-comp587-s20/calculator/blob/master/src/UITests/TestFramework/SimpleArithmeticGenerator.cs
- Scientific Calculator
  - Scientific Calculator Tests - https://github.com/csun-comp587-s20/calculator/blob/master/src/UITests/ScientificCalculatorTests.cs
  - Created tests that include coverage for - Logs, Abs Value, Mod, Geometry, Trigonometry, Ceiling, Floor
- Programmer Calculator
  - Programmer Calculator Tests - https://github.com/csun-comp587-s20/calculator/blob/master/src/UITests/ProgrammerCalculatorTests.cs
  - Created tests that include coverage for – Bitwise Operation, Bit Shift Operations
- History & Memory
  - History Tests - https://github.com/csun-comp587-s20/calculator/blob/master/src/UITests/HistoryAndMemoryTest.cs
  - Created tests include coverage for – Adding/Updating/Recalling/Clearing Values from Memory, Checking History is adding properly.
- Supporting Framework
  - Created a support framework to simplify testing using Page Object Model
  - https://github.com/csun-comp587-s20/calculator/tree/master/src/UITests/TestFramework
  -

Additional detail about how each feature in the three modes can be found here: https://github.com/csun-comp587-s20/calculator/blob/master/final_calculator_feature_tracking.xlsx



The single failure is due to the Standard Calculator mode, not properly handling order of operations.

Lessons Learned

I would have pick a SUT that was written in a language I am more familiar with and am able to create Unit Tests + UI Tests in order to have code coverage. I would research my tools beforehand, and they are actively update. Although, I am familiar with Selenium, because I have used Appium for Mobile App UI Testing, WinAppDriver was not that different but does not have the same level of support and maturity.