**HW 02a - Testing a legacy program and reporting on testing results**

SSW-567 – SW Testing, Qual. Assur. & Maint

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Assignment Description**:

Sometimes you will be given a program that someone else has written, and you will be asked to fix, update and enhance that program.   In this assignment you will start with an existing implementation of the classify triangle program that will be given to you.   You will also be given a starter test program that tests the classify triangle program, but those tests are not complete.

* These are the two files:  Triangle.py and TestTriangle.py
  + [***Triangle.py***](https://sit.instructure.com/courses/40463/files/6396462/download?wrap=1)is a starter implementation of the triangle classification program.
  + [***TestTriangle.py***](https://sit.instructure.com/courses/40463/files/6396461/download?wrap=1)**c**ontains a starter set of unittest test cases to test the classifyTriangle() function in the file Triangle.py file.

In order to determine if the program is correctly implemented, you will need to update the set of test cases in the test program.  You will need to update the test program until you feel that your tests adequately test all of the conditions.   Then you should run the complete set of tests against the original triangle program to see how correct the triangle program is.    Capture and then report on those results in a formal test report described below.   For this first part you should not make any changes to the classify triangle program.  You should only change the test program.

Based on the results of your initial tests, you will then update the classify triangle program to fix all defects.  Continue to run the test cases as you fix defects until all of the defects have been fixed.   Run one final execution of the test program and capture and then report on those results in a formal test report described below.

**Author**: Srikanth Uppada

**Summary**:

Written total 21 test cases to verify the initial *“classifyTriangle”* program. Using test framework was able to identify 11 failures. This helped to pinpoint the root cause of the source code error in *“classifyTriangle”* program. There were multiple issues in the program. This approach is straight forward to identify the issues right away in the initial stages.

Below is the initial summary of test cases results:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Inputs**  **(Side a, Side b, Side c)** | **Expected Results** | **Actual Results** | **Pass or Fail** |
| 1 | 3,4,5 | Right | InvalidInput | Fail |
| 2 | 5,3,4 | Right | InvalidInput | Fail |
| 3 | 1,1,1 | Equilateral | InvalidInput | Fail |
| 4 | 2,2,1 | Isoceles | InvalidInput | Fail |
| 5 | 5, 7, 7 | Isoceles | InvalidInput | Fail |
| 6 | 9,3,9 | Isoceles | InvalidInput | Fail |
| 7 | 15,17,8 | Right | InvalidInput | Fail |
| 8 | 2,5,6 | Scalene | InvalidInput | Fail |
| 9 | 201,15,24 | InvalidInput | InvalidInput | Pass |
| 10 | 50,201,40 | InvalidInput | InvalidInput | Pass |
| 11 | 70,120,201 | InvalidInput | InvalidInput | Pass |
| 12 | -1,5,6 | InvalidInput | InvalidInput | Pass |
| 13 | 8,-6,3 | InvalidInput | InvalidInput | Pass |
| 14 | 16,13,-4 | InvalidInput | InvalidInput | Pass |
| 15 | 0.5,1.0,2.35 | InvalidInput | InvalidInput | Pass |
| 16 | 0,199,199 | InvalidInput | InvalidInput | Pass |
| 17 | 199,0,199 | InvalidInput | InvalidInput | Pass |
| 18 | 199,199,0 | InvalidInput | InvalidInput | Pass |
| 19 | 10,12,24 | NotATriangle | InvalidInput | Fail |
| 20 | 6,2,3 | NotATriangle | InvalidInput | Fail |
| 21 | 1,3,2 | NotATriangle | InvalidInput | Fail |

**Honor pledge:**

*“I pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination. I further pledge that I have not copied any material from a book, article, the Internet or any other source except where I have expressly cited the source.”*

**Detailed results:**

Provided different inputs for each side (a,b,c) of the triangle to verify if any particular side is triggering an invalid scenario. This technique is used to test the robustness of each input.

Below is the test execution summary of triangle program:

|  |  |
| --- | --- |
| **Test details** | **Test Run 1** |
| Tests Planned | 21 |
| Tests Executed | 21 |
| Tests Passed | 10 |
| Defects Found | 11 |
| Defects Fixed | 0 |

**Note: 1.** **Test run1: initial***classifyTriangle().*

For output result file, refer file: Report\_TestTriangle\_Initial.html for **Test run1**.

Github details: <https://github.com/sriksrik7/SSW-567>

Folder: HW02a