**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.

**Author:** Kevin Ferreras

**Summary:**

*Part 1 Test Report – Test Set against buggy classifyTriangle():*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Input | Expected Results | Actual Result | Pass or Fail |
| 1.0.1 | a = 3; b = 4; c = 5 | "Right: 3,4,5 is a Right triangle" | "InvalidInput" | Fail |
| 1.0.2 | a = 5; b = 3; c = 4 | "Right: 5,3,4 is a Right triangle" | "InvalidInput" | Fail |
| 1.0.3 | a = 1; b = 1; c = 1 | "Equilateral: 1,1,1 is an Equilateral Triangle" | "InvalidInput" | Fail |
| 1.0.4 | a = 7; b = 7; c = 4 | "Isosceles: 7,7,4 is an Isosceles Triangle" | "InvalidInput" | Fail |
| 1.0.5 | a = 6; b = 5; c = 7 | "Scalene: 6,5,7 is a Scalene Triangle" | "InvalidInput" | Fail |
| 1.0.6 | a = 300; b = 40; c = 50 | "Invalid Input" | "InvalidInput" | Fail |
| a = 70; b = 700; c = 40 | "Invalid Input" | "InvalidInput" | Fail |
| a = 50; b = 30; c = 400 | "Invalid Input" | "InvalidInput" | Fail |
| a = 600; b = 600; c = 600 | "Invalid Input" | "InvalidInput" | Fail |
| a = 201; b = 40; c = 40 | "Invalid Input" | "InvalidInput" | Fail |
| a = 50; b = 201; c = 40 | "Invalid Input" | "InvalidInput" | Fail |
| a = 20; b = 40; c = 201 | "Invalid Input" | "InvalidInput" | Fail |
| 1.0.7 | a = -3; b = 4; c = 5 | "Invalid Input" | "InvalidInput" | Fail |
| a = 7; b = -7; c = 4 | "Invalid Input" | "InvalidInput" | Fail |
| a = 5; b = 3; c = -4 | "Invalid Input" | "InvalidInput" | Fail |
| a = -1; b = -1; c = -1 | "Invalid Input" | "InvalidInput" | Fail |
| 1.0.8 | a = 3.2; b = 4; c = 5 | "Invalid Input" | "InvalidInput" | Fail |
| a = 7; b = 7.1; c = 4 | "Invalid Input" | "InvalidInput" | Fail |
| a = 5; b = 3; c = 4.0 | "Invalid Input" | "InvalidInput" | Fail |
| a = 1.5; b = 1.5; c = 1.5 | "Invalid Input" | "InvalidInput" | Fail |
| 1.0.9 | a = 3; b = 4; c = 9 | "Not a Triangle" | "InvalidInput" | Fail |
| a = 2; b = 4; c = 7 | "Not a Triangle" | "InvalidInput" | Fail |

*Part 2 Test Report - Test Set against improved classifyTriangle():*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Input | Expected Results | Actual Result | Pass or Fail |
| 1.1.1 | a = 3; b = 4; c = 5 | "Right: 3,4,5 is a Right triangle" | "Right: 3,4,5 is a Right triangle" | Pass |
| 1.1.2 | a = 4; b = 3; c = 5 | "Right: 4,3,5 is a Right triangle" | "Right: 4,3,5 is a Right triangle" | Pass |
| 1.1.3 | a = 1; b = 1; c = 1 | "Equilateral: 1,1,1 is an Equilateral Triangle" | "Equilateral: 1,1,1 is an Equilateral Triangle" | Pass |
| 1.1.4 | a = 7; b = 7; c = 4 | "Isosceles: 7,7,4 is an Isosceles Triangle" | "Isosceles: 7,7,4 is an Isosceles Triangle" | Pass |
| 1.1.5 | a = 6; b = 5; c = 7 | "Scalene: 6,5,7 is a Scalene Triangle" | "Scalene: 6,5,7 is a Scalene Triangle" | Pass |
| 1.1.6 | a = 300; b = 40; c = 50 | "Invalid Input" | "Invalid Input" | Pass |
| a = 70; b = 700; c = 40 | "Invalid Input" | "Invalid Input" | Pass |
| a = 50; b = 30; c = 400 | "Invalid Input" | "Invalid Input" | Pass |
| a = 600; b = 600; c = 600 | "Invalid Input" | "Invalid Input" | Pass |
| a = 201; b = 40; c = 40 | "Invalid Input" | "Invalid Input" | Pass |
| a = 50; b = 201; c = 40 | "Invalid Input" | "Invalid Input" | Pass |
| a = 20; b = 40; c = 201 | "Invalid Input" | "Invalid Input" | Pass |
| 1.1.7 | a = -3; b = 4; c = 5 | "Invalid Input" | "Invalid Input" | Pass |
| a = 7; b = -7; c = 4 | "Invalid Input" | "Invalid Input" | Pass |
| a = 5; b = 3; c = -4 | "Invalid Input" | "Invalid Input" | Pass |
| a = -1; b = -1; c = -1 | "Invalid Input" | "Invalid Input" | Pass |
| 1.1.8 | a = 3.2; b = 4; c = 5 | "Invalid Input" | "Invalid Input" | Pass |
| a = 7; b = 7.1; c = 4 | "Invalid Input" | "Invalid Input" | Pass |
| a = 5; b = 3; c = 4.0 | "Invalid Input" | "Invalid Input" | Pass |
| a = 1.5; b = 1.5; c = 1.5 | "Invalid Input" | "Invalid Input" | Pass |
| 1.1.9 | a = 3; b = 4; c = 9 | "Not a Triangle" | "Not a Triangle" | Pass |
| a = 2; b = 4; c = 7 | "Not a Triangle" | "Not a Triangle" | Pass |

*Test Run Matrix:*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Test Run 1 | Test Run 2 | Test Run 3 | Test Run 4 | Test Run 5 | Test Run 6 |
| Tests Planned | 9 | 9 | 9 | 9 | 9 | 9 |
| Tests Executed | 9 | 9 | 9 | 9 | 9 | 9 |
| Tests Passed | 0 | 3 | 3 | 4 | 8 | 9 |
| Defects Found | 9 | 6 | 6 | 5 | 1 | 0 |
| Defects Fixed | 0 | 3 | 3 | 4 | 8 | 9 |

*Reflection:*

After working through this assignment, I have a better understanding and appreciation for what it means to read, correct, and test legacy code. Initially, it seemed to me that I would be able to catch all the bugs in one code review and one test run. However, that was not the case, but one of the things that I found to have worked well was having the enhanced test suites that I was able to frequently run. I found that the test suites worked well because they caught bugs in the code that I had missed during my initial code review, and lead me to review the code again and think through why a failure was being generated.

*Test Strategy:*

When writing my test cases, the strategy that I used to decide whether I had sufficient number of test cases was to make sure that I had at least one input case for each scenario that could render a triangle invalid or break one of the specific program requirements. For example, in the case of the different triangle classifications (Equilateral, Right, Isosceles, Scalene), I used one set of valid inputs for each classification to test that the program could recognize valid triangles. Similarly, for the “Sum of two Sides” property of a triangle, I used a set of valid inputs in order to test that the program could identify this case. For the cases of: negative values, decimal values, and values greater than 200, I used multiple sets of inputs to test that the program could identify these values whether the program received the values for “a”, “b”, “c”, or all.

**Honor Pledge:**

I pledge my honor I have abided by the Stevens honor system.

**Detailed Results:**

* No assumptions or constraints were made for this assignment
* For data input description, please see “*Test Strategy”* section above
* For results of my work, please see “*Summary”* section above