**Author:** Atishay Jain  
  
**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.

* [***Triangle.py***](https://sit.instructure.com/courses/60673/files/10121006/download?wrap=1)is a starter implementation of the triangle classification program.
* [***TestTriangle.py***](https://sit.instructure.com/courses/60673/files/10121003/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.

Note that you should NOT simply replace the logic with your logic from Assignment 1.  Test teams typically don't have the luxury of rewriting code from scratch and instead must fix what's delivered to the test team.

[*Triangle.py*](https://sit.instructure.com/courses/60673/files/10121006/download?wrap=1)*contains an implementation of the classifyTriangle() function with a few bugs.*  
[*TestTriangle.py*](https://sit.instructure.com/courses/60673/files/10121003/download?wrap=1)*contains the initial set of test cases*

***Author:*** *Atishay Jain*

**Honor Pledge:** I hereby agree that all the code changes and new test cases have been written by me.

**Summary:** The program triangle.py had a few bugs including syntax bugs which have been fixed. I have also added a few more test cases and fixed errors in testtriangle.py file also. In the updated file   
after user enters correct values for triangle then it outputs correct answers for the type of classified triangle. In the given testtriangle.py program.

The tests which were run aimed at: if the given sides/input forms a triangle, if the given sides form a right-angle triangle or not, if the given sides input form an equilateral triangle or not, if the given sides input form an isosceles triangle or not, if the given sides form a scalene triangle or not if the sides are even valid or not.

I ran 9 tests on the previous code that was given. From which 7 failed. The test that got passed were testStringTriangle and testTriangleSides.

Testing if triangle is Right triangle.

Testing if triangle is Equilateral Triangle.

Testing if the triangle is valid or not.

Testing if triangle is scalene or not.

Testing if Triangle does form a triangle or not.

Testing if triangle is isosceles triangle or not.

Testing if triangle can accept string value as input or not.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Input | Expected Results | Actual Result | Pass or Fail |
| testRightTriangleA | 3, 4, 5 | Right | InvalidInput | Fail |
| testRightTriangleB | 5, 3, 4 | Right | InvalidInput | Fail |
| testEquilateralTriangles | 1, 1, 1 | Equilateral | InvalidInput | Fail |
| testIfTriangle | 10,4,3 | NotATriangle | InvalidInput | Fail |
| testScaleneTriangle | 2,3,4 | Scalene | InvalidInput | Fail |
| testStringTriangle | a, b, c | Error | Error | Type Error |
| testTriangleSides | 10,-10,-20 | InvalidInput | InvalidInput | Pass |
| testIsocelesTriangleA | 8,8,10 | Isoceles | InvalidInput | Fail |
| testIsocelesTriangleB | 3,3,4 | Isoceles | InvalidInput | Fail |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Input | Expected Results | Actual Result | Pass or Fail |
| testRightTriangleA | 3, 4, 5 | Right | Right | Pass |
| testRightTriangleB | 5, 3, 4 | Right | Right | Pass |
| testEquilateralTriangles | 1, 1, 1 | Equilateral | Equilateral | Pass |
| testIfTriangle | 10,4,3 | NotATriangle | NotATriangle | Pass |
| testScaleneTriangle | 2,3,4 | Scalene | Scalene | Pass |
| testStringTriangle | a, b, c | Error | Error | Type Error |
| testTriangleSides | 10,-10,-20 | InvalidInput | InvalidInput | Pass |
| testIsocelesTriangleA | 8,8,10 | Isoceles | Isoceles | Pass |
| testIsocelesTriangleB | 3,3,4 | Isoceles | Isoceles | Pass |

|  |  |
| --- | --- |
| Test Run 1 | Test Run 2 |
| Tests Planned | testRightTriangleA, testRightTriangleB, testEquilateralTriangles, testIfTriangle, testIsoscelesTriangleA, testIsoscelesTriangleB, testScaleneTriangle, testStringTriangle, testTriangleSides | testRightTriangleA, testRightTriangleB, testEquilateralTriangles, testIfTriangle, testIsoscelesTriangleA, testIsoscelesTriangleB, testScaleneTriangle, testStringTriangle, testTriangleSides |
| Tests Executed | testRightTriangleA, testRightTriangleB, testEquilateralTriangles, testIfTriangle, testIsoscelesTriangleA, testIsoscelesTriangleB, testScaleneTriangle, testStringTriangle, testTriangleSides | testRightTriangleA, testRightTriangleB, testEquilateralTriangles, testIfTriangle, testIsoscelesTriangleA, testIsoscelesTriangleB, testScaleneTriangle, testStringTriangle, testTriangleSides |
| Tests Passed | testTriangleSides | testRightTriangleA, testRightTriangleB, testEquilateralTriangles, testIfTriangle, testIsoscelesTriangleA, testIsoscelesTriangleB, testScaleneTriangle, testStringTriangle, testTriangleSides |
| Defects Found | The logic for checking if the given sides do not form a triangle is wrong.  The logic for checking if the given triangle is an equilateral triangle is wrong.  The logic for checking if the given triangle is a right triangle is wrong.  The logic for checking if the given triangle is a scalene triangle is wrong. | None |
| Defects Fixed | Rewrote the logic for all the logical errors mentioned above. | None |

In this assignment I learned how to read other’s code and make changes to it. I learned how to debug an existing code. I also learned how test cases are ran and how we can add other test cases to existing files and make it run. I learned that string inputs cannot be added as an input value for triangle.