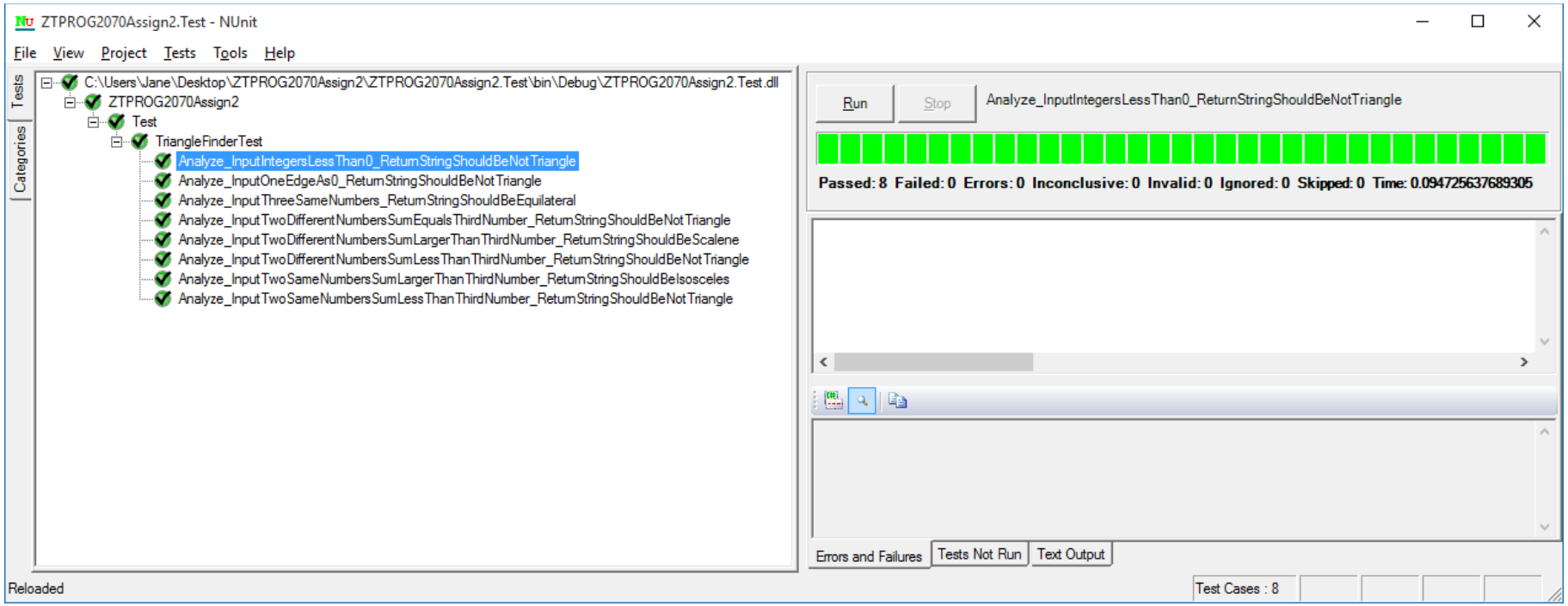

ZTPROG2070Assign2.Test - NUnit
File View Project Tests Tools Help
Tests
Categories
C:\Users\Jane\Desktop\ZTPROG2070Assign2\ZTPROG2070Assign2.Test\bin\Debug\ZTPROG2070Assign2.Test.dll
ZTPROG2070Assign2
Test
TriangleFinderTest
Analyze_InputIntegersLessThan0_ReturnStringShouldBeNotTriangle
Analyze_InputOneEdgeAs0_ReturnStringShouldBeNotTriangle
Analyze_InputThreeSameNumbers_ReturnStringShouldBeEquilateral
Analyze_InputTwoDifferentNumbersSumEqualsThirdNumber_ReturnStringShouldBeNotTriangle
Analyze_InputTwoDifferentNumbersSumLargerThanThirdNumber_ReturnStringShouldBeScalene
Analyze_InputTwoDifferentNumbersSumLessThanThirdNumber_ReturnStringShouldBeNotTriangle
Analyze_InputTwoSameNumbersSumLargerThanThirdNumber_ReturnStringShouldBeIsosceles
Analyze_InputTwoSameNumbersSumLessThanThirdNumber_ReturnStringShouldBeNotTriangle
Run
Stop
Analyze_InputIntegersLessThan0_ReturnStringShouldBeNotTriangle
Passed: 8 Failed: 0 Errors: 0 Inconclusive: 0 Invalid: 0 Ignored: 0 Skipped: 0 Time: 0.094725637689305
Errors and Failures
Tests Not Run
Text Output
Reloaded
Test Cases : 8

Unit Tests for Analyze:

1. public void Analyze_InputIntegersLessThan0_ReturnStringShouldBeNotTriangle()

   Edges are -1, 5, 6 : for edges of a triangle should not be less than 0

2. public void Analyze_InputOneEdgeAs0_ReturnStringShouldBeNotTriangle()

   Edges are 1, 0, 2 : for 0 is a special number and edges should not be 0

3. public void Analyze_InputThreeSameNumbers_ReturnStringShouldBeEquilateral()

   Edges are 3, 3, 3 : three same numbers to test equilateral triangle

4. public void
   Analyze_InputTwoSameNumbersSumLargerThanThirdNumber_ReturnStringShouldBeIsosceles()

   Edges are 6, 6, 8 : two same numbers and its sum is larger than the third number, to test isosceles triangle

5. public void
   Analyze_InputTwoSameNumbersSumLessThanThirdNumber_ReturnStringShouldBeNotTriangle()

   Edges are 7, 7, 14 : two same numbers and its sum just equals the third number, which cannot form a triangle

6. public void
   Analyze_InputTwoDifferentNumbersSumLessThanThirdNumber_ReturnStringShouldBeNotTriangle()

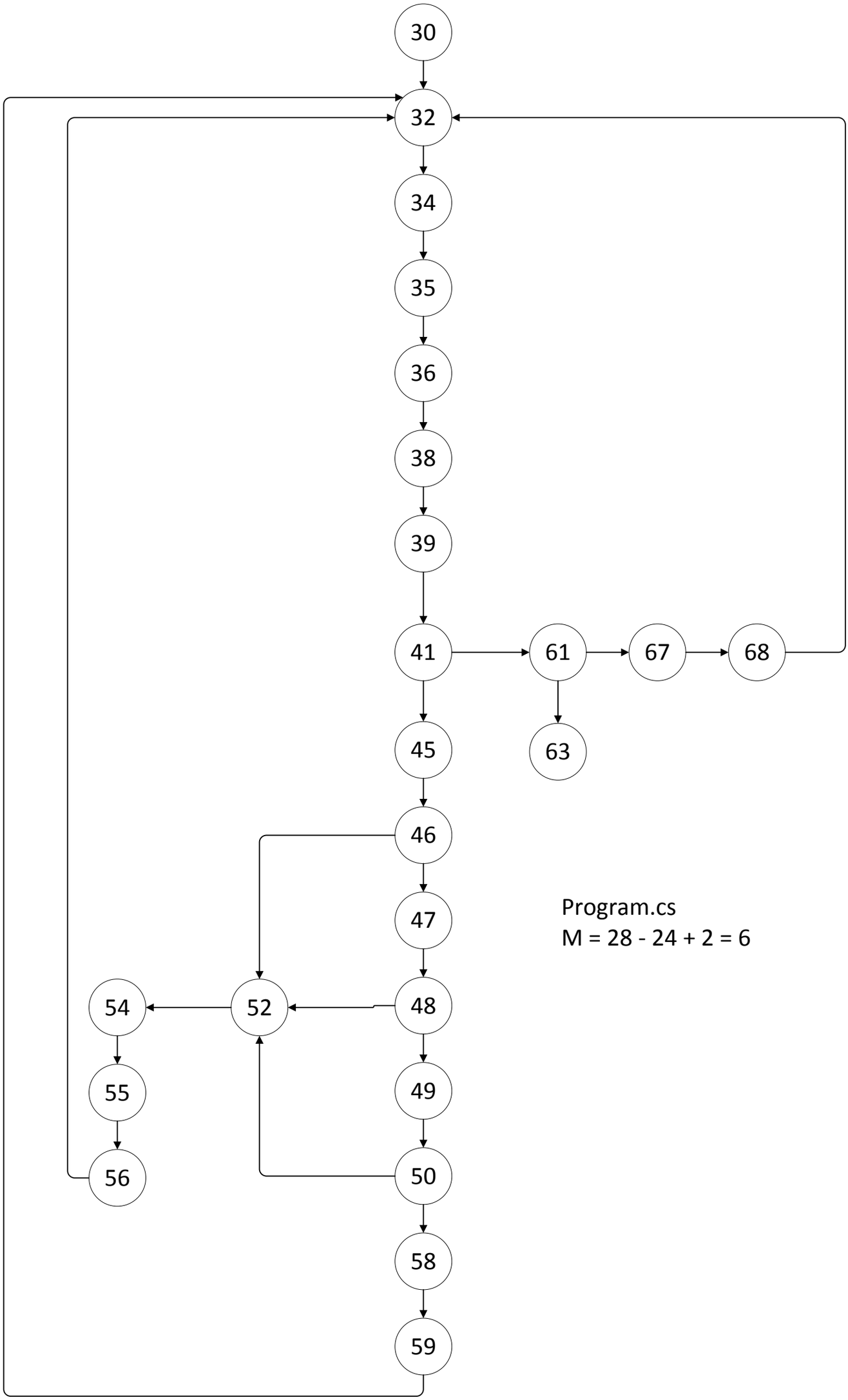
   Edges are 9999999, 999999, 10999999 : relatively large numbers and the third number is larger than the sum of the other two by 1 – the smallest integer, which cannot form a triangle

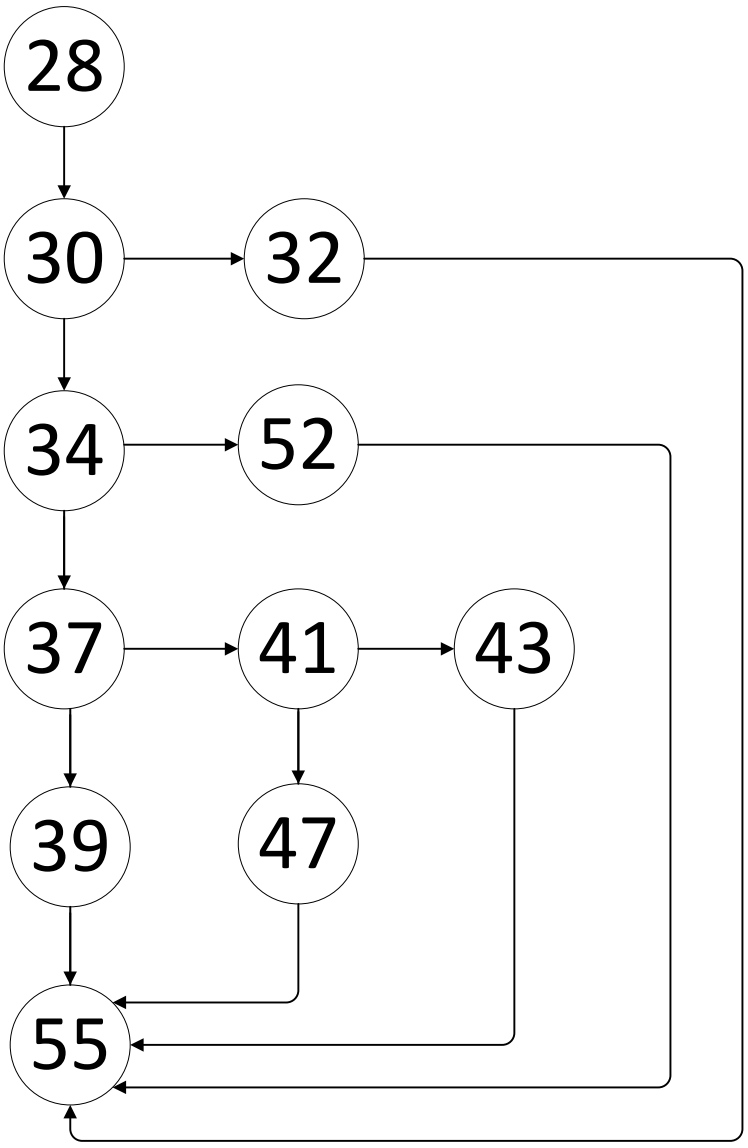
7. public void
   Analyze_InputTwoDifferentNumbersSumEqualsThirdNumber_ReturnStringShouldBeNotTriangle()

   Edges are 99999999, 9999999, 109999998 : relatively large numbers and the third number equals the sum of the other two, which cannot form a triangle

8. public void
   Analyze_InputTwoDifferentNumbersSumLargerThanThirdNumber_ReturnStringShouldBeScalene()

   Edges are 2, 3, 4 : three different integers to test the smallest scalene triangle used integers


30
32
34
35
36
38
39
41
61
67
68
45
63
46
47
48
52
54
55
56
49
50
58
59
Program.cs
M = 28 - 24 + 2 = 6



TriangleFinder.cs
M = 14 - 11 + 2 = 5

Cyclomatic Complexity calculation :

Program.cs

M = 28 – 24 + 2 = 6

There are a while loop and two if statements containing a single condition. According to the CC calculation formula, the CC is 6 and it's less than 10. There is no need to split it into multiple modules. It is able to be tested and maintained without difficulty.

TriangleFinder.cs

M = 14 – 11 + 2 = 5

There are two if statements and within the second if statement, there are another two if statements. Each if statement contains two or three conditions. According to the CC calculation formula, the CC is 5 and it's less than 10. There is no need to split it into multiple modules. It is able to be tested and maintained without difficulty.