**Assignment – 1**

**Unit Testing**

**PROG8170**

**Software Quality Assurance Techniques**

**Professor: Mrs F Sharifi**

**Name:** Suhas Vemuganti

**Student Id:** 8618719

**Program Source Code**

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks; 6

7 namespace Assignment1 8 {

|  |  |  |
| --- | --- | --- |
| 9 | class Program |  |
| 10 | { |
| 11 | static void Main(String[] args) |
| 12 | { |
| 13 | string userInput = ""; |
| 14 | bool givenMenu = false; |
| 15 | string g = string.Empty; |
| 16 | while (givenMenu == false) |
| 17 | { |
| 18 | Console.WriteLine("1 = Enter dimensions of | Triangle"); |
| 19 | Console.WriteLine("2 = Exit \n"); |  |
| 20 | Console.WriteLine("Please select an option | \n"); |
| 21 | userInput = Console.ReadLine(); |  |
| 22 | if (userInput != "1" && userInput != "2") |  |
| 23 | { |  |

24

25 }

Console.WriteLine("That's not a valid menu option, please try again:\n");

26 else if (userInput == "1")

27 {

1. Console.WriteLine("...............Enter three sides of traingle ");
2. Console.WriteLine("Enter First side of triangle: ");
3. int a = int.Parse(Console.ReadLine());
4. Console.WriteLine("Enter Second Side of traingle : ");
5. int b = int.Parse(Console.ReadLine());
6. Console.WriteLine("Enter Third Side of traingle : ");
7. int c = int.Parse(Console.ReadLine());
8. g = TriangleSolver.Analyze(a, b, c);
9. Console.WriteLine("{0}", g);

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 37 |  |  |  |  | } |
| 38 |  |  |  |  | else |
| 39 |  |  |  |  | { |
| 40 |  |  |  |  | givenMenu = true; |
| 41 |  |  |  |  | break; |
| 42 |  |  |  |  | } |
| 43 |  |  |  | } |  |
| 44 |  |  | } |  |  |
| 45 |  | } |  |  |  |
| 46 | } |  |  |  |  |
|  |  |  |  |  |  |

**Triangle Solver Code**

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;

7 namespace Assignment1 8 {

|  |  |  |
| --- | --- | --- |
| 9 |  | public static class TriangleSolver |
| 10 |  | { |
| 11 |  | public static string Analyze(int a, int b, int c) |
| 12 |  | { |
| 13 |  | string result = string.Empty; |
| 14 |  | if (a + b > c && b + c > a && c + a > b) |
| 15 |  | { |
| 16 |  | Console.WriteLine("Given inputs form a triangle."); |
| 17 |  | if (a == b && b == c && c == a) |
| 18 |  | { |
| 19 |  | result = "Triangle is Equilateral."; |
| 20 |  | } |
| 21 |  | else if (a == b || b == c || c == a) |
| 22 |  | { |
| 23 |  | result = "Triangle is Isosceles."; |
| 24 |  | } |
| 25 |  | else |
| 26 |  | { |
| 27 |  | result = "Triangle is Scalene."; |
| 28 |  | } |
| 29 |  | } |
| 30 |  | else |
| 31 |  | { |
| 32 |  | result = "Given inputs doesn’t form a triangle."; |
| 33 |  | } |
| 34 |  | return result; |
| 35 |  | } |
| 36 |  | } |
| 37 | } |  |

**Unit Test Code**

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Text;
5. using System.Threading.Tasks;
6. using Assignment1;
7. using NUnit.Framework; 8

9 namespace Assignment1\_tests 10 {

1. [TestFixture]
2. class Assignment1\_tests 13 {
3. [Test]
4. public static void Analyze\_ainput5\_binput5\_cinput10\_Outputnotatriangle() 16 {
5. //Arrange
6. int a = 5;
7. int b = 5;
8. int c = 10;
9. string result = TriangleSolver.Analyze(a, b, c);
10. string expectedoutput = result; 23
11. //Act
12. string actualresult = "Given inputs doesn’t form a triangle."; 26
13. //Assert
14. Assert.AreEqual(expectedoutput, actualresult);
15. }
16. [Test]
17. public void Analyze\_ainput10\_binput10\_cinput10\_OutputEquilateral() 32 {
18. //Arrange
19. int a = 10;
20. int b = 10;
21. int c = 10;
22. string result = TriangleSolver.Analyze(a, b, c);
23. string expectedoutput = result; 39
24. //Act
25. string actualresult = "Triangle is Equilateral."; 42
26. //Assert
27. Assert.AreEqual(expectedoutput, actualresult); 45 }
28. [Test]
29. public void Analyze\_ainput8\_binput8\_cinput10\_OutputIsosceles() 48 {
30. //Arrange
31. int a = 8;
32. int b = 8;
33. int c = 10;

53

54

55

56

57

58

59

60

61 }

string result = TriangleSolver.Analyze(a, b, c); string expectedoutput = result;

//Act

string actualresult = "Triangle is Isosceles.";

//Assert

Assert.AreEqual(expectedoutput, actualresult);

1. [Test]
2. public void Analyze\_ainput11\_binput16\_cinput20\_OutputScalene() 64 {
3. //Arrange
4. int a = 11;
5. int b = 16;
6. int c = 20;
7. string result = TriangleSolver.Analyze(a, b, c);
8. string expectedoutput = result; 71
9. //Act
10. string actualresult = "Triangle is Scalene."; 74
11. //Assert
12. Assert.AreEqual(expectedoutput, actualresult); 77

78 }

1. [Test]
2. public void Analyze\_ainput11\_binput11\_cinput22\_Outputnotatriangle() 81 {
3. //Arrange
4. int a = 11;
5. int b = 11;
6. int c = 22;
7. string result = TriangleSolver.Analyze(a, b, c);
8. string expectedoutput = result; 88
9. //Act
10. string actualresult = "Given inputs doesn’t form a triangle."; 91
11. //Assert
12. Assert.AreEqual(expectedoutput, actualresult); 94 }
13. [Test]
14. public void Analyze\_ainput20\_binput20\_cinput20\_OutputEquilateral() 97 {
15. //Arrange
16. int a = 20;
17. int b = 20;
18. int c = 20;
19. string result = TriangleSolver.Analyze(a, b, c);
20. string expectedoutput = result; 104

105 //Act

|  |  |  |  |
| --- | --- | --- | --- |
| 106 |  |  | string actualresult = "Triangle is Equilateral."; |
| 107 |  |  |  |
| 108 |  |  | //Assert |
| 109 |  |  | Assert.AreEqual(expectedoutput, actualresult); |
| 110 |  |  | } |
| 111 |  |  | [Test] |
| 112 |  |  | public void Analyze\_ainput60\_binput70\_cinput60\_OutputIsosceles() |
| 113 |  |  | { |
| 114 |  |  | //Arrange |
| 115 |  |  | int a = 60; |
| 116 |  |  | int b = 70; |
| 117 |  |  | int c = 60; |
| 118 |  |  | string result = TriangleSolver.Analyze(a, b, c); |
| 119  120  121 |  |  | string expectedoutput = result;  //Act |
| 122  123 |  |  | string actualresult = "Triangle is Isosceles."; |
| 124 |  |  | //Assert |
| 125 |  |  | Assert.AreEqual(expectedoutput, actualresult); |
| 126 |  |  | } |
| 127 |  |  | [Test] |
| 128 |  |  | public void Analyze\_ainput30\_binput36\_cinput20\_OutputScalene() |
| 129 |  |  | { |
| 130 |  |  | //Arrange |
| 131 |  |  | int a = 30; |
| 132 |  |  | int b = 36; |
| 133 |  |  | int c = 20; |
| 134 |  |  | string result = TriangleSolver.Analyze(a, b, c); |
| 135 |  |  | string expectedoutput = result; |
| 136 |  |  |  |
| 137 |  |  | //Act |
| 138 |  |  | string actualresult = "Triangle is Scalene."; |
| 139 |  |  |  |
| 140 |  |  | //Assert |
| 141 |  |  | Assert.AreEqual(expectedoutput, actualresult); |
| 142 |  |  |  |
| 143 |  |  | } |
| 144 |  | } |  |
| 145 | } |  |  |

**Control Flow Graph**

**Cyclomatic Complexity Calculation**

Code Complexity formula is:

Cyclomatic complexity = E – N + 2 \* P

Where,

E is number of edges in the flow

N is number of nodes in the flow

P is number of nodes having the exit points

For Example:

E = 12

N = 10

P = 1

Cyclomatic complexity = 12 – 10 + 2 \* 1

= 2+2

= 4

**GITHUB Link**

<https://github.com/suhas-vemuganti/Assignment1-SQA>

