Assignment 2

Name – Ashutosh Chaudhary

Student ID – 8858206

Course Code – PROG 8173

We will use the following code for the testing.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using TriangleSolver;

using NUnit.Framework;

namespace TestTriangle

{

[TestFixture]

public class TestTriangle

{

[Test]

public void AnalyzeTriangle\_AllSidesEqual\_ReturnsEquilateral()

{

int sideA = 7;

int sideB = 7;

int sideC = 7;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Equilateral triangle"));

}

[Test]

public void Analyzetriangle\_TwoSidesEqual1\_ReturnsIsosceles()

{

int sideA = 9;

int sideB = 9;

int sideC = 5;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Isosceles triangle"));

}

[Test]

public void Analyzetriangle\_TwoSidesEqual2\_ReturnsIsosceles()

{

int sideA = 4;

int sideB = 4;

int sideC = 6;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Isosceles triangle"));

}

[Test]

public void Analyzetriangle\_TwoSidesEqual3\_ReturnsIsosceles()

{

int sideA = 9;

int sideB = 9;

int sideC = 5;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Isosceles triangle"));

}

[Test]

public void Analyzetriangle\_AllSidesDifferent1\_ReturnsScalene()

{

int sideA = 9;

int sideB = 7;

int sideC = 5;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Scalene triangle"));

}

[Test]

public void Analyzetriangle\_AllSidesDifferent2\_ReturnsScalene()

{

int sideA = 6;

int sideB = 4;

int sideC = 8;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Scalene triangle"));

}

[Test]

public void Analyzetriangle\_AllSidesDifferent3\_ReturnsScalene()

{

int sideA = 8;

int sideB = 6;

int sideC = 5;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Scalene triangle"));

}

[Test]

public void Analyzetriangle\_AllSidesDifferent4\_ReturnsScalene()

{

int sideA = 8;

int sideB = 7;

int sideC = 5;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Scalene triangle"));

}

[Test]

public void Analyzetriangle\_AllSidesDifferent5\_ReturnsScalene()

{

int sideA = 8;

int sideB = 3;

int sideC = 9;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Scalene triangle"));

}

[Test]

public void Analyzetriangle\_ZeroSide1\_ReturnsError()

{

int sideA = 0;

int sideB = 3;

int sideC = 9;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Invalid Triangle - a zero has been detected"));

}

[Test]

public void Analyzetriangle\_ZeroSide2\_ReturnsError()

{

int sideA = 2;

int sideB = 0;

int sideC = 8;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Invalid Triangle - a zero has been detected"));

}

[Test]

public void Analyzetriangle\_ZeroSide3\_ReturnsError()

{

int sideA = 5;

int sideB = 6;

int sideC = 0;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("Invalid Triangle - a zero has been detected"));

}

[Test]

public void Analyzetriangle\_InvalidTriangle1\_ReturnsInvalid()

{

int sideA = 1;

int sideB = 2;

int sideC = 3;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("INVALID!!"));

}

[Test]

public void Analyzetriangle\_InvalidTriangle2\_ReturnsInvalid()

{

int sideA = 2;

int sideB = 3;

int sideC = 1;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

Assert.That(result, Is.EqualTo("INVALID!!"));

}

[Test]

public void Analyzetriangle\_InvalidTriangle3\_ReturnsInvalid()

{

int sideA = 3;

int sideB = 2;

int sideC = 1;

string result = Triangle.AnalyzeTriangle(sideA, sideB, sideC);

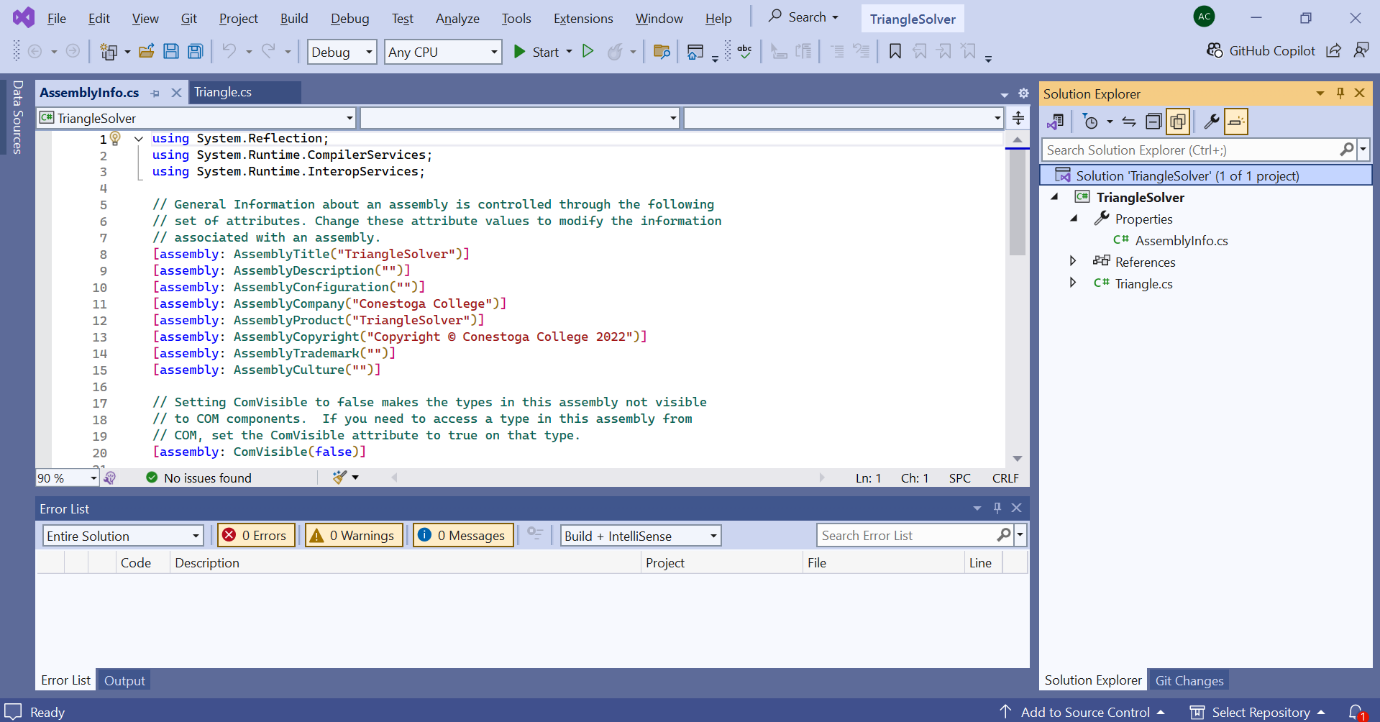
Assert.That(result, Is.EqualTo("INVALID!!"));

}

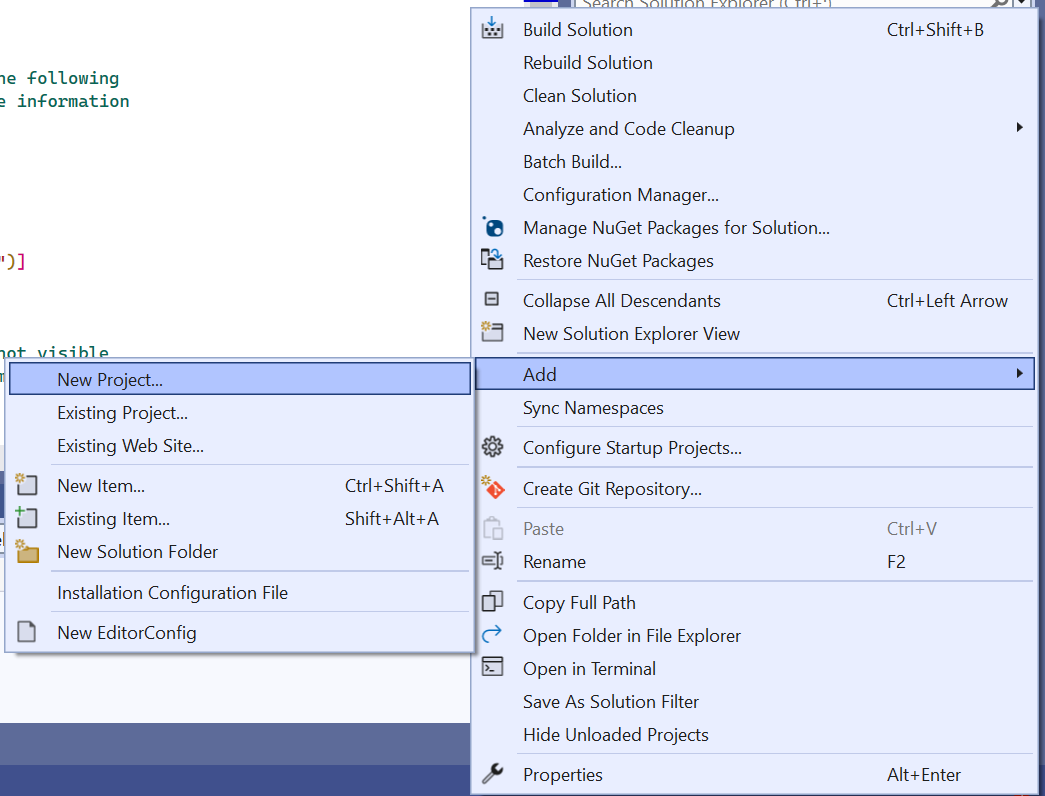
}

}

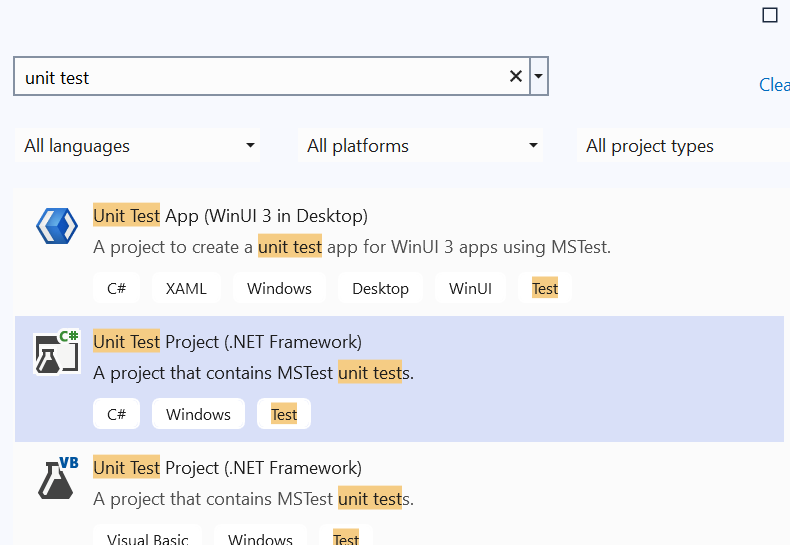
So Firstly, we will open the solution provided to us as a zip file along with the assignment.

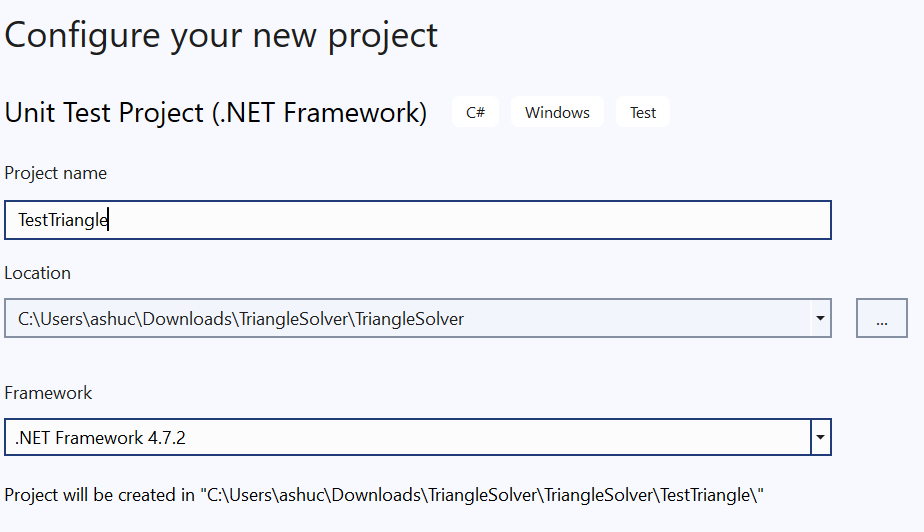


Now Add a new project into the same solution.



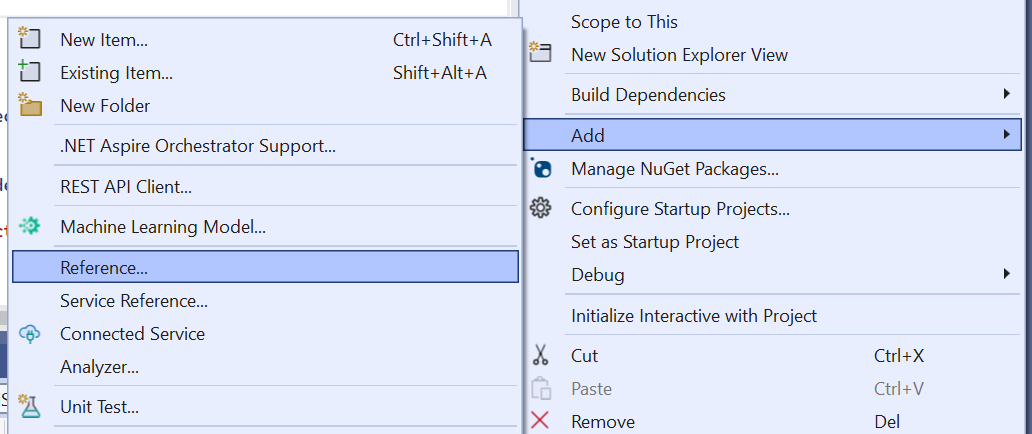
Configure the project for testing.



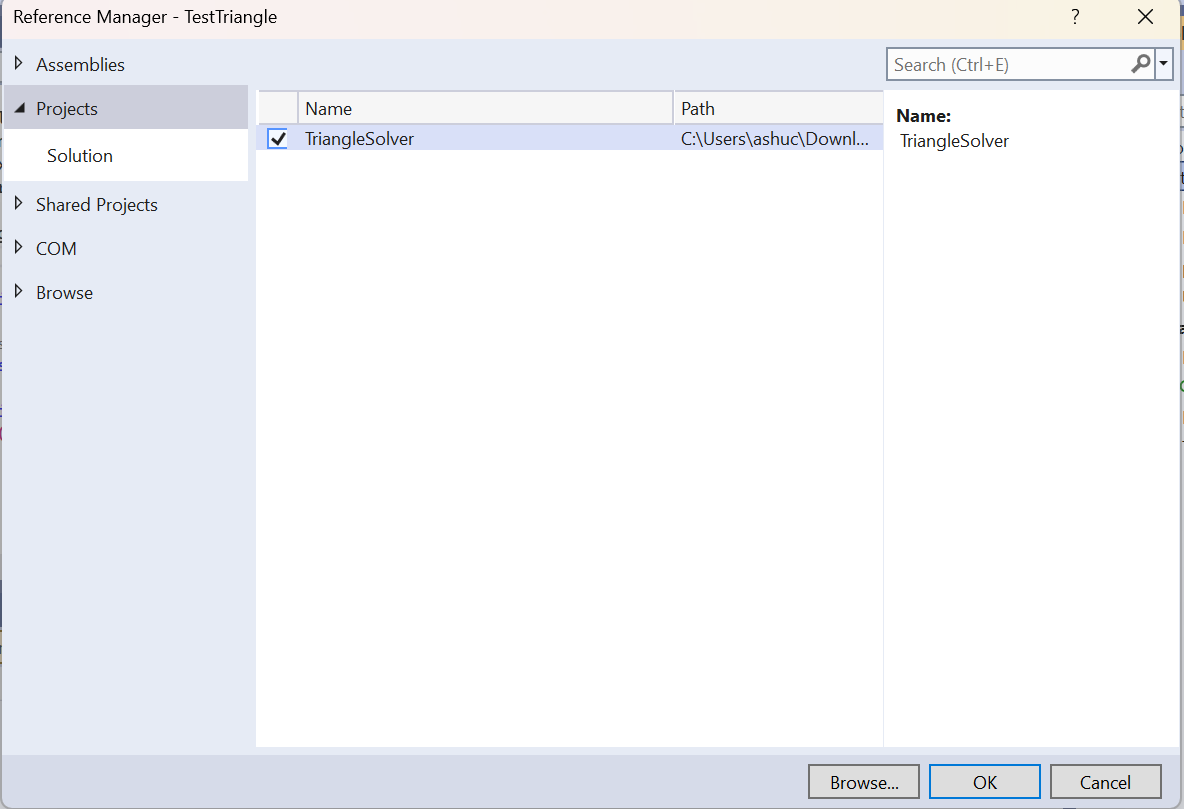


Create the project.

Then we will add the reference of this new project with the old and main project.



Select and add

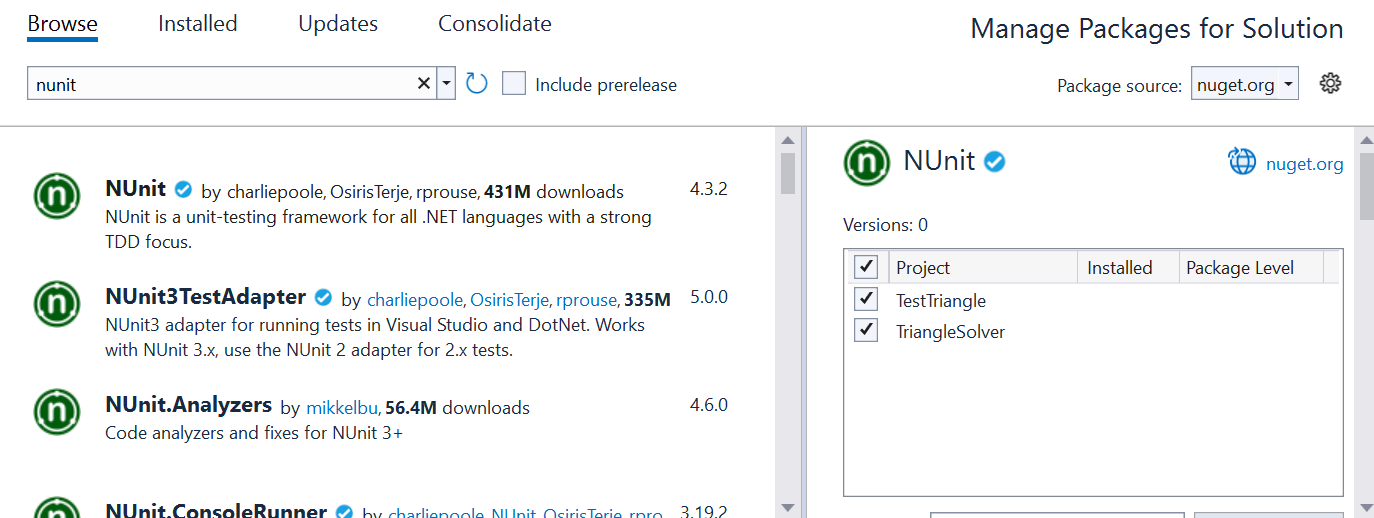


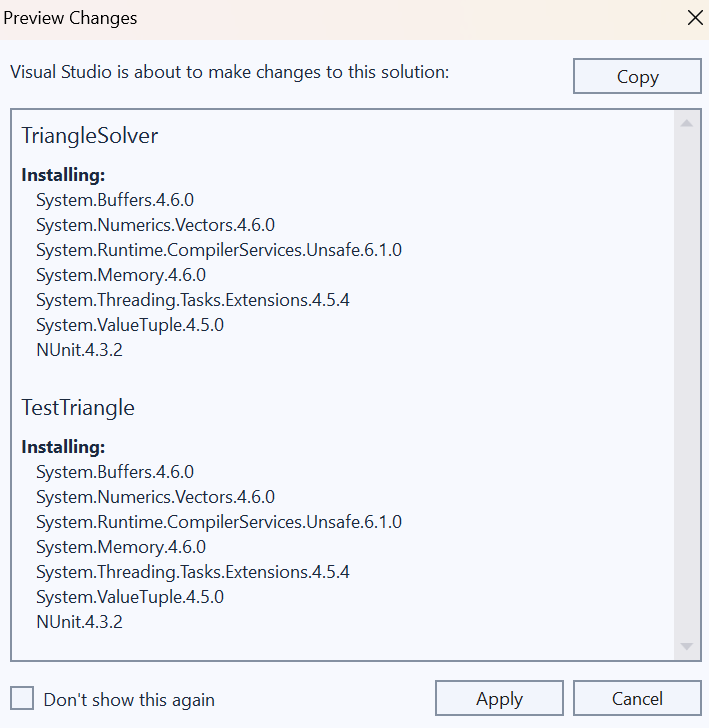
Now we will add packages.

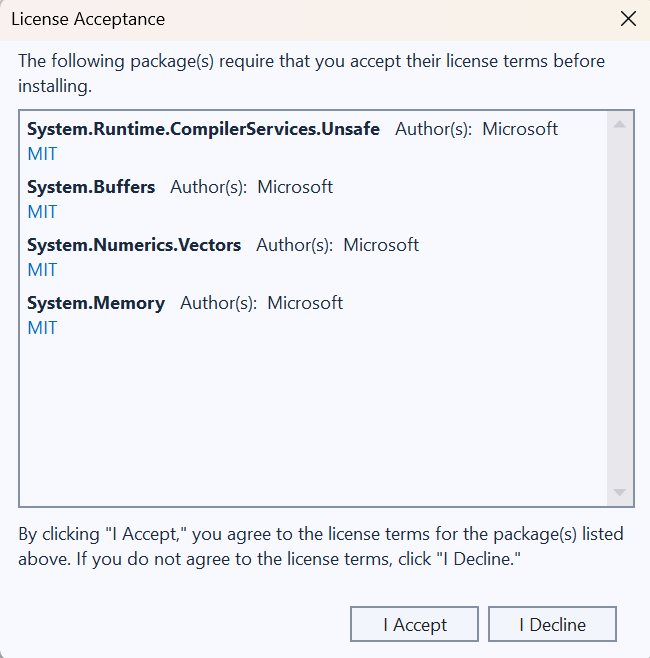
Install 3 packages.

* NUnit
* NUnit3TestAdapter
* NUnit.Console

Select, accept and Install.





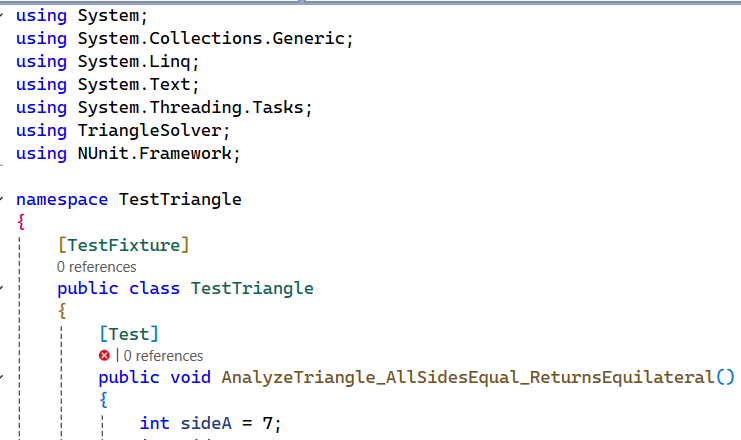


Add the following at top

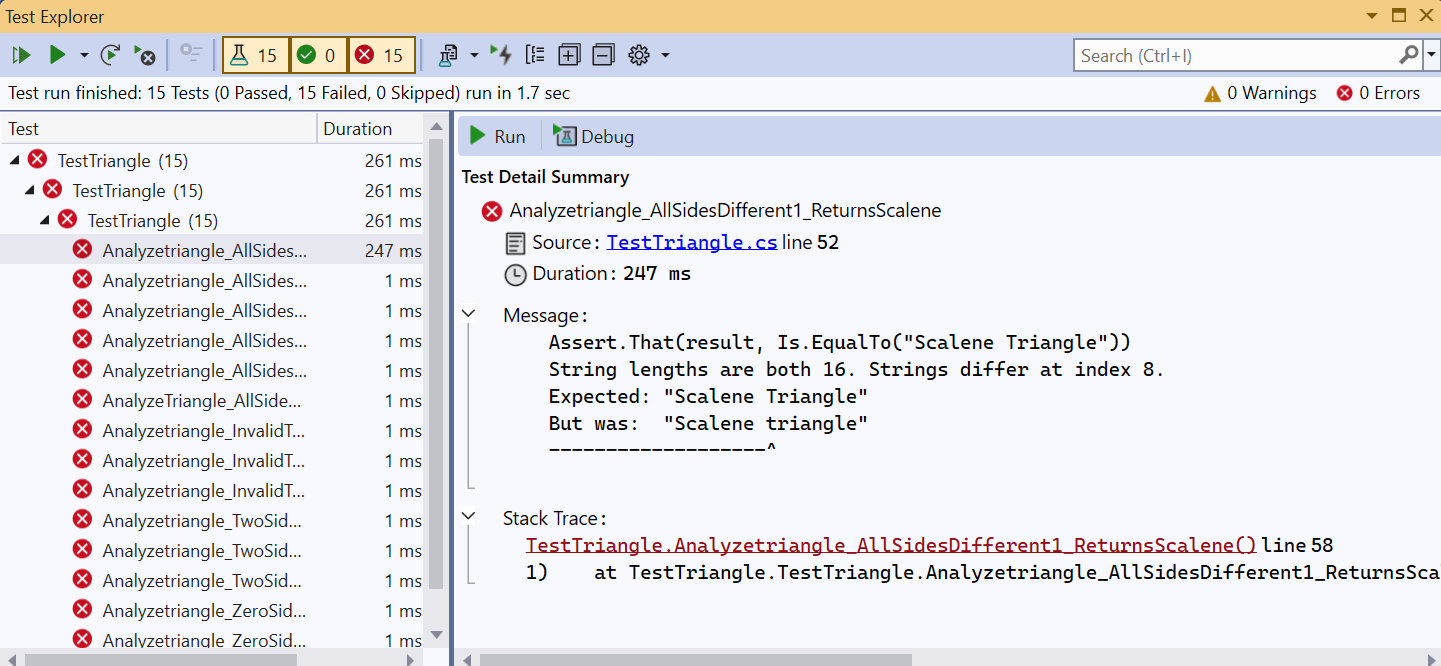
Using TriangleSolver

Using NUnit.Framework

Then write the complete code as given above.

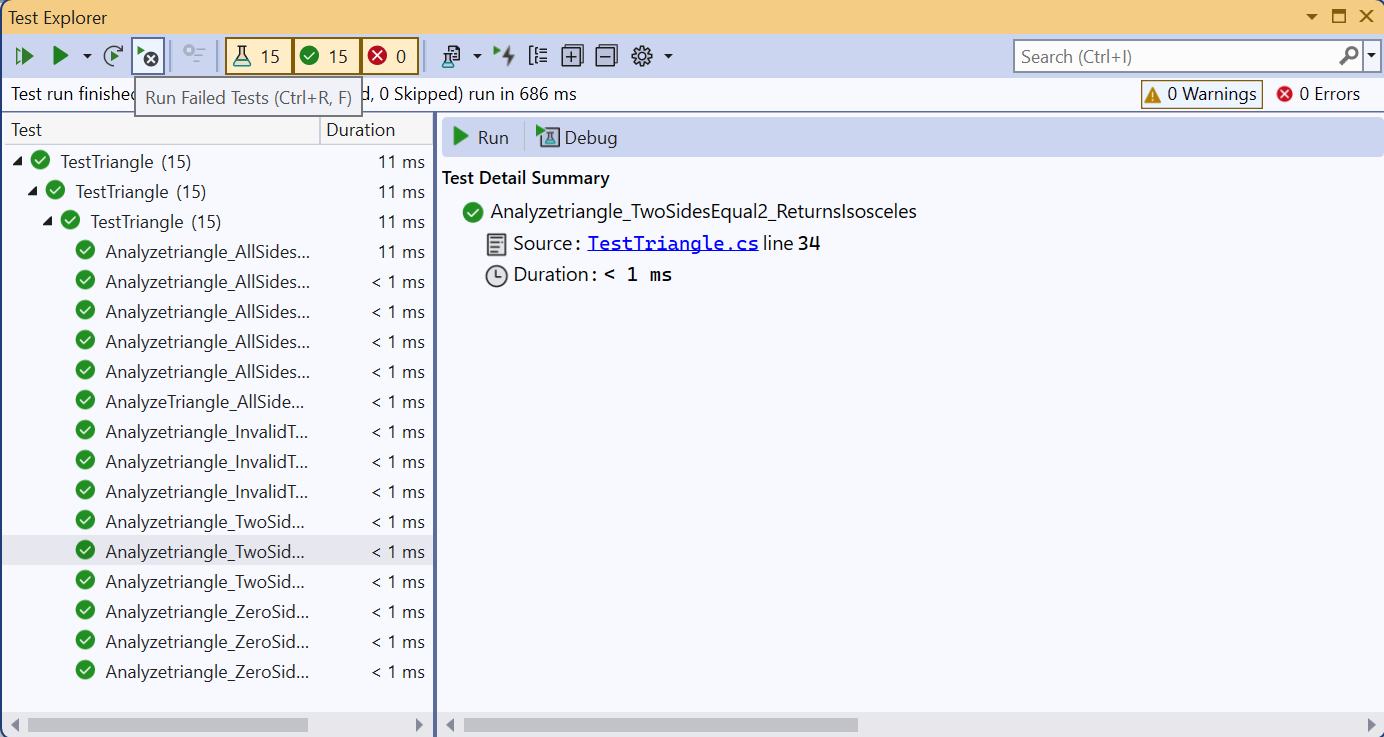


Now after this perform testing



Testing failed as there were silly minute errors.

We will fix them and we will test again.



All Testing Done and Succeeded.

No Error now.