1)

MathLibrary.cs

using System;

namespace CalcLibrary

{

public class MathLibrary : IMathLibrary

{

double result = 0;

public double Addition(double a, double b)

{

result = a + b;

return result;

}

public double Subtraction(double a, double b)

{

result = a - b;

return result;

}

public double Multiplication(double a, double b)

{

result = a \* b;

return result;

}

public double Division(double a, double b)

{

if (b == 0)

throw new ArgumentException("Second Parameter Can't be Zero");

result = a / b;

return result;

}

}

}

IMathLibrary.cs

namespace CalcLibrary

{

interface IMathLibrary

{

double Addition(double a, double b);

double Subtraction(double a, double b);

double Multiplication(double a, double b);

double Division(double a, double b);

}

}

CalcLibraryTests.cs

using CalcLibrary;

using NUnit.Framework;

namespace CalcLibrary\_Tests

{

[TestFixture]

public class CalculatorTests

{

[Test]

public void Addition\_InputTwoDoubleValues\_ReturnsSum()

{

MathLibrary mathLibrary = new MathLibrary();

//Arrange

double expectedResult = 4;

//Act

double Result = mathLibrary.Addition(2, 2);

//Assert

Assert.AreEqual(expectedResult, Result);

}

[Test]

public void Subtraction\_InputTwoDoubleValues\_ReturnsDifference()

{

MathLibrary mathLibrary = new MathLibrary();

double expectedResult = 5;

double Result = mathLibrary.Subtraction(10, 5);

Assert.AreEqual(expectedResult, Result);

}

[Test]

public void Multiplication\_InputTwoDoubleValues\_ReturnsProduct()

{

MathLibrary mathLibrary = new MathLibrary();

double expectedResult = 16;

double Result = mathLibrary.Multiplication(8, 2);

Assert.AreEqual(expectedResult, Result);

}

[Test]

public void Division\_InputTwoDoubleValues\_ReturnsQuotient()

{

MathLibrary mathLibrary = new MathLibrary();

double expectedResult = 10;

double Result = mathLibrary.Division(20, 2);

Assert.AreEqual(expectedResult, Result);

}

}

}

2)

SimpleCalculator.cs

using System;

namespace CalcLibrary

{

public class SimpleCalculator : IMathLibrary

{

double result = 0;

public double Addition(double a, double b)

{

result = a + b;

return result;

}

public double Subtraction(double a, double b)

{

result = a - b;

return result;

}

public double Multiplication(double a, double b)

{

result = a \* b;

return result;

}

public double Division(double a, double b)

{

if (b == 0)

throw new ArgumentException("Second Parameter Can't be Zero");

result = a / b;

return result;

}

public void AllClear()

{

result = 0;

}

public double GetResult

{

get { return result; }

}

}

}

IMathLibrary.cs

namespace CalcLibrary

{

interface IMathLibrary

{

double Addition(double a, double b);

double Subtraction(double a, double b);

double Multiplication(double a, double b);

double Division(double a, double b);

}

}

SimpleCalculatorTests.cs

using CalcLibrary;

using NUnit.Framework;

using System;

namespace CalcLibrary\_Tests

{

[TestFixture]

public class SimpleCalculatorTests

{

private SimpleCalculator \_simpleCalculator;

private double \_result;

[SetUp]

public void Initialize()

{

\_simpleCalculator = new SimpleCalculator();

\_result = 0;

}

[Test]

[TestCase(10, 20, 30)]

[TestCase(1, 2, 3)]

public void Addition\_ValidInputs\_ExpectedResult(double a, double b, double expected)

{

\_result = \_simpleCalculator.Addition(a, b);

Assert.AreEqual(expected, \_result);

}

[Test]

[TestCase(10, 20, -10)]

[TestCase(10, 2, 8)]

public void Subtraction\_ValidInputs\_ExpectedResult(double a, double b, double expected)

{

\_result = \_simpleCalculator.Subtraction(a, b);

Assert.AreEqual(expected, \_result);

}

[Test]

[TestCase(10, 20, 200)]

[TestCase(10, 0, 0)]

public void Multiplication\_ValidInputs\_ExpectedResult(double a, double b, double expected)

{

\_result = \_simpleCalculator.Multiplication(a, b);

Assert.AreEqual(expected, \_result);

}

[Test]

[TestCase(10, 2, 5)]

public void Division\_ValidInputs\_ExpectedResult(double a, double b, double expected)

{

\_result = \_simpleCalculator.Division(a, b);

Assert.AreEqual(expected, \_result);

}

[Test]

[TestCase(10, 0)]

public void Division\_InValidInputs\_ExpectedException(double a, double b)

{

Assert.Throws<ArgumentException>(() => \_simpleCalculator.Division(a, b));

}

[TearDown]

public void CleanUp()

{

\_simpleCalculator.AllClear();

\_result = \_simpleCalculator.GetResult;

}

}

}