using NUnit.Framework;

using CalcLibrary;

using System;

namespace CalcLibrary.Tests

{

[TestFixture]

public class SimpleCalculatorTests

{

private SimpleCalculator calculator;

[SetUp]

public void Setup()

{

calculator = new SimpleCalculator();

}

[TestCase(5, 3, 8)]

[TestCase(-2, 4, 2)]

public void Addition\_ReturnsExpected(double a, double b, double expected)

{

double result = calculator.Addition(a, b);

Assert.That(result, Is.EqualTo(expected));

}

[TestCase(10, 3, 7)]

public void Subtraction\_ReturnsExpected(double a, double b, double expected)

{

double result = calculator.Subtraction(a, b);

Assert.That(result, Is.EqualTo(expected));

}

[TestCase(2, 3, 6)]

public void Multiplication\_ReturnsExpected(double a, double b, double expected)

{

double result = calculator.Multiplication(a, b);

Assert.That(result, Is.EqualTo(expected));

}

[TestCase(10, 2, 5)]

public void Division\_ReturnsExpected(double a, double b, double expected)

{

double result = calculator.Division(a, b);

Assert.That(result, Is.EqualTo(expected));

}

[Test]

public void Division\_ByZero\_ThrowsException()

{

var ex = Assert.Throws<ArgumentException>(() => calculator.Division(5, 0));

Assert.That(ex.Message, Is.EqualTo("Second Parameter Can't be Zero"));

}

[Test]

public void AllClear\_ResetsResultToZero()

{

calculator.Addition(2, 2);

calculator.AllClear();

Assert.That(calculator.GetResult, Is.EqualTo(0));

}

[Test]

public void GetResult\_ReturnsLastResult()

{

calculator.Multiplication(3, 5);

Assert.That(calculator.GetResult, Is.EqualTo(15));

}

}

}