18.1-Writing unit tests delegates

Program

```
using System;

public class MathOperations
{
    // Delegate for a binary operation
    public delegate int BinaryOperation(int a, int b);

    // Method to multiply two numbers
    public int Multiply(int a, int b)
    {
        return a * b;
    }

    // Method to divide two numbers
    public int Divide(int a, int b)
    {
        if (b == 0)
        {
            throw new ArgumentException("Cannot divide by zero.");
        }
        return a / b;
    }
}
```

```
public class Program
{
    public static void Main(string[] args)
    {
        // Create an instance of MathOperations
        MathOperations mathOperations = new MathOperations();

        // Perform multiplication operation
        int resultMultiply = mathOperations.Multiply(5, 3);
        Console.WriteLine("Multiplication result: " + resultMultiply);

        // Perform division operation
        int resultDivide = mathOperations.Divide(10, 2);
        Console.WriteLine("Division result: " + resultDivide);

        // Run unit tests
        TestMultiplication();
        TestDivision();
    }

    // Unit test for multiplication operation
    public static void TestMultiplication()
    {
        MathOperations mathOperations = new MathOperations();
        MathOperations.BinaryOperation multiplyDelegate = mathOperations.Multiply;
        int result = multiplyDelegate(5, 3);
        Console.WriteLine("Unit Test - Multiplication: " + (result == 15 ? "Passed" : "Failed"));
    }

    // Unit test for division operation
    public static void TestDivision()
    {
        MathOperations mathOperations = new MathOperations();
        MathOperations.BinaryOperation divideDelegate = mathOperations.Divide;

        int result = divideDelegate(10, 2);
        Console.WriteLine("Unit Test - Division: " + (result == 5 ? "Passed" : "Failed"));
    }
}
```