

STEP 1: Create Calculator Class Library

1. Click Create a new project
2. Select Class Library (.NET)
3. Click Next
4. Project name: CalculatorLibrary
5. Click Create

STEP 2: Add Calculator Class

1. In Solution Explorer
2. Open Class .cs
3. Rename it to Calculator .cs

The screenshot shows the Visual Studio interface with the following details:

- Code Editor:** The main window displays the `Calculator.cs` file. The code defines a `Calculator` class with four methods: `Add`, `Subtract`, `Multiply`, and `Divide`. The `Divide` method includes exception handling for division by zero.
- Solution Explorer:** On the right, the Solution Explorer shows the project structure:
 - Solution 'CalculatorLibrary' (2 of 2 projects)**
 - CalculatorLibrary**
 - Properties
 - References
 - Calculator.cs** (selected)
 - CalculatorLibrary.Tests**
 - Properties
 - Dependencies
 - CalculatorTests.cs

At the bottom of the screen, status bar text indicates "Ln: 48, Ch: 1 SPC MIXED UTF-8 with BOM".

```
using System;

namespace CalculatorLibrary
{
    /// <summary>
    /// Simple Calculator with basic operations
    /// </summary>
    public class Calculator
    {
        /// <summary>
        /// Add two numbers
        /// </summary>
        public int Add(int a, int b)
        {
            return a + b;
        }

        /// <summary>
        /// Subtract two numbers
        /// </summary>
        public int Subtract(int a, int b)
        {
            return a - b;
        }

        /// <summary>
        /// Multiply two numbers
    }
}
```

```
/// </summary>
public int Multiply(int a, int b)
{
    return a * b;
}

/// <summary>
/// Divide two numbers
/// </summary>
/// <exception cref="DivideByZeroException">When dividing by zero</exception>
public int Divide(int a, int b)
{
    if (b == 0)
        throw new DivideByZeroException("Cannot divide by zero");

    return a / b;
}
}
```

STEP 3: Create NUnit Test Project

1. Right-click Solution
2. Click Add → New Project
3. Select NUnit Test Project (.NET)
4. Click Next
5. Project name: CalculatorLibrary.Tests

STEP 4 : Add Reference to CalculatorLibrary

1. Right-click CalculatorLibrary.Tests
2. Click Add → Project Reference

The screenshot shows the Visual Studio IDE interface. The Solution Explorer on the right displays two projects: 'CalculatorLibrary' and 'CalculatorLibrary.Tests'. The 'CalculatorLibrary' project contains files for 'Properties', 'References', 'Calculator.cs', and 'Dependencies'. The 'CalculatorLibrary.Tests' project contains files for 'Properties', 'References', 'CalculatorTests.cs', and 'Dependencies'. The main code editor window on the left shows the 'CalculatorTests.cs' file, which contains test code for a 'Calculator' class using NUnit. The code includes setup and teardown methods, as well as several test methods for addition, subtraction, multiplication, division, and division by zero.

```
CalculatorTests.cs
1  using CalculatorLibrary;
2  using NUnit.Framework;
3
4  namespace CalculatorNUnit
5  {
6      [TestFixture]
7      0 references
8      public class CalculatorTests
9      {
10         private Calculator _calculator = null;
11
12         [SetUp]
13         0 references
14         public void Setup()
15         {
16             _calculator = new Calculator();
17         }
18
19         [Test]
20         0 references
21         public void Add_5And3_Returns8()
22         {
23             int result = _calculator.Add(5, 3);
24             Assert.That(result, Is.EqualTo(8));
25         }
26
27         [Test]
28         0 references
29         public void Subtract_10Minus5_Returns5()
30         {
31             int result = _calculator.Subtract(10, 5);
32             Assert.That(result, Is.EqualTo(5));
33         }
34
35         [Test]
36         0 references
37         public void Multiply_6Times7_Returns42()
38         {
39             int result = _calculator.Multiply(6, 7);
40             Assert.That(result, Is.EqualTo(42));
41         }
42
43         [Test]
44         0 references
45         public void Divide_10By2_Returns5()
46         {
47             int result = _calculator.Divide(10, 2);
48             Assert.That(result, Is.EqualTo(5));
49         }
50
51         [Test]
52         0 references
53         public void Divide_ByZero_ThrowsException()
54     }
```

The screenshot shows the Visual Studio IDE interface. The left side features the code editor with the file 'CalculatorTests.cs' open. The code is written in C# and contains several test methods using the NUnit framework. The right side shows the 'Solution Explorer' pane, which displays two projects: 'CalculatorLibrary' and 'CalculatorLibrary.Tests'. The 'CalculatorLibrary' project contains files for 'Properties', 'References', and 'Calculator.cs'. The 'CalculatorLibrary.Tests' project contains files for 'Properties', 'References', 'Dependencies', and 'CalculatorTests.cs'. The 'CalculatorTests.cs' file is also listed under the 'CalculatorLibrary.Tests' node.

```
17
18
19     |     int result = _calculator.Add(5, 3);
20     |     Assert.That(result, Is.EqualTo(8));
21     |
22
23     [Test]
24     |     public void Subtract_10Minus5_Returns5()
25     {
26         |         int result = _calculator.Subtract(10, 5);
27         |         Assert.That(result, Is.EqualTo(5));
28     }
29
30     [Test]
31     |     public void Multiply_6Times7_Returns42()
32     {
33         |         int result = _calculator.Multiply(6, 7);
34         |         Assert.That(result, Is.EqualTo(42));
35     }
36
37     [Test]
38     |     public void Divide_10By2_Returns5()
39     {
40         |         int result = _calculator.Divide(10, 2);
41         |         Assert.That(result, Is.EqualTo(5));
42     }
43
44     [Test]
45     |     public void Divide_ByZero_ThrowsException()
46     {
47         |         Assert.Throws<DivideByZeroException>(() => _calculator.Divide(10, 0));
48     }
49
50
51     // Data-driven test example
52     [TestCase(2, 3, 8)]
53     [TestCase(10, 2, 20)]
54     [TestCase(4, 5, 0)]
55     |     public void Add_DataDrivenTests(int a, int b, int expected)
56     {
57         |         int result = _calculator.Add(a, b);
58         |         Assert.That(result, Is.EqualTo(expected));
59     }
60
61 }
62
```

CalculatorTests.cs Code:

```
using CalculatorLibrary;
using NUnit.Framework;
namespace CalculatorNUnit
{
    [TestFixture]
    public class CalculatorTests
    {
        private Calculator _calculator = null!;

        [SetUp]
        public void Setup()
        {

```

```
    _calculator = new Calculator();

}

[Test]
public void Add_5And3_Returns8()
{
    int result = _calculator.Add(5, 3);
    Assert.That(result, Is.EqualTo(8));
}

[Test]
public void Subtract_10Minus5_Returns5()
{
    int result = _calculator.Subtract(10, 5);
    Assert.That(result, Is.EqualTo(5));
}

[Test]
public void Multiply_6Times7_Returns42()
{
    int result = _calculator.Multiply(6, 7);
    Assert.That(result, Is.EqualTo(42));
}

[Test]
public void Divide_10By2_Returns5()
```

```

{
    int result = _calculator.Divide(10, 2);
    Assert.That(result, Is.EqualTo(5));
}

[Test]
public void Divide_ByZero_ThrowsException()
{
    Assert.Throws<DivideByZeroException>(() => _calculator.Divide(10, 0));
}

// Data-driven test example
[TestCase(2, 3, 5)]
[TestCase(10, 20, 30)]
[TestCase(-5, 5, 0)]
public void Add_DataDrivenTests(int a, int b, int expected)
{
    int result = _calculator.Add(a, b);
    Assert.That(result, Is.EqualTo(expected));
}
}
}

```

Step 5 : Run Unit Tests

1. Menu → Test → Test Explorer
2. Click **Run All Tests**

CalculatorTests.cs X Calculator.cs

Test Explorer

Test run finished: 8 Tests (8 Passed, 0 Failed, 0 Skipped) run in 200 ms

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
CalculatorLibrary.Tests (8)	7 ms		
CalculatorNUnit (8)	7 ms		
CalculatorTests (8)	7 ms		
Add_5And3_Returns8	4 ms		
Add_DataDrivenTests (3)	< 1 ms		
Divide_10By2_Returns5	< 1 ms		
Divide_ByZero_ThrowsException	3 ms		
Multiply_6Times7_Returns42	< 1 ms		
Subtract_10Minus5_Returns5	< 1 ms		

Run Debug P

Group Summary

CalculatorLibrary.Tests

Tests in group : 8

Total Duration: 7

Outcomes

8 Passed

```
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
```

```
    Assert.Throws<DivideByZeroException>(() => _calculator.Divide(10, 0));
}

// Data-driven test example
[TestCase(2, 3, 5)]
[TestCase(10, 20, 30)]
```