Formación de Testing en .NET

Nafarroako Gobernua
Gobierno de Navarra

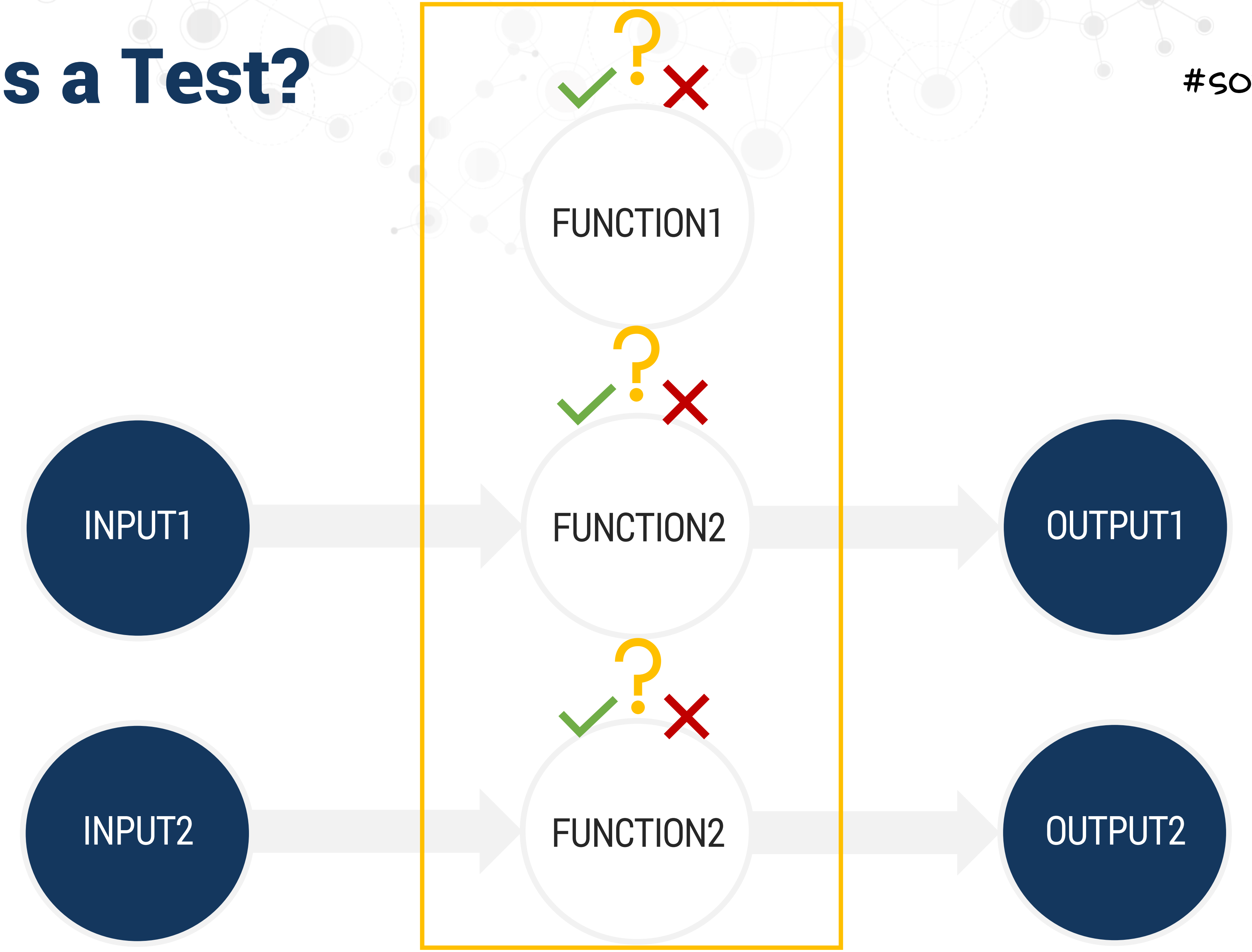
hiberus©

La compañía hiperespecializada en las TIC

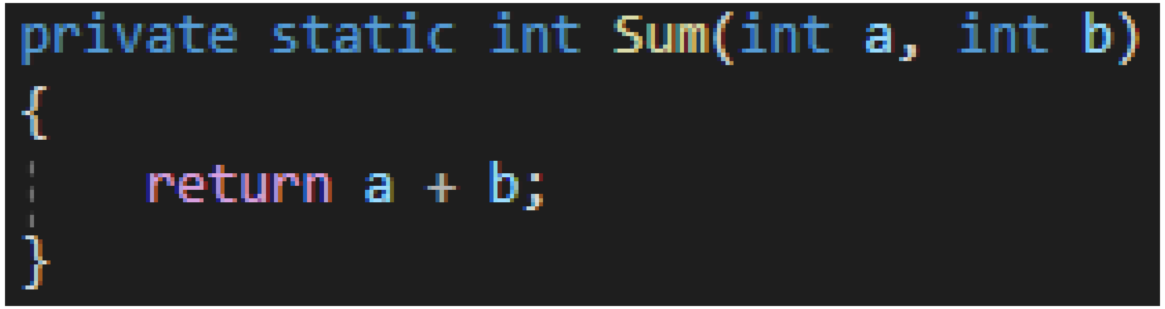
Tema 2: xUnit

Nafarroako Gobernua · Gobierno de Navarra

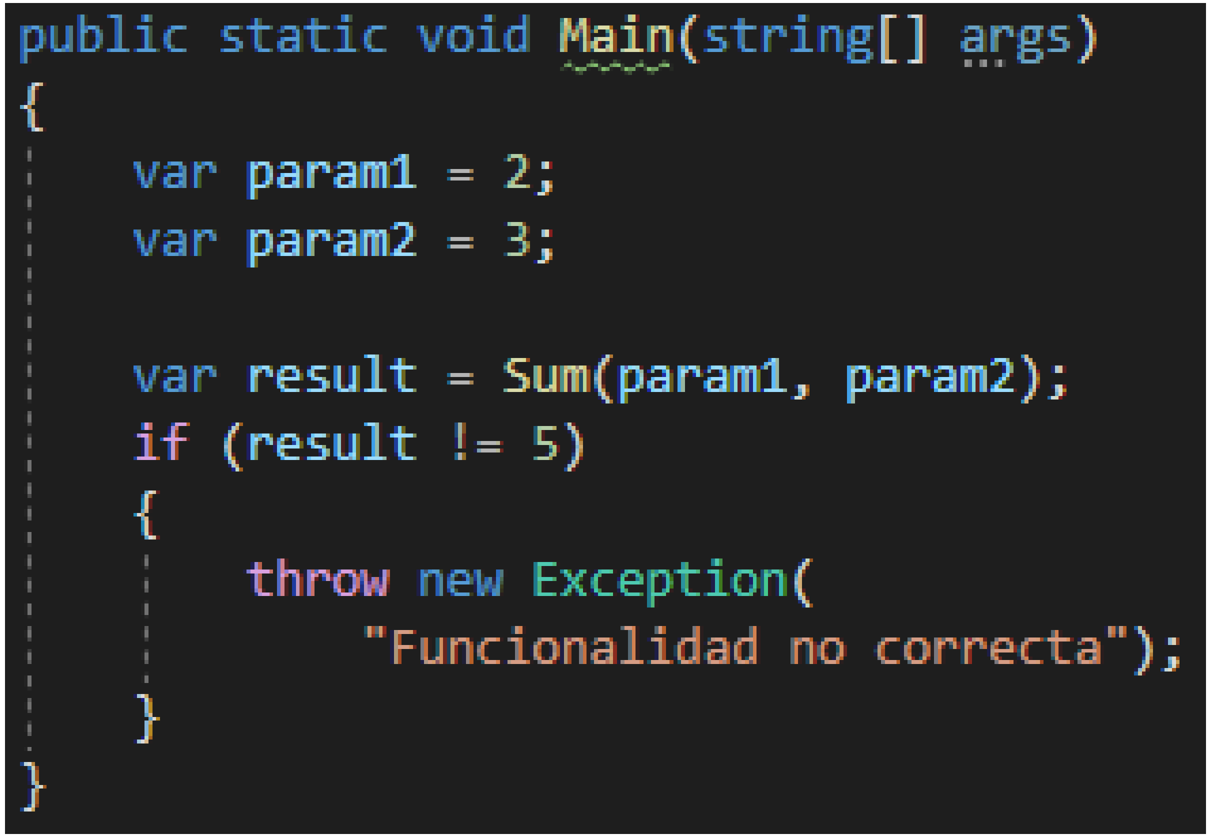
What is a Test?



What is a Test?



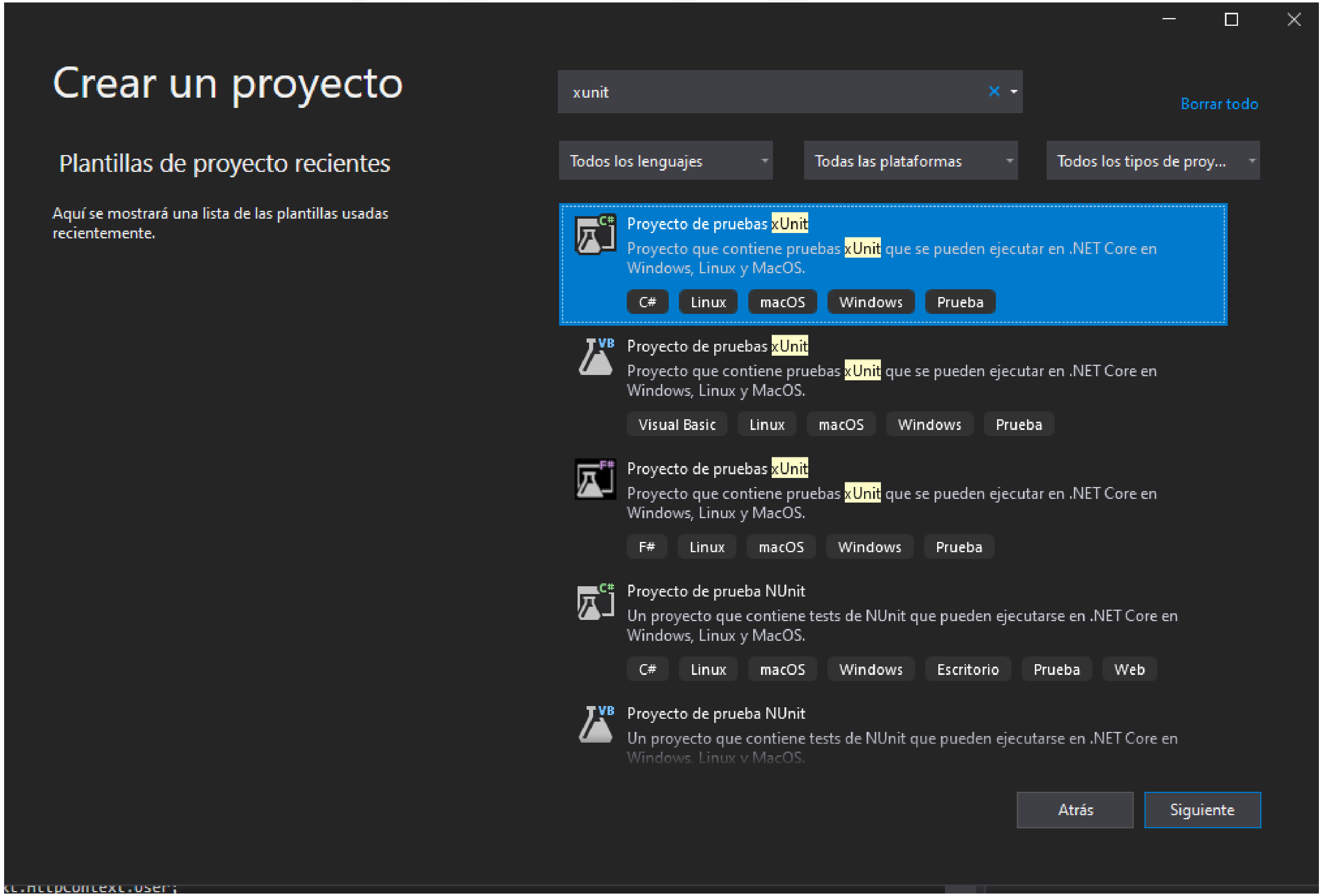
```
private static int Sum(int a, int b)
{
    return a + b;
}
```



```
public static void Main(string[] args)
{
    var param1 = 2;
    var param2 = 3;

    var result = Sum(param1, param2);
    if (result != 5)
    {
        throw new Exception(
            "Funcionalidad no correcta");
    }
}
```

xUnit 101

Creating a project



Crear un proyecto

Plantillas de proyecto recientes

Aquí se mostrará una lista de las plantillas usadas recientemente.

xunit

Borrar todo

Todos los lenguajes | Todas las plataformas | Todos los tipos de proy...

Proyecto de pruebas xUnit
Proyecto que contiene pruebas xUnit que se pueden ejecutar en .NET Core en Windows, Linux y MacOS.
C# | Linux | macOS | Windows | Prueba

Proyecto de pruebas xUnit
Proyecto que contiene pruebas xUnit que se pueden ejecutar en .NET Core en Windows, Linux y MacOS.
Visual Basic | Linux | macOS | Windows | Prueba

Proyecto de pruebas xUnit
Proyecto que contiene pruebas xUnit que se pueden ejecutar en .NET Core en Windows, Linux y MacOS.
F# | Linux | macOS | Windows | Prueba

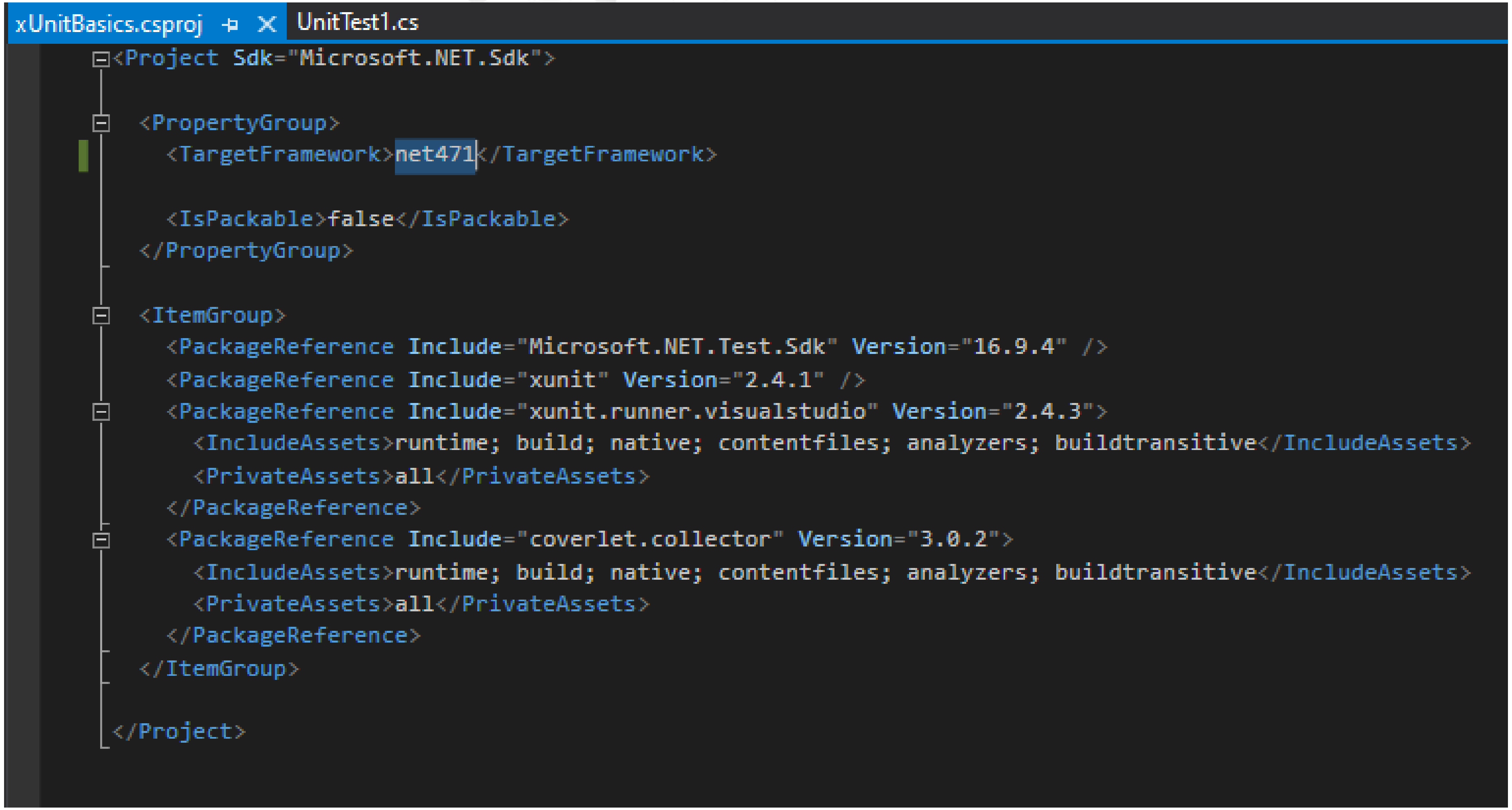
Proyecto de prueba NUnit
Un proyecto que contiene tests de NUnit que pueden ejecutarse en .NET Core en Windows, Linux y MacOS.
C# | Linux | macOS | Windows | Escritorio | Prueba | Web

Proyecto de prueba NUnit
Un proyecto que contiene tests de NUnit que pueden ejecutarse en .NET Core en Windows, Linux y MacOS.

```
xt.HttpContext.User;
```

xUnit 101

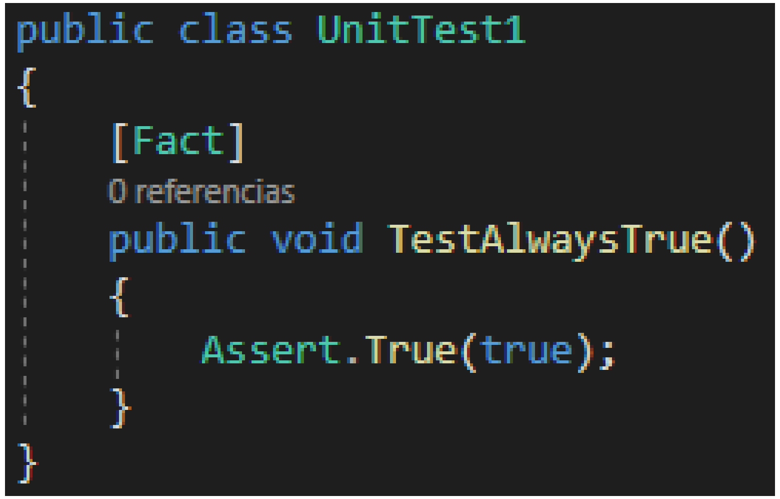
Creating a project



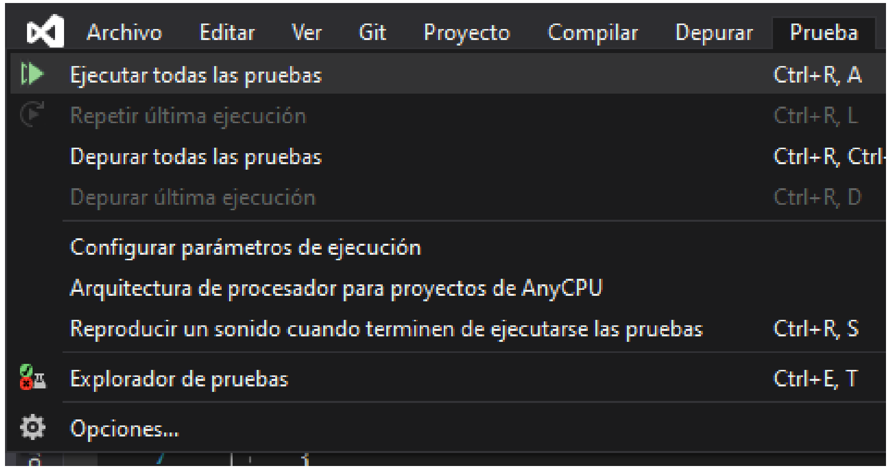
```xml
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <TargetFramework>net471</TargetFramework>

    <IsPackable>false</IsPackable>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="Microsoft.NET.Test.Sdk" Version="16.9.4" />
    <PackageReference Include="xunit" Version="2.4.1" />
    <PackageReference Include="xunit.runner.visualstudio" Version="2.4.3">
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
      <PrivateAssets>all</PrivateAssets>
    </PackageReference>
    <PackageReference Include="coverlet.collector" Version="3.0.2">
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
      <PrivateAssets>all</PrivateAssets>
    </PackageReference>
  </ItemGroup>

</Project>
```

xUnit 101

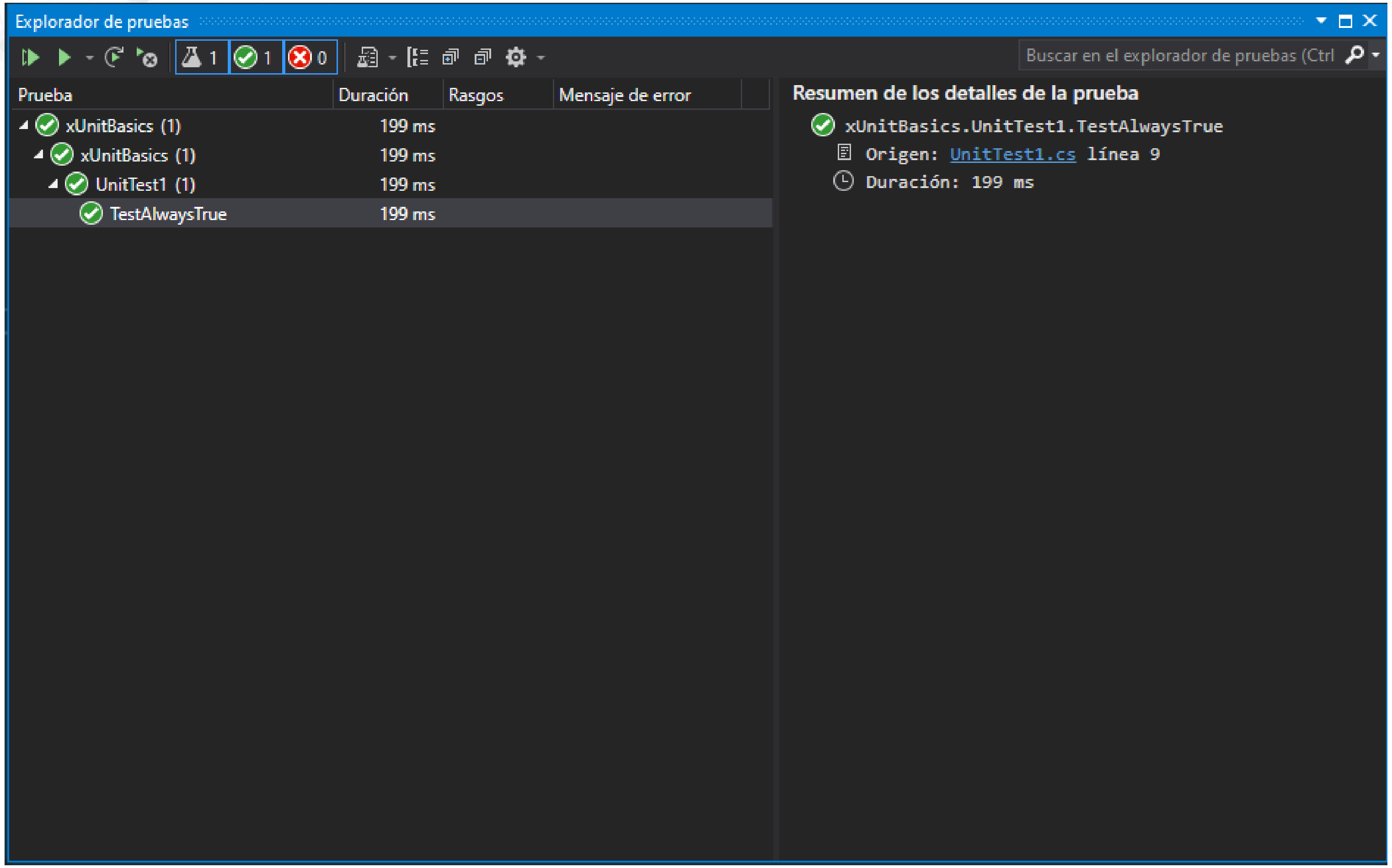
My first test



```
public class UnitTest1
{
    [Fact]
    0 referencias
    public void TestAlwaysTrue()
    {
        Assert.True(true);
    }
}
```



Archivo
Editar
Ver
Git
Proyecto
Compilar
Depurar
Prueba
Ejecutar todas las pruebas Ctrl+R, A
Repetir última ejecución Ctrl+R, L
Depurar todas las pruebas Ctrl+R, Ctrl-
Depurar última ejecución Ctrl+R, D
Configurar parámetros de ejecución
Arquitectura de procesador para proyectos de AnyCPU
Reproducir un sonido cuando terminen de ejecutarse las pruebas Ctrl+R, S
Explorador de pruebas Ctrl+E, T
Opciones...



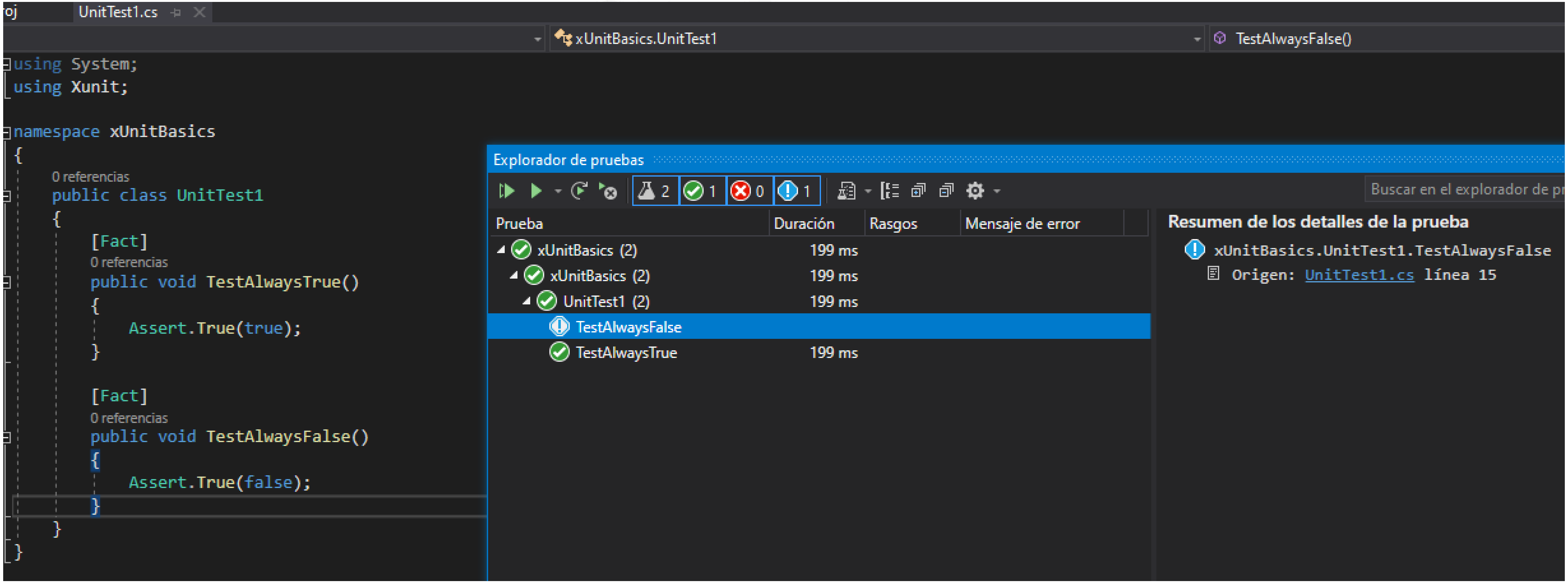
Prueba	Duración	Rasgos	Mensaje de error
✓ xUnitBasics (1)	199 ms		
✓ xUnitBasics (1)	199 ms		
✓ UnitTest1 (1)	199 ms		
✓ TestAlwaysTrue	199 ms		

Resumen de los detalles de la prueba

```
✓ xUnitBasics.UnitTest1.TestAlwaysTrue
   Origen: UnitTest1.cs línea 9
   Duración: 199 ms
```

xUnit 101

My first test



```
using System;
using Xunit;

namespace xUnitBasics
{
    0 referencias
    public class UnitTest1
    {
        [Fact]
        0 referencias
        public void TestAlwaysTrue()
        {
            Assert.True(true);
        }

        [Fact]
        0 referencias
        public void TestAlwaysFalse()
        {
            Assert.True(false);
        }
    }
}
```

Explorador de pruebas

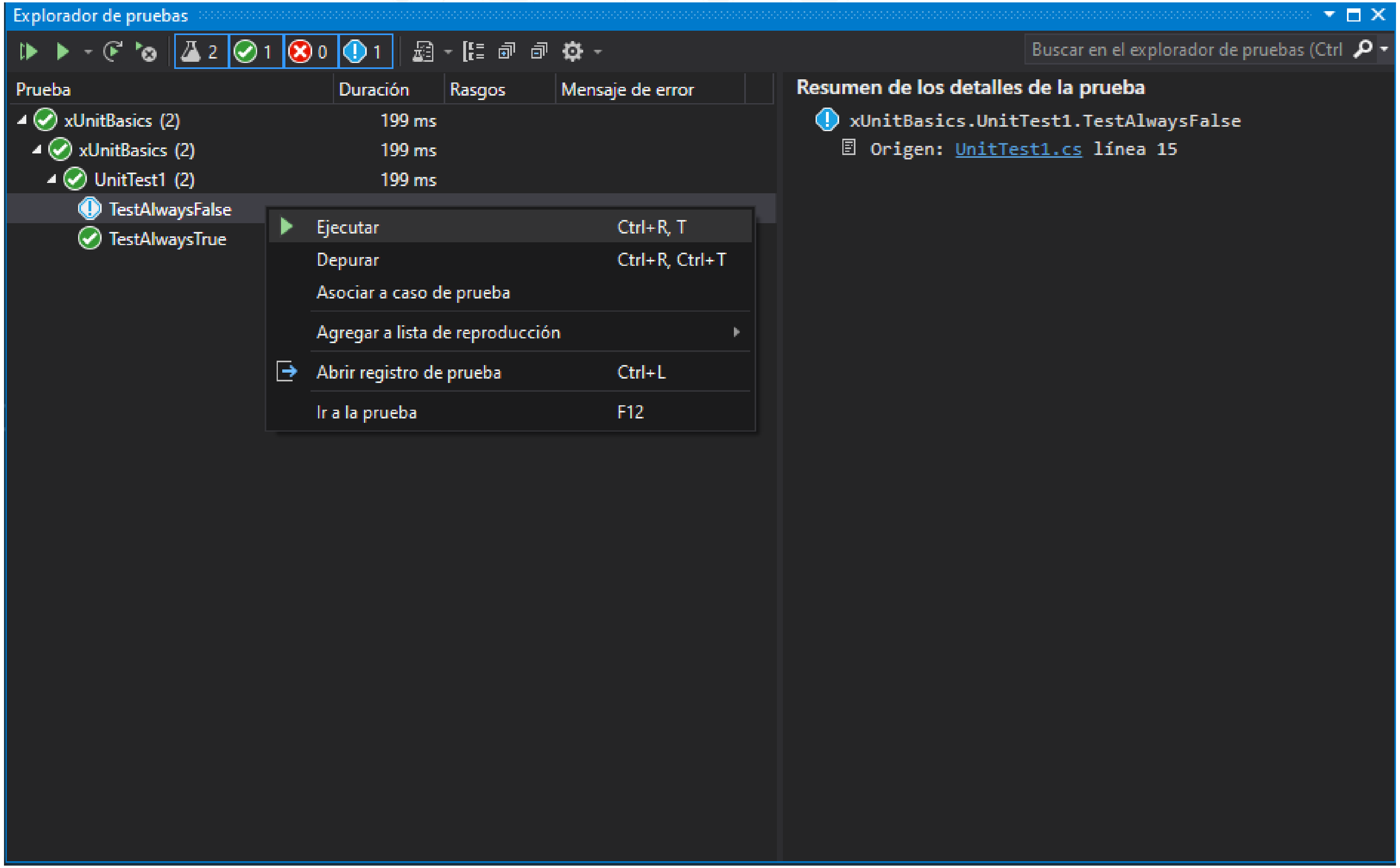
Prueba	Duración	Rasgos	Mensaje de error
xUnitBasics (2)	199 ms		
xUnitBasics (2)	199 ms		
UnitTest1 (2)	199 ms		
TestAlwaysFalse			
TestAlwaysTrue	199 ms		

Resumen de los detalles de la prueba

```
xUnitBasics.UnitTest1.TestAlwaysFalse
  Origen: UnitTest1.cs línea 15
```

xUnit 101

My first test



Prueba	Duración	Rasgos	Mensaje de error
xUnitBasics (2)	199 ms		
xUnitBasics (2)	199 ms		
UnitTest1 (2)	199 ms		
TestAlwaysFalse			
TestAlwaysTrue			

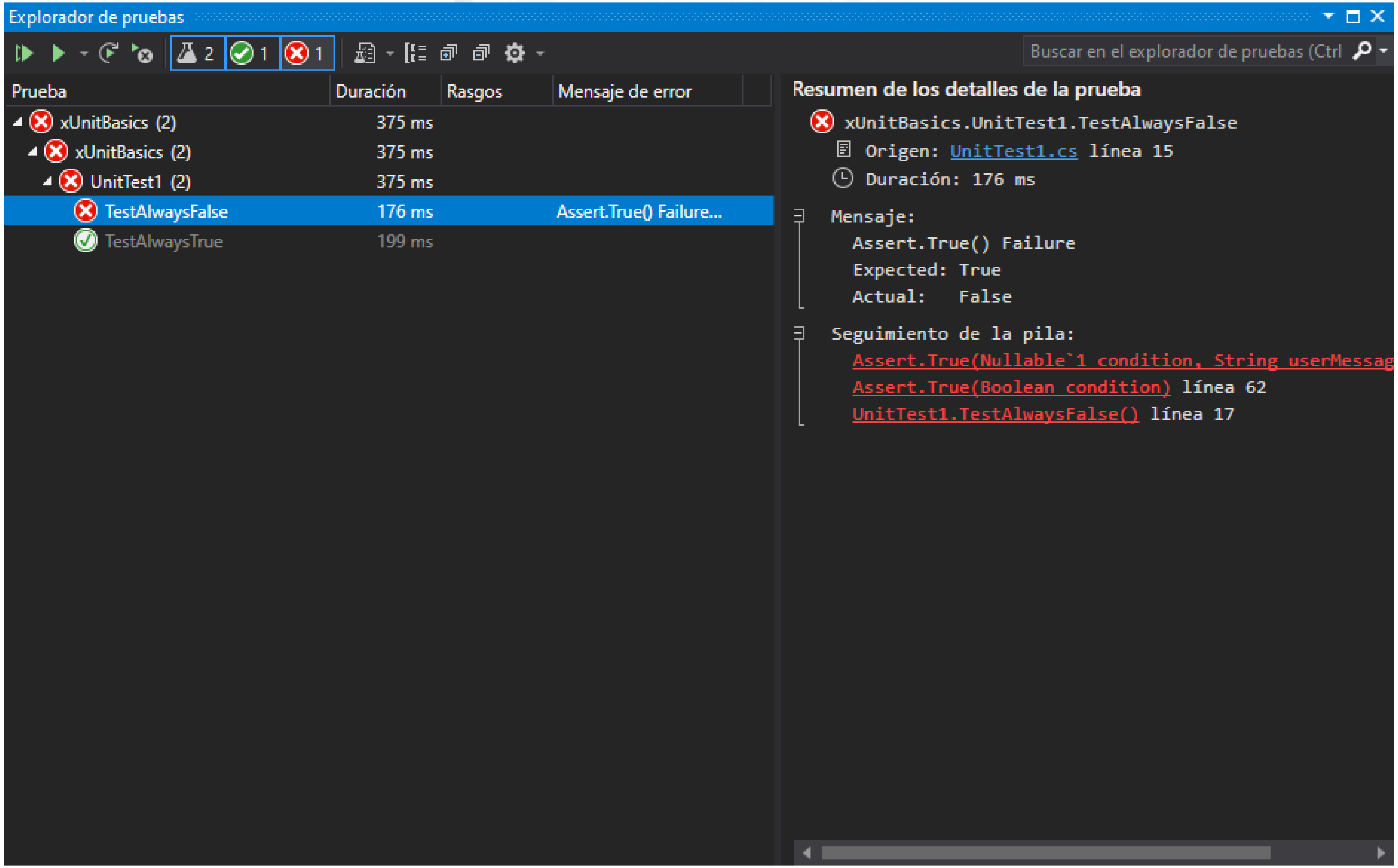
Resumen de los detalles de la prueba

```
xUnitBasics.UnitTest1.TestAlwaysFalse
  Origen: UnitTest1.cs línea 15
```

xUnit 101

My first test

#somoshiberus



Prueba	Duración	Rasgos	Mensaje de error
xUnitBasics (2)	375 ms		
xUnitBasics (2)	375 ms		
UnitTest1 (2)	375 ms		
TestAlwaysFalse	176 ms		Assert.True() Failure...
TestAlwaysTrue	199 ms		

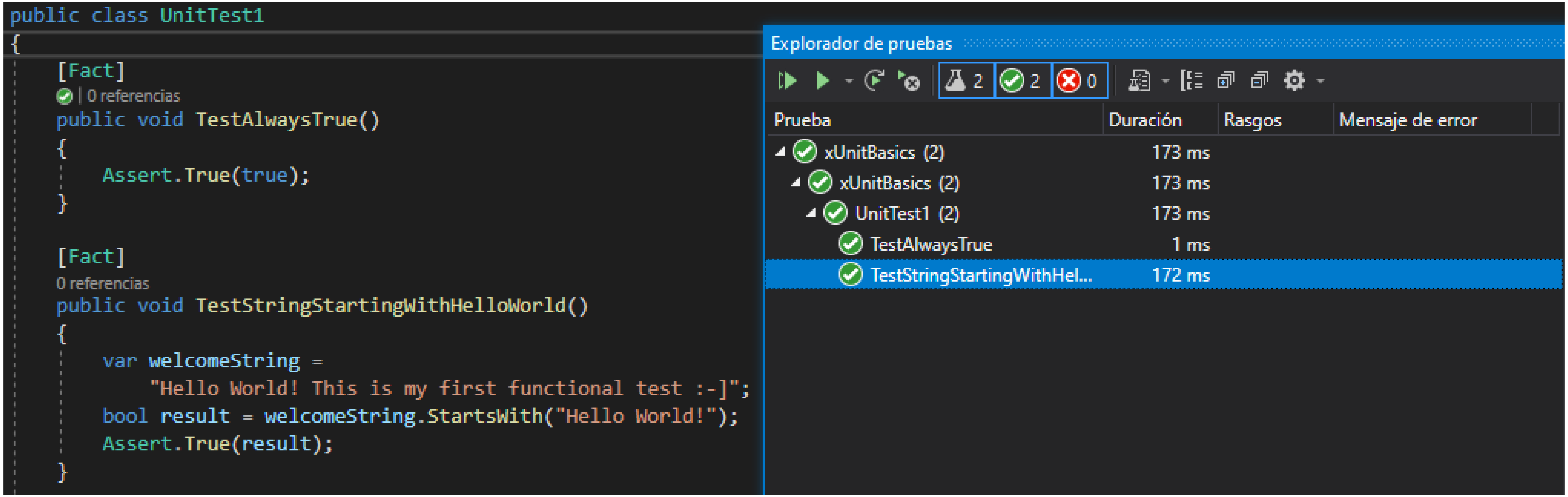
Resumen de los detalles de la prueba

```
xUnitBasics.UnitTest1.TestAlwaysFalse
  Origen: UnitTest1.cs línea 15
  Duración: 176 ms

Mensaje:
  Assert.True() Failure
  Expected: True
  Actual:   False

Seguimiento de la pila:
  Assert.True(Nullable`1 condition, String userMessag
  Assert.True(Boolean condition) línea 62
  UnitTest1.TestAlwaysFalse() línea 17
```

Nafarroako Gobernua Gobierno de Navarra

xUnit 101

My first test



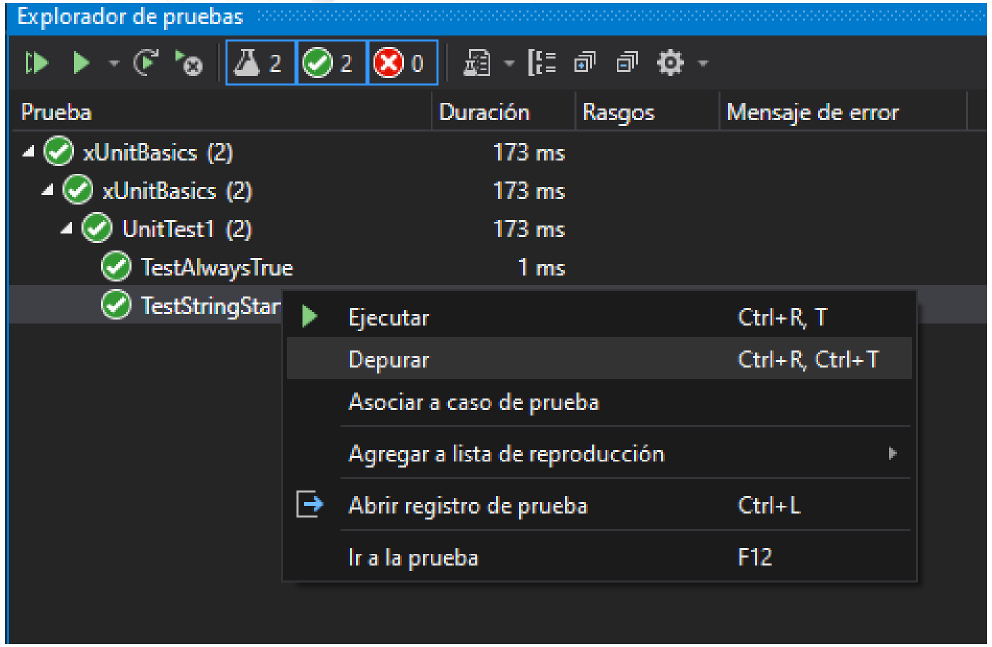
```
public class UnitTest1
{
    [Fact]
    ✓ | 0 referencias
    public void TestAlwaysTrue()
    {
        Assert.True(true);
    }

    [Fact]
    0 referencias
    public void TestStringStartingWithHelloWorld()
    {
        var welcomeString =
            "Hello World! This is my first functional test :-]";
        bool result = welcomeString.StartsWith("Hello World!");
        Assert.True(result);
    }
```

Explorador de pruebas

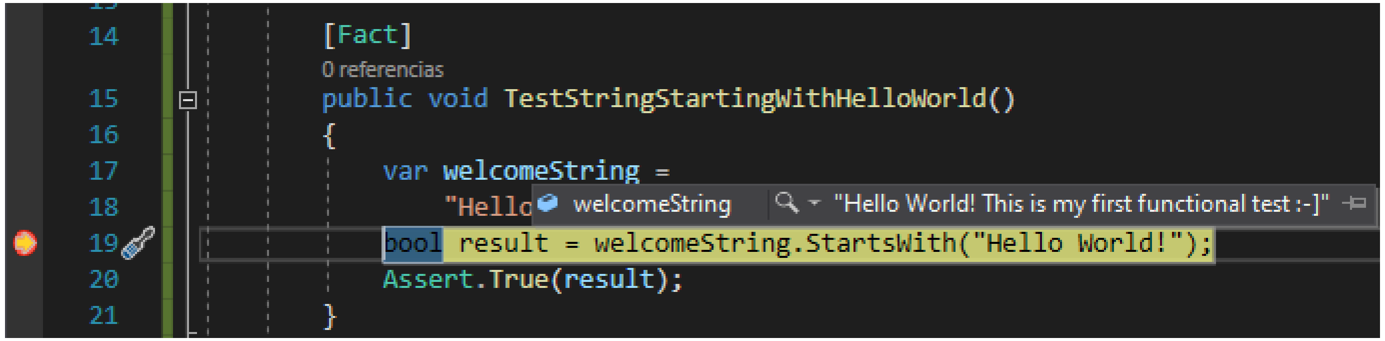
Prueba	Duración	Rasgos	Mensaje de error
✓ xUnitBasics (2)	173 ms		
✓ xUnitBasics (2)	173 ms		
✓ UnitTest1 (2)	173 ms		
✓ TestAlwaysTrue	1 ms		
✓ TestStringStartingWithHel...	172 ms		

xUnit 101

My first test



Prueba	Duración	Rasgos	Mensaje de error
xUnitBasics (2)	173 ms		
xUnitBasics (2)	173 ms		
UnitTest1 (2)	173 ms		
TestAlwaysTrue	1 ms		
TestStringStar			

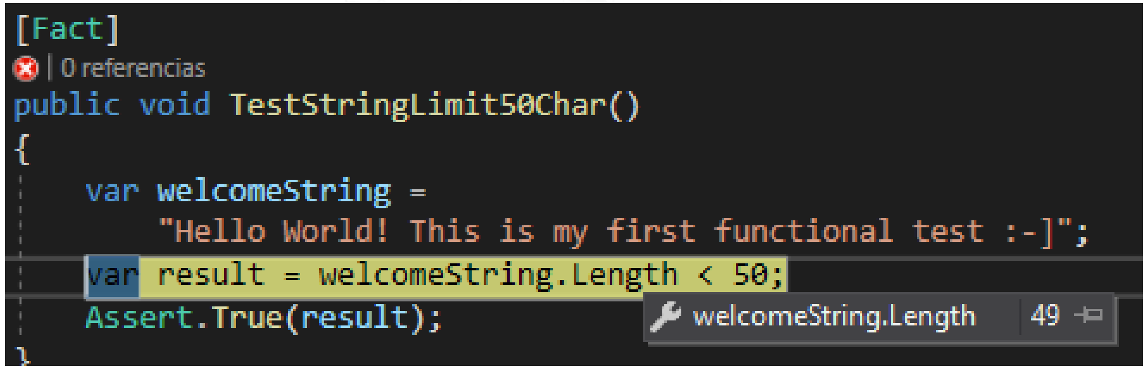

welcomeString "Hello World! This is my first functional test :-]"

```
[Fact]
0 referencias
public void TestStringStartingWithHelloWorld()
{
    var welcomeString =
        "Hello
    bool result = welcomeString.StartsWith("Hello World!");
    Assert.True(result);
}
```

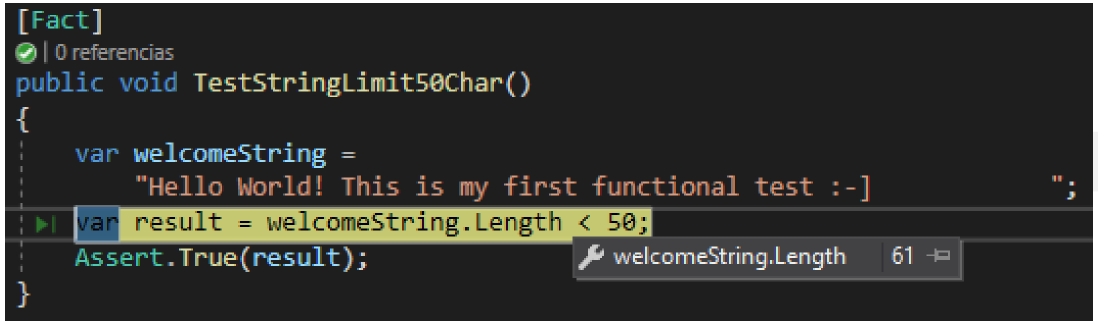
xUnit 101

My first test

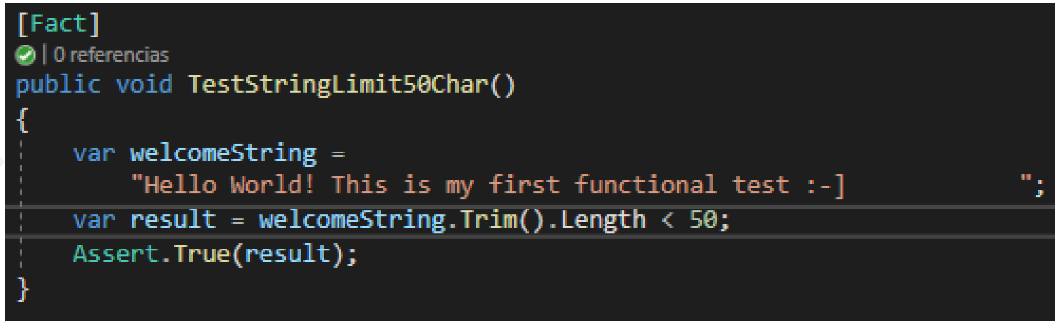
#somoshiberus



```
[Fact]
| 0 referencias
public void TestStringLimit50Char()
{
    var welcomeString =
        "Hello World! This is my first functional test :-]";
    var result = welcomeString.Length < 50;
    Assert.True(result);                  welcomeString.Length   49
}
```



```
[Fact]
| 0 referencias
public void TestStringLimit50Char()
{
    var welcomeString =
        "Hello World! This is my first functional test :-]           ";
    var result = welcomeString.Length < 50;
    Assert.True(result);                  welcomeString.Length   61
}
```

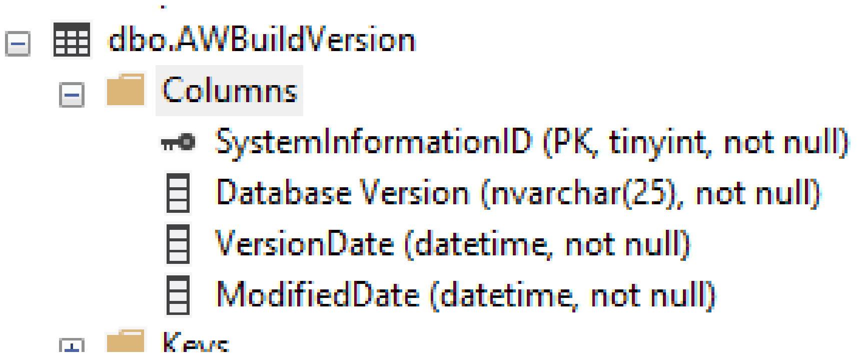


```
[Fact]
| 0 referencias
public void TestStringLimit50Char()
{
    var welcomeString =
        "Hello World! This is my first functional test :-]           ";
    var result = welcomeString.Trim().Length < 50;
    Assert.True(result);
}
```

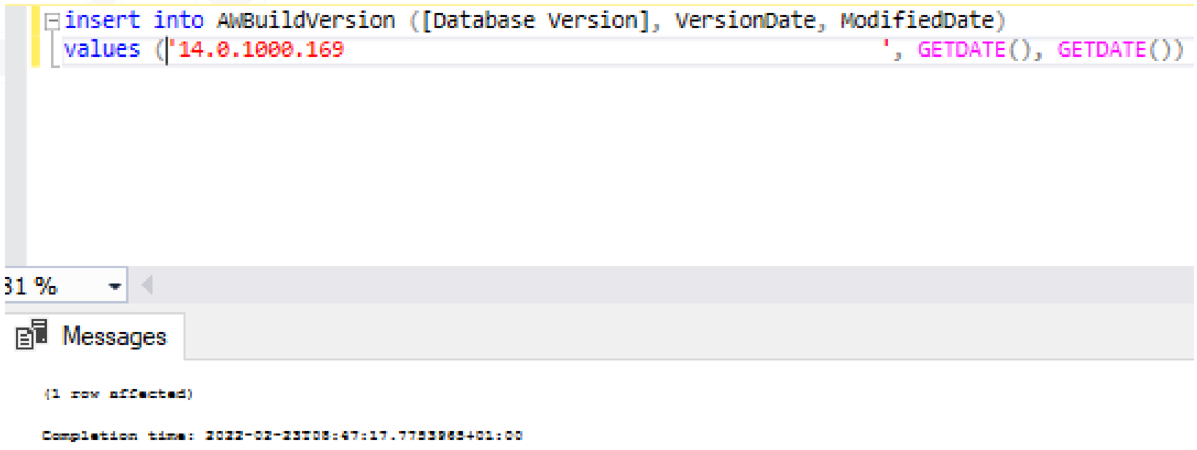
xUnit 101

My first test

#somoshiberus

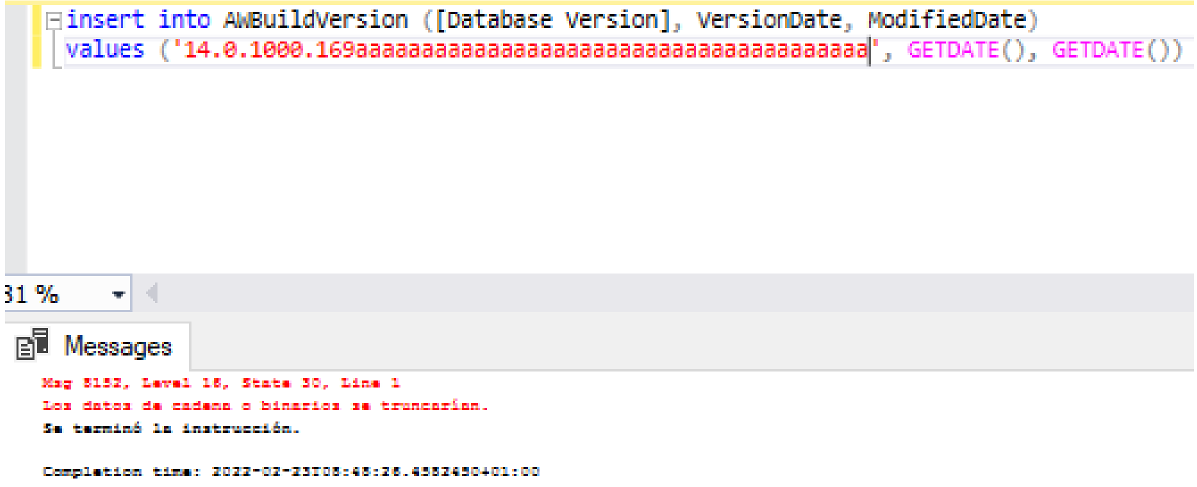


- dbo.AWBuildVersion
  - Columns
    - SystemInformationID (PK, tinyint, not null)
    - Database Version (nvarchar(25), not null)
    - VersionDate (datetime, not null)
    - ModifiedDate (datetime, not null)
  - Keys



```
insert into AWBuildVersion ([Database Version], VersionDate, ModifiedDate)
values ('14.0.1000.169                                        ', GETDATE(), GETDATE())
```

```
(1 row affected)

Completion time: 2022-02-23T08:47:17.7753969+01:00
```

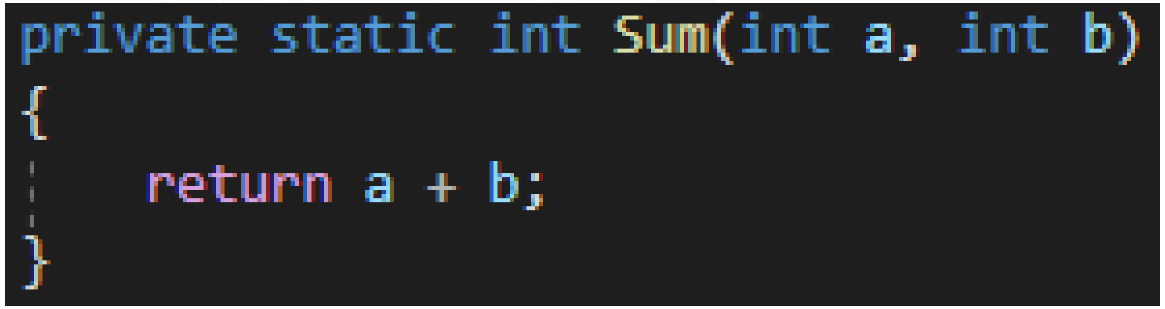


```
insert into AWBuildVersion ([Database Version], VersionDate, ModifiedDate)
values ('14.0.1000.169aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa', GETDATE(), GETDATE())
```

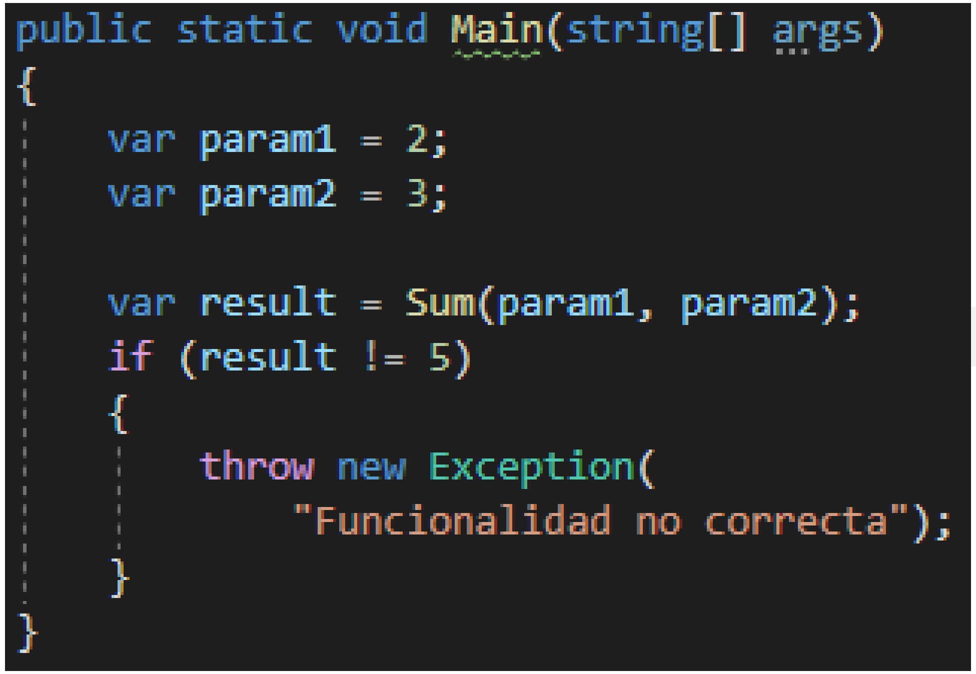
```
Msg 8152, Level 16, State 30, Line 1
Los datos de cadena o binarios se truncarían.
Se terminó la instrucción.

Completion time: 2022-02-23T08:48:26.4552450+01:00
```

xUnit 101

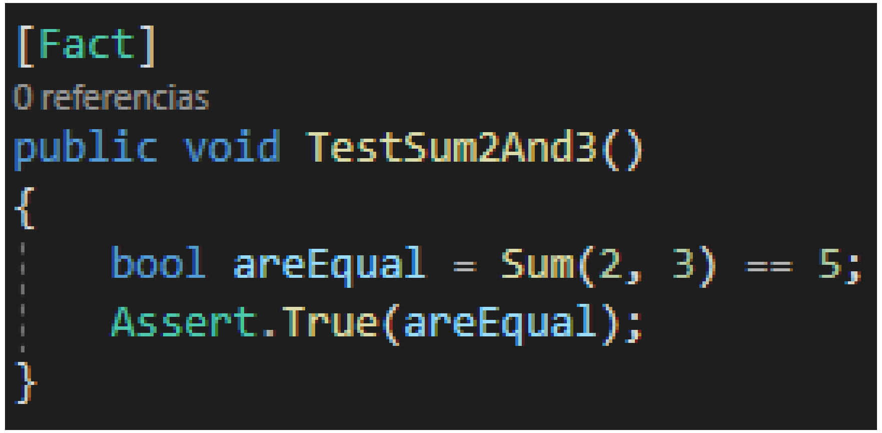
How to test



```
private static int Sum(int a, int b)
{
    return a + b;
}
```



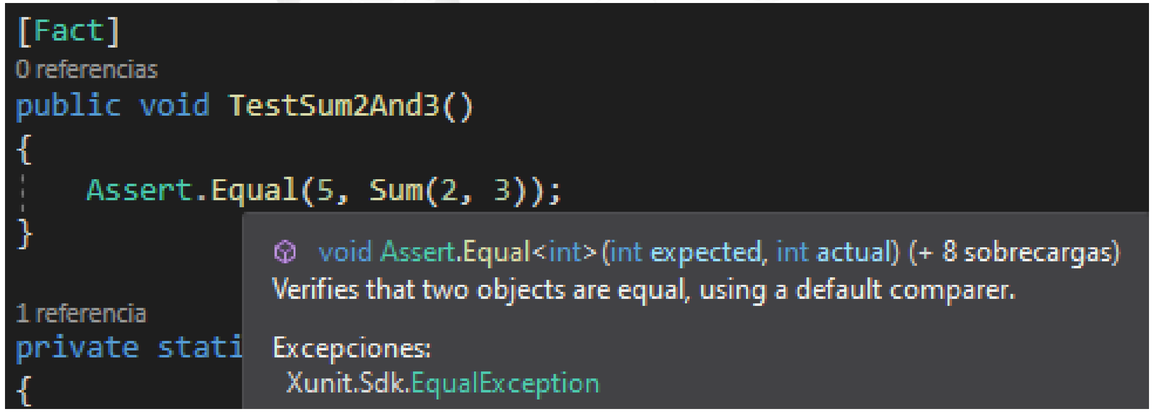
```
public static void Main(string[] args)
{
    var param1 = 2;
    var param2 = 3;

    var result = Sum(param1, param2);
    if (result != 5)
    {
        throw new Exception(
            "Funcionalidad no correcta");
    }
}
```



```
[Fact]
0 referencias
public void TestSum2And3()
{
    bool areEqual = Sum(2, 3) == 5;
    Assert.True(areEqual);
}
```

xUnit 101

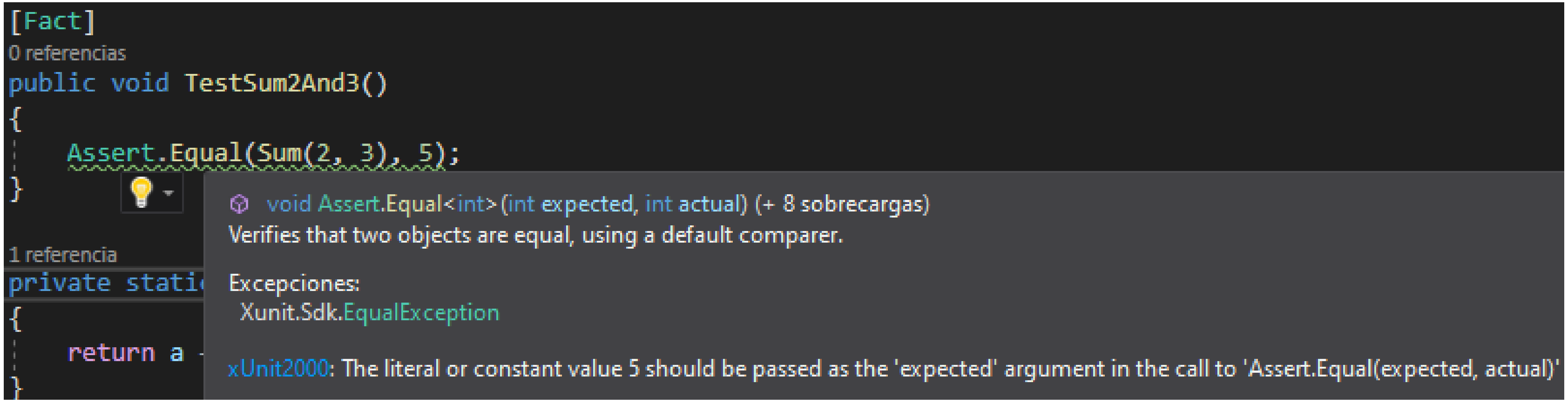
How to test



```
[Fact]
0 referencias
public void TestSum2And3()
{
    Assert.Equal(5, Sum(2, 3));
}

1 referencia
private stati
{
```

void Assert.Equal<int>(int expected, int actual) (+ 8 sobrecargas)
Verifies that two objects are equal, using a default comparer.

Excepciones:
Xunit.Sdk.EqualException



```
[Fact]
0 referencias
public void TestSum2And3()
{
    Assert.Equal(Sum(2, 3), 5);
}

1 referencia
private stati
{
    return a
}
```

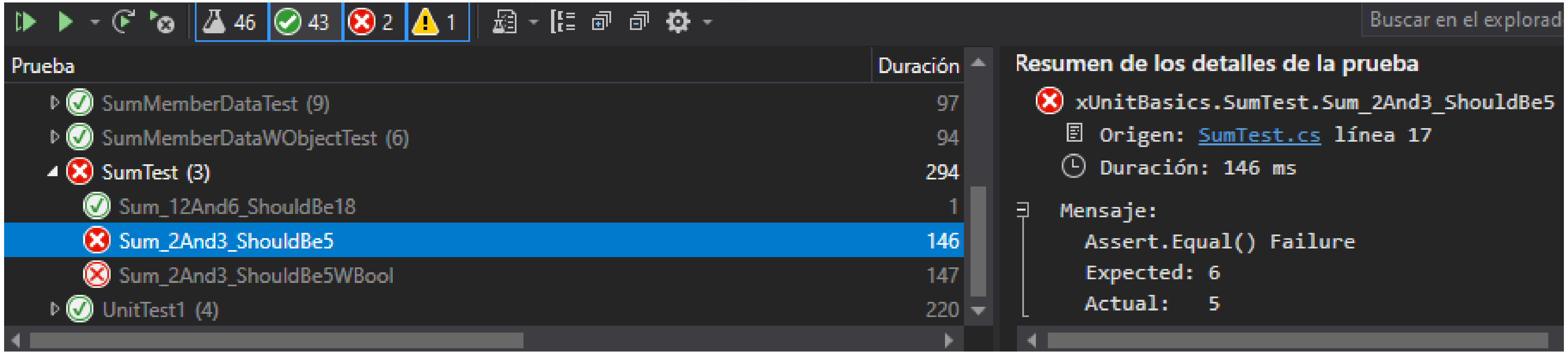
void Assert.Equal<int>(int expected, int actual) (+ 8 sobrecargas)
Verifies that two objects are equal, using a default comparer.

Excepciones:
Xunit.Sdk.EqualException

xUnit2000: The literal or constant value 5 should be passed as the 'expected' argument in the call to 'Assert.Equal(expected, actual)'

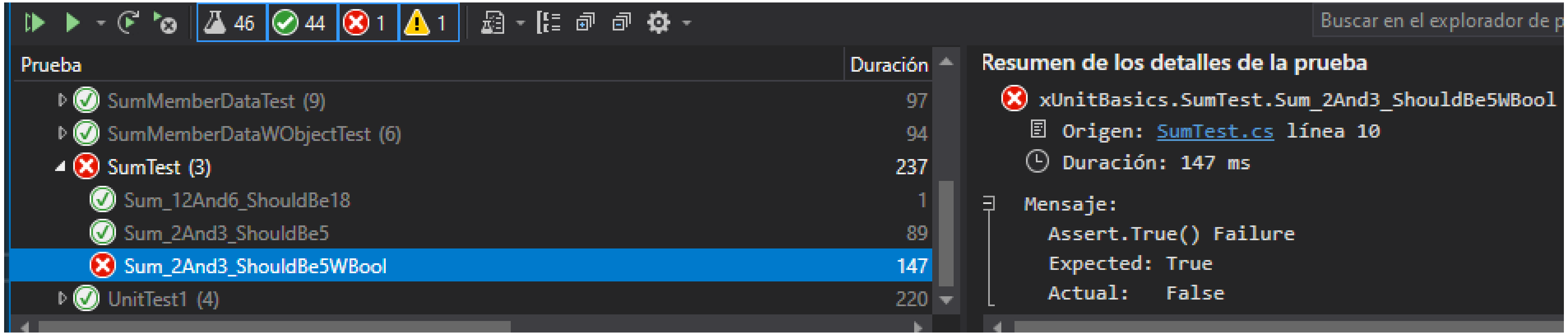
xUnit 101

How to test



```
Prueba                                   Duración
▷ ✓ SumMemberDataTest (9)                      97
▷ ✓ SumMemberDataWObjectTest (6)               94
◢ ✗ SumTest (3)                               294
    ✓ Sum_12And6_ShouldBe18                     1
    ✗ Sum_2And3_ShouldBe5                     146
    ✗ Sum_2And3_ShouldBe5WBool                147
▷ ✓ UnitTest1 (4)                             220
```

```
Resumen de los detalles de la prueba
 ✗ xUnitBasics.SumTest.Sum_2And3_ShouldBe5
    Origen: SumTest.cs línea 17
    Duración: 146 ms

  Mensaje:
    Assert.Equal() Failure
    Expected: 6
    Actual:   5
```



```
Prueba                                   Duración
▷ ✓ SumMemberDataTest (9)                      97
▷ ✓ SumMemberDataWObjectTest (6)               94
◢ ✗ SumTest (3)                               237
    ✓ Sum_12And6_ShouldBe18                     1
    ✓ Sum_2And3_ShouldBe5                      89
    ✗ Sum_2And3_ShouldBe5WBool                147
▷ ✓ UnitTest1 (4)                             220
```

```
Resumen de los detalles de la prueba
 ✗ xUnitBasics.SumTest.Sum_2And3_ShouldBe5WBool
    Origen: SumTest.cs línea 10
    Duración: 147 ms

  Mensaje:
    Assert.True() Failure
    Expected: True
    Actual:   False
```

xUnit 101

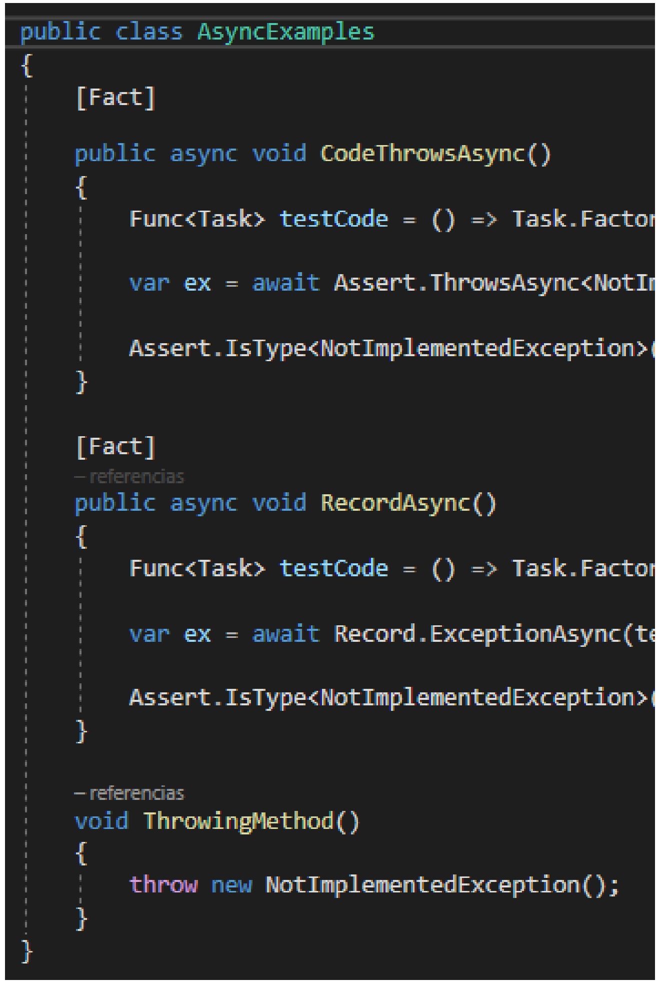
Methods to test

Assert.True	Assert.False
Assert.Equal	Assert.NotEqual
Assert.Contains	Assert.NotContains
Assert.Null	Assert.NotNull
Assert.Same	Assert.NotSame
Assert.StrictEqual	Assert.NotStrictEqual
Assert.InRange	Assert.NotInRange
Assert.Empty	Assert.NotEmpty
Assert.IsType	Assert.IsNotType
Assert.StartsWith	Assert.EndsWith
Assert.Matches	Assert.NotMatches

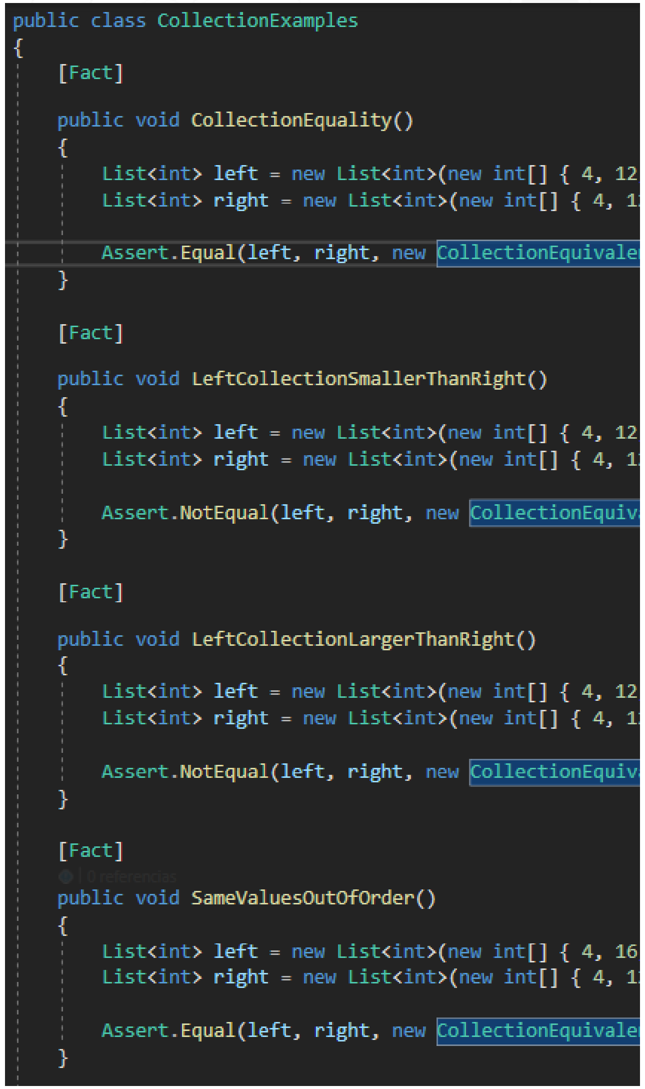
Assert.All
Assert.Collection
Assert.Equals
Assert.IsAssignableFrom
Assert.Throws
Assert.Subset
Assert.Superset
…

xUnit 101

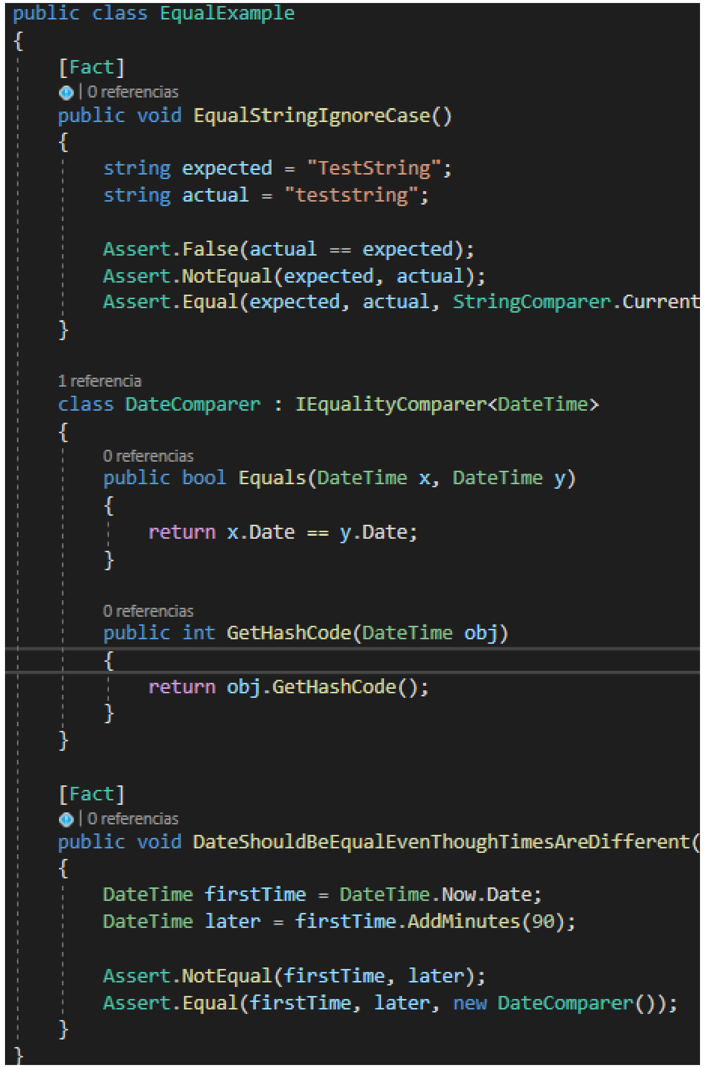
Methods to test



```
public class AsyncExamples
{
    [Fact]
    public async void CodeThrowsAsync()
    {
        Func<Task> testCode = () => Task.Factor

        var ex = await Assert.ThrowsAsync<NotIm

        Assert.IsType<NotImplementedException>(
    }

    [Fact]
    – referencias
    public async void RecordAsync()
    {
        Func<Task> testCode = () => Task.Factor

        var ex = await Record.ExceptionAsync(te

        Assert.IsType<NotImplementedException>(
    }

    – referencias
    void ThrowingMethod()
    {
        throw new NotImplementedException();
    }
}
```



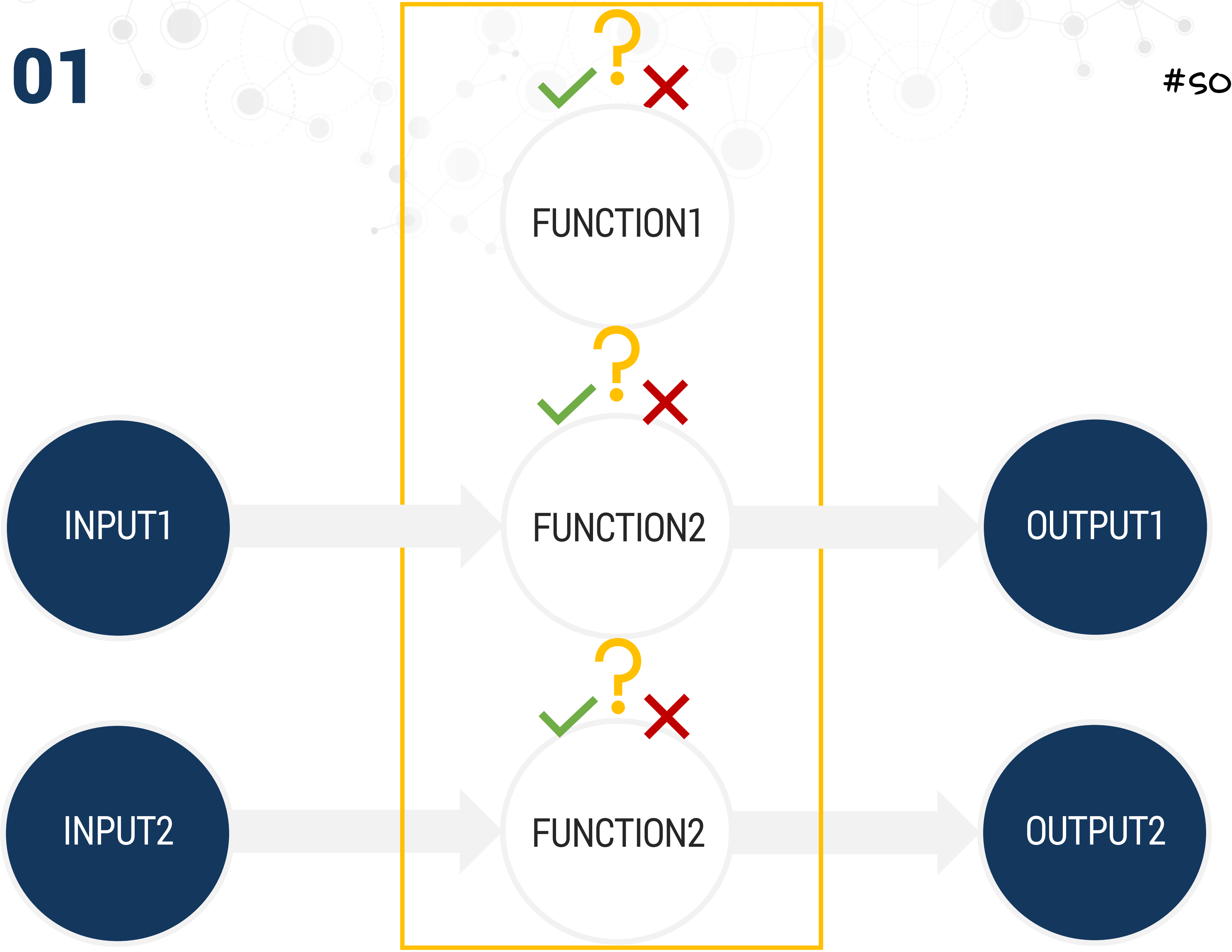
```
public class CollectionExamples
{
    [Fact]
    public void CollectionEquality()
    {
        List<int> left = new List<int>(new int[] { 4, 12
        List<int> right = new List<int>(new int[] { 4, 1

        Assert.Equal(left, right, new CollectionEquivale
    }

    [Fact]
    public void LeftCollectionSmallerThanRight()
    {
        List<int> left = new List<int>(new int[] { 4, 12
        List<int> right = new List<int>(new int[] { 4, 1

        Assert.NotEqual(left, right, new CollectionEquiv
    }

    [Fact]
    public void LeftCollectionLargerThanRight()
    {
        List<int> left = new List<int>(new int[] { 4, 12
        List<int> right = new List<int>(new int[] { 4, 1

        Assert.NotEqual(left, right, new CollectionEquiv
    }

    [Fact]
    | 0 referencias
    public void SameValuesOutOfOrder()
    {
        List<int> left = new List<int>(new int[] { 4, 16
        List<int> right = new List<int>(new int[] { 4, 1

        Assert.Equal(left, right, new CollectionEquivale
    }
```



```
public class EqualExample
{
    [Fact]
    | 0 referencias
    public void EqualStringIgnoreCase()
    {
        string expected = "TestString";
        string actual = "teststring";

        Assert.False(actual == expected);
        Assert.NotEqual(expected, actual);
        Assert.Equal(expected, actual, StringComparer.Current
    }

    1 referencia
    class DateComparer : IEqualityComparer<DateTime>
    {
        0 referencias
        public bool Equals(DateTime x, DateTime y)
        {
            return x.Date == y.Date;
        }

        0 referencias
        public int GetHashCode(DateTime obj)
        {
            return obj.GetHashCode();
        }
    }

    [Fact]
    | 0 referencias
    public void DateShouldBeEqualEvenThoughTimesAreDifferent(
    {
        DateTime firstTime = DateTime.Now.Date;
        DateTime later = firstTime.AddMinutes(90);

        Assert.NotEqual(firstTime, later);
        Assert.Equal(firstTime, later, new DateComparer());
    }
}
```

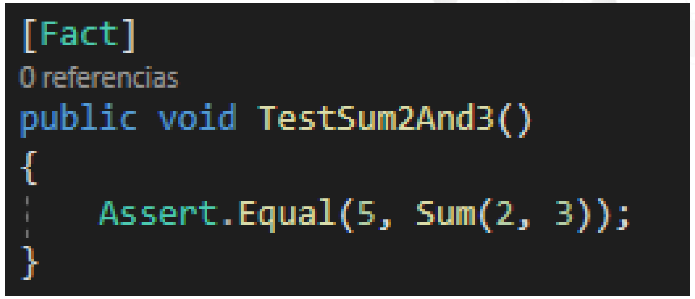
xUnit 101

How to test

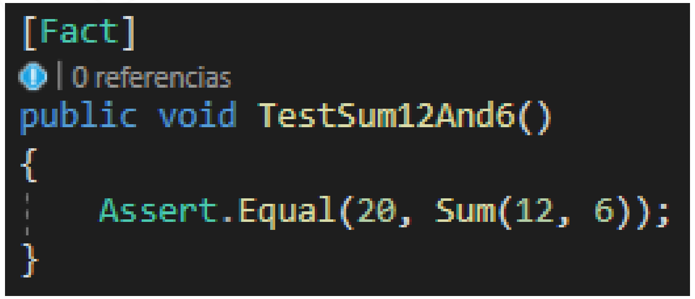


xUnit 101

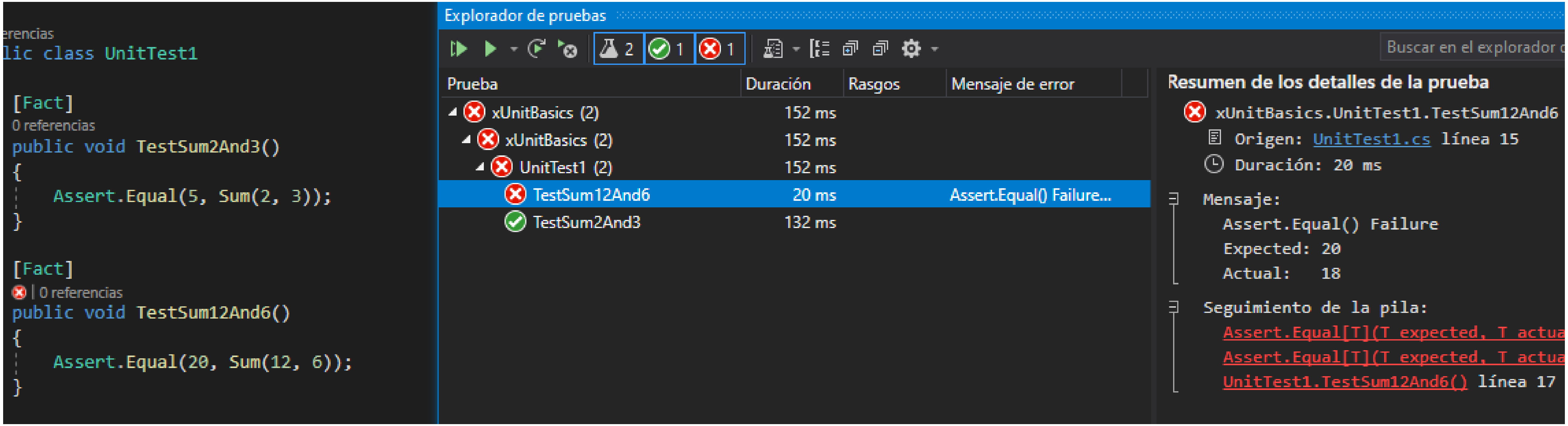
How to test



```
[Fact]
0 referencias
public void TestSum2And3()
{
    Assert.Equal(5, Sum(2, 3));
}
```



```
[Fact]
● | 0 referencias
public void TestSum12And6()
{
    Assert.Equal(20, Sum(12, 6));
}
```



```
erencias
lic class UnitTest1

    [Fact]
    0 referencias
    public void TestSum2And3()
    {
        Assert.Equal(5, Sum(2, 3));
    }

    [Fact]
    ✗ | 0 referencias
    public void TestSum12And6()
    {
        Assert.Equal(20, Sum(12, 6));
    }
```

Explorador de pruebas

Prueba	Duración	Rasgos	Mensaje de error
◢ ✗ xUnitBasics (2)	152 ms		
◢ ✗ xUnitBasics (2)	152 ms		
◢ ✗ UnitTest1 (2)	152 ms		
✗ TestSum12And6	20 ms		Assert.Equal() Failure...
✓ TestSum2And3	132 ms		

Resumen de los detalles de la prueba

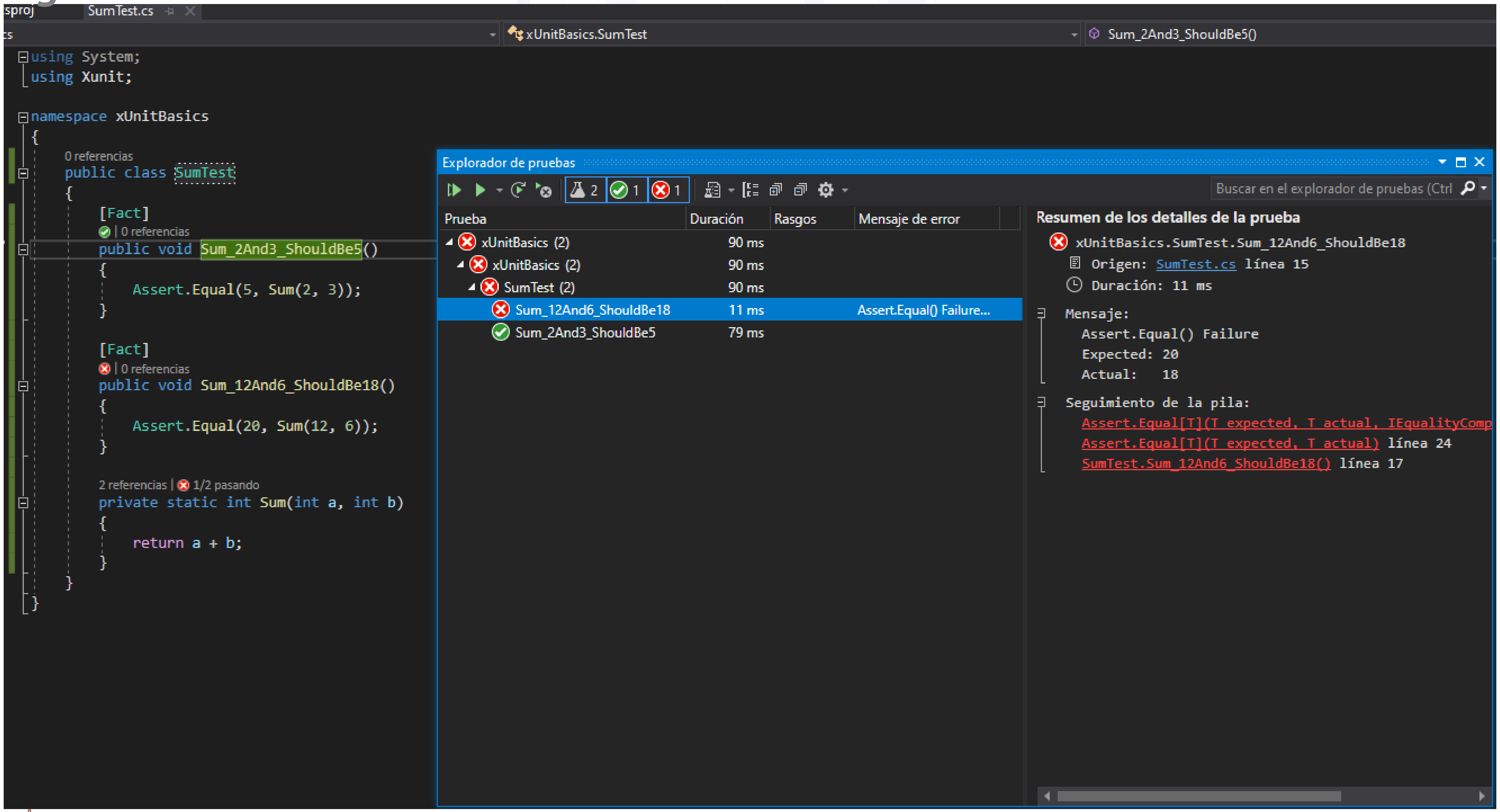
```
✗ xUnitBasics.UnitTest1.TestSum12And6
   Origen: UnitTest1.cs línea 15
   Duración: 20 ms

Mensaje:
  Assert.Equal() Failure
  Expected: 20
  Actual:   18

Seguimiento de la pila:
  Assert.Equal[T](T expected, T actua
  Assert.Equal[T](T expected, T actua
  UnitTest1.TestSum12And6() línea 17
```

xUnit 101

Naming conventions

#somoshiberus

https://docs.microsoft.com/en-us/dotnet/core/testing/unit-testing-best-practices



```
using System;
using Xunit;

namespace xUnitBasics
{
    0 referencias
    public class SumTest
    {
        [Fact]
        ✓ | 0 referencias
        public void Sum_2And3_ShouldBe5()
        {
            Assert.Equal(5, Sum(2, 3));
        }

        [Fact]
        ⊗ | 0 referencias
        public void Sum_12And6_ShouldBe18()
        {
            Assert.Equal(20, Sum(12, 6));
        }

        2 referencias | ⊗ 1/2 pasando
        private static int Sum(int a, int b)
        {
            return a + b;
        }
    }
}
```

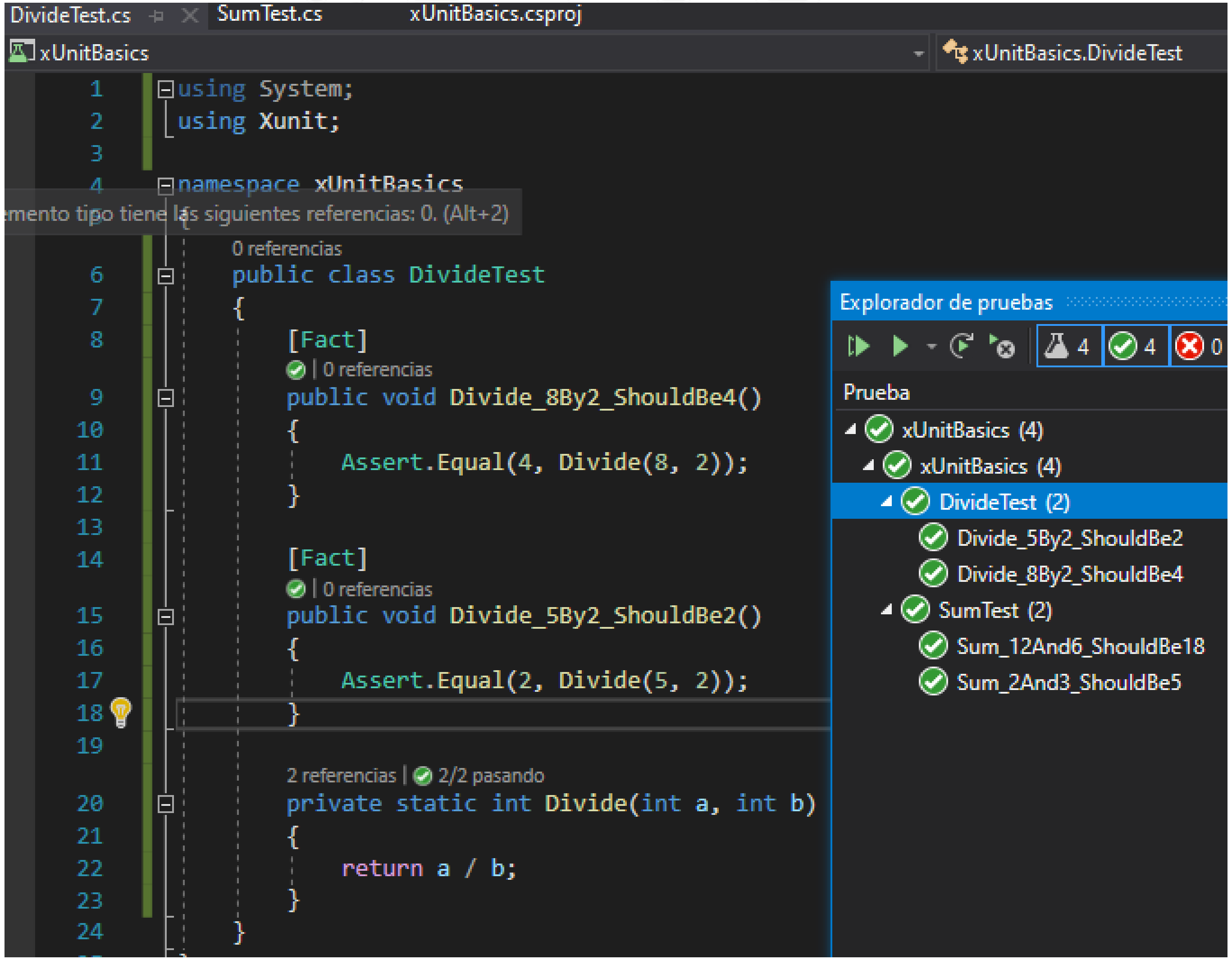
Prueba	Duración	Rasgos	Mensaje de error
⊗ xUnitBasics (2)	90 ms		
⊗ xUnitBasics (2)	90 ms		
⊗ SumTest (2)	90 ms		
⊗ Sum_12And6_ShouldBe18	11 ms		Assert.Equal() Failure...
✓ Sum_2And3_ShouldBe5	79 ms		

Resumen de los detalles de la prueba

```
⊗ xUnitBasics.SumTest.Sum_12And6_ShouldBe18
  Origen: SumTest.cs línea 15
  Duración: 11 ms

Mensaje:
  Assert.Equal() Failure
  Expected: 20
  Actual:   18

Seguimiento de la pila:
  Assert.Equal[T](T expected, T actual, IEqualityComp
  Assert.Equal[T](T expected, T actual) línea 24
  SumTest.Sum_12And6_ShouldBe18() línea 17
```

xUnit 101

Naming conventions



mento tipo tiene las siguientes referencias: 0. (Alt+2)

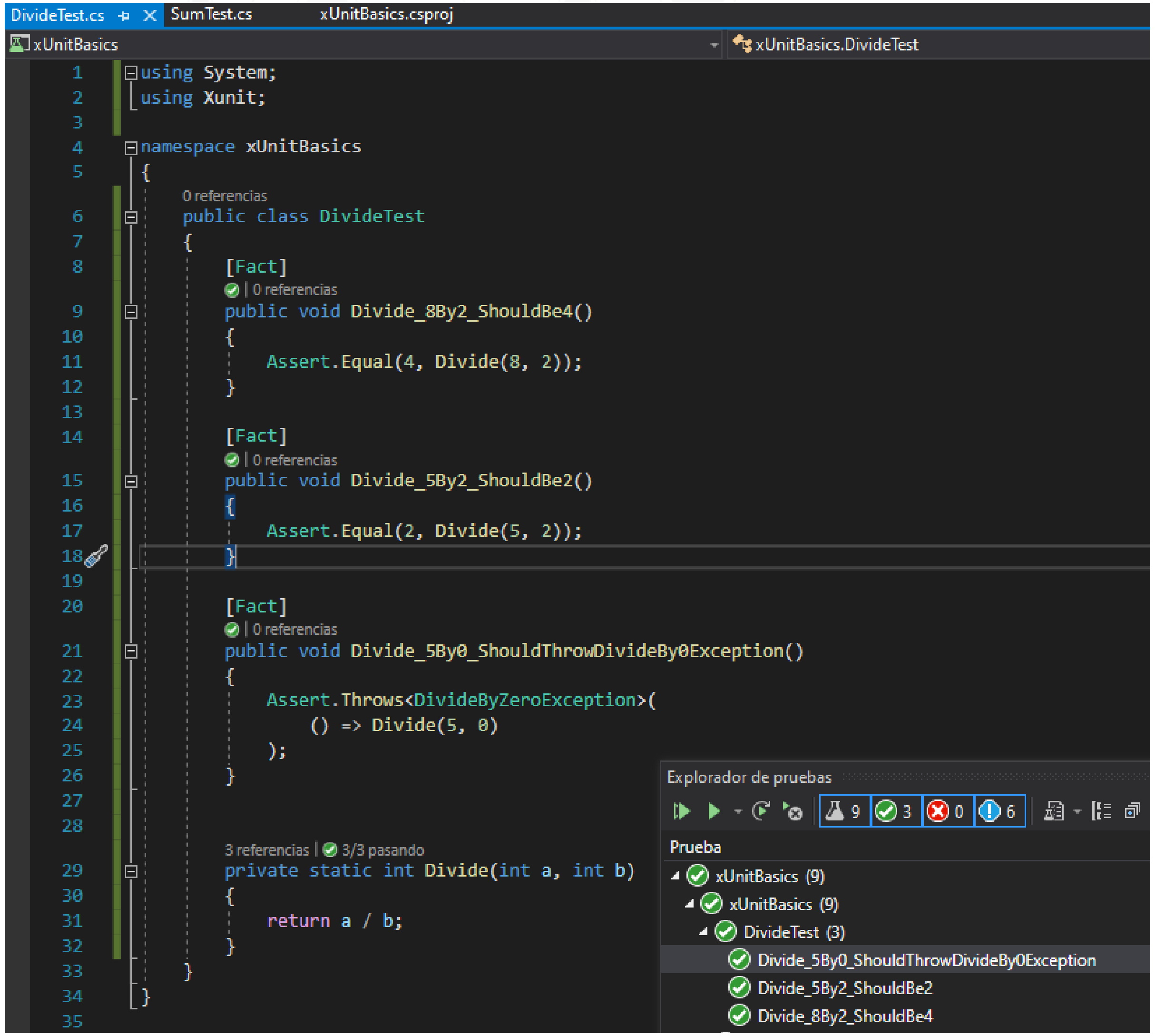
```
using System;
using Xunit;

namespace xUnitBasics
{
    0 referencias
    public class DivideTest
    {
        [Fact]
        ✓ | 0 referencias
        public void Divide_8By2_ShouldBe4()
        {
            Assert.Equal(4, Divide(8, 2));
        }

        [Fact]
        ✓ | 0 referencias
        public void Divide_5By2_ShouldBe2()
        {
            Assert.Equal(2, Divide(5, 2));
        }

        2 referencias | ✓ 2/2 pasando
        private static int Divide(int a, int b)
        {
            return a / b;
        }
    }
```

Prueba

```
◢ ✓ xUnitBasics (4)
  ◢ ✓ xUnitBasics (4)
    ◢ ✓ DivideTest (2)
        ✓ Divide_5By2_ShouldBe2
        ✓ Divide_8By2_ShouldBe4
    ◢ ✓ SumTest (2)
        ✓ Sum_12And6_ShouldBe18
        ✓ Sum_2And3_ShouldBe5
```

xUnit 101

Success when exception



```
using System;
using Xunit;

namespace xUnitBasics
{
    0 referencias
    public class DivideTest
    {
        [Fact]
        | 0 referencias
        public void Divide_8By2_ShouldBe4()
        {
            Assert.Equal(4, Divide(8, 2));
        }

        [Fact]
        | 0 referencias
        public void Divide_5By2_ShouldBe2()
        {
            Assert.Equal(2, Divide(5, 2));
        }


        [Fact]
        | 0 referencias
        public void Divide_5By0_ShouldThrowDivideBy0Exception()
        {
            Assert.Throws<DivideByZeroException>(
                () => Divide(5, 0)
            );
        }


        3 referencias | 3/3 pasando
        private static int Divide(int a, int b)
        {
            return a / b;
        }
    }
}
```

Explorador de pruebas

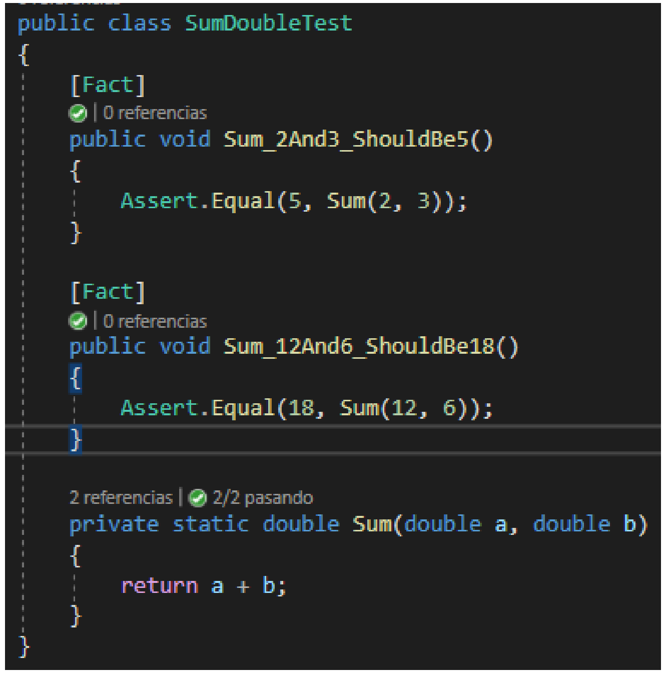
9 ✓ 3 ✗ 0 ! 6

Prueba

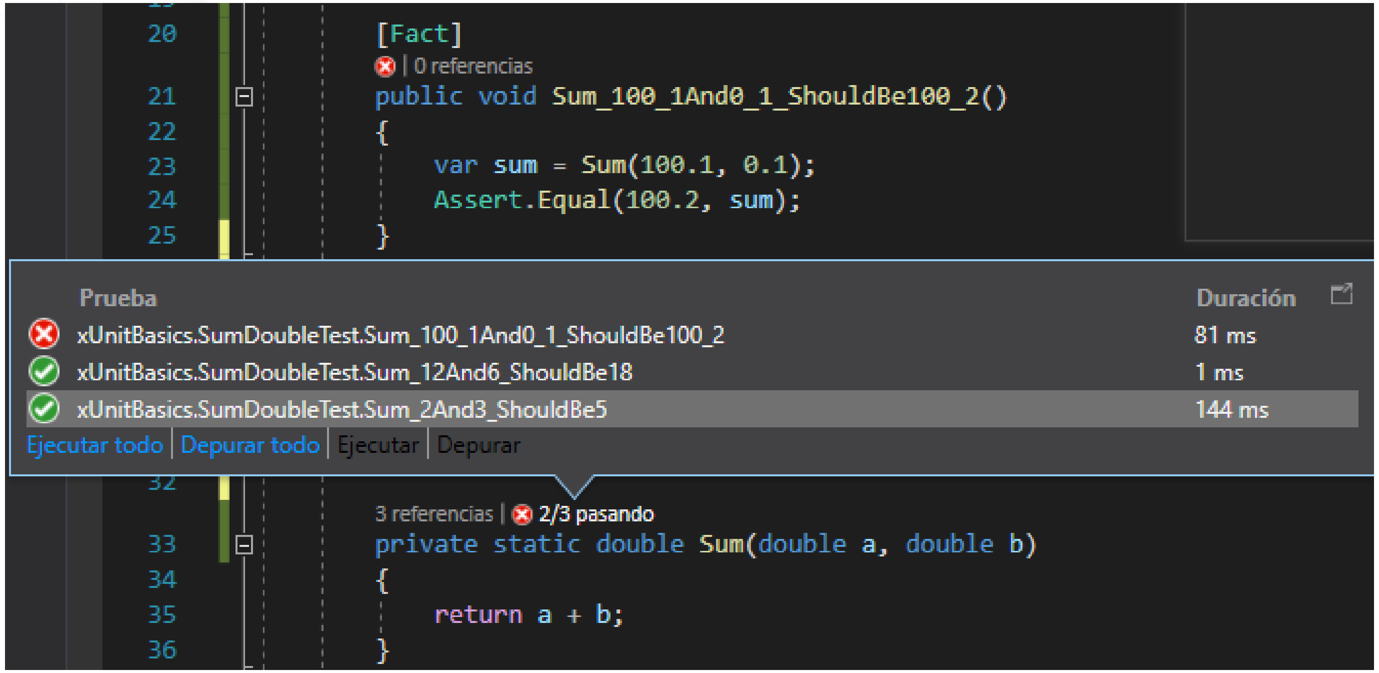
- ✓ xUnitBasics (9)
  - ✓ xUnitBasics (9)
    - ✓ DivideTest (3)
      - ✓ Divide_5By0_ShouldThrowDivideBy0Exception
      - ✓ Divide_5By2_ShouldBe2
      - ✓ Divide_8By2_ShouldBe4

xUnit 101

Double Trouble



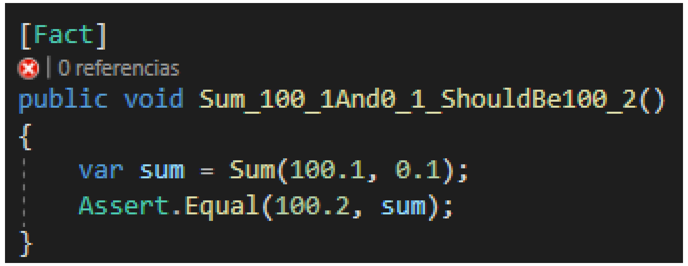
```
public class SumDoubleTest
{
    [Fact]
    ✓ | 0 referencias
    public void Sum_2And3_ShouldBe5()
    {
        Assert.Equal(5, Sum(2, 3));
    }

    [Fact]
    ✓ | 0 referencias
    public void Sum_12And6_ShouldBe18()
    {
        Assert.Equal(18, Sum(12, 6));
    }

    2 referencias | ✓ 2/2 pasando
    private static double Sum(double a, double b)
    {
        return a + b;
    }
}
```



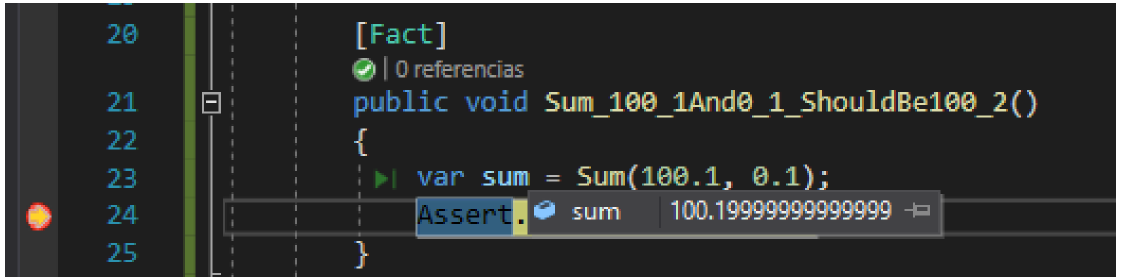
```
    [Fact]
    ✗ | 0 referencias
    public void Sum_100_1And0_1_ShouldBe100_2()
    {
        var sum = Sum(100.1, 0.1);
        Assert.Equal(100.2, sum);
    }
```

Prueba	Duración
✗ xUnitBasics.SumDoubleTest.Sum_100_1And0_1_ShouldBe100_2	81 ms
✓ xUnitBasics.SumDoubleTest.Sum_12And6_ShouldBe18	1 ms
✓ xUnitBasics.SumDoubleTest.Sum_2And3_ShouldBe5	144 ms

```
    3 referencias | ✗ 2/3 pasando
    private static double Sum(double a, double b)
    {
        return a + b;
    }
```



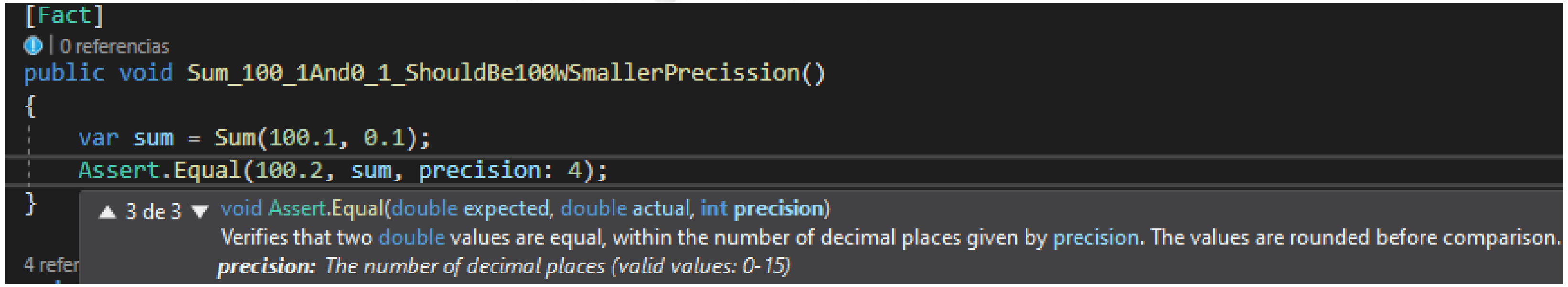
```
[Fact]
✗ | 0 referencias
public void Sum_100_1And0_1_ShouldBe100_2()
{
    var sum = Sum(100.1, 0.1);
    Assert.Equal(100.2, sum);
}
```



```
    [Fact]
    ✓ | 0 referencias
    public void Sum_100_1And0_1_ShouldBe100_2()
    {
        var sum = Sum(100.1, 0.1);
        Assert.   sum   100.19999999999999
    }
```

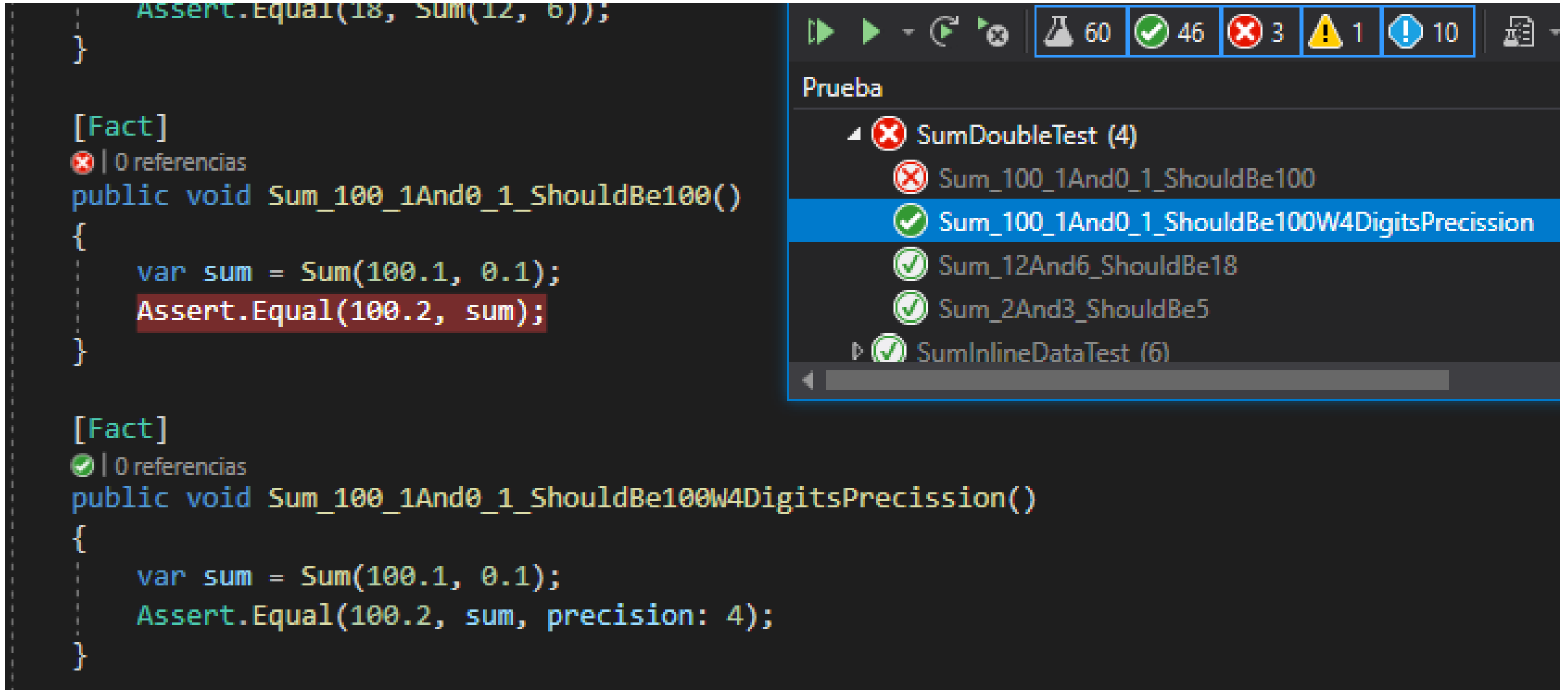
xUnit 101

Double Trouble



```
[Fact]
| 0 referencias
public void Sum_100_1And0_1_ShouldBe100WSmallerPrecission()
{
    var sum = Sum(100.1, 0.1);
    Assert.Equal(100.2, sum, precision: 4);
}
4 refer
```

▲ 3 de 3 ▼ void Assert.Equal(double expected, double actual, int precision)
Verifies that two double values are equal, within the number of decimal places given by precision. The values are rounded before comparison.
precision: The number of decimal places (valid values: 0-15)

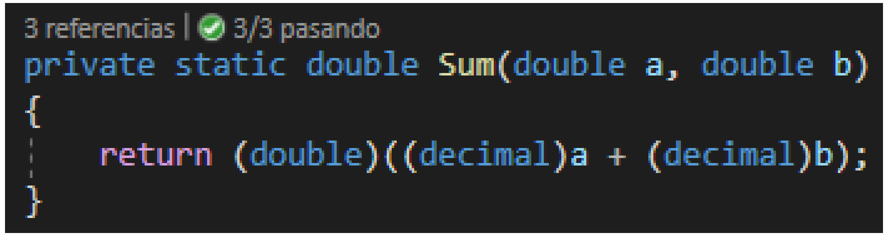


```
    Assert.Equal(18, Sum(12, 6));
}

[Fact]
| 0 referencias
public void Sum_100_1And0_1_ShouldBe100()
{
    var sum = Sum(100.1, 0.1);
    Assert.Equal(100.2, sum);
}

[Fact]
| 0 referencias
public void Sum_100_1And0_1_ShouldBe100W4DigitsPrecission()
{
    var sum = Sum(100.1, 0.1);
    Assert.Equal(100.2, sum, precision: 4);
}
```

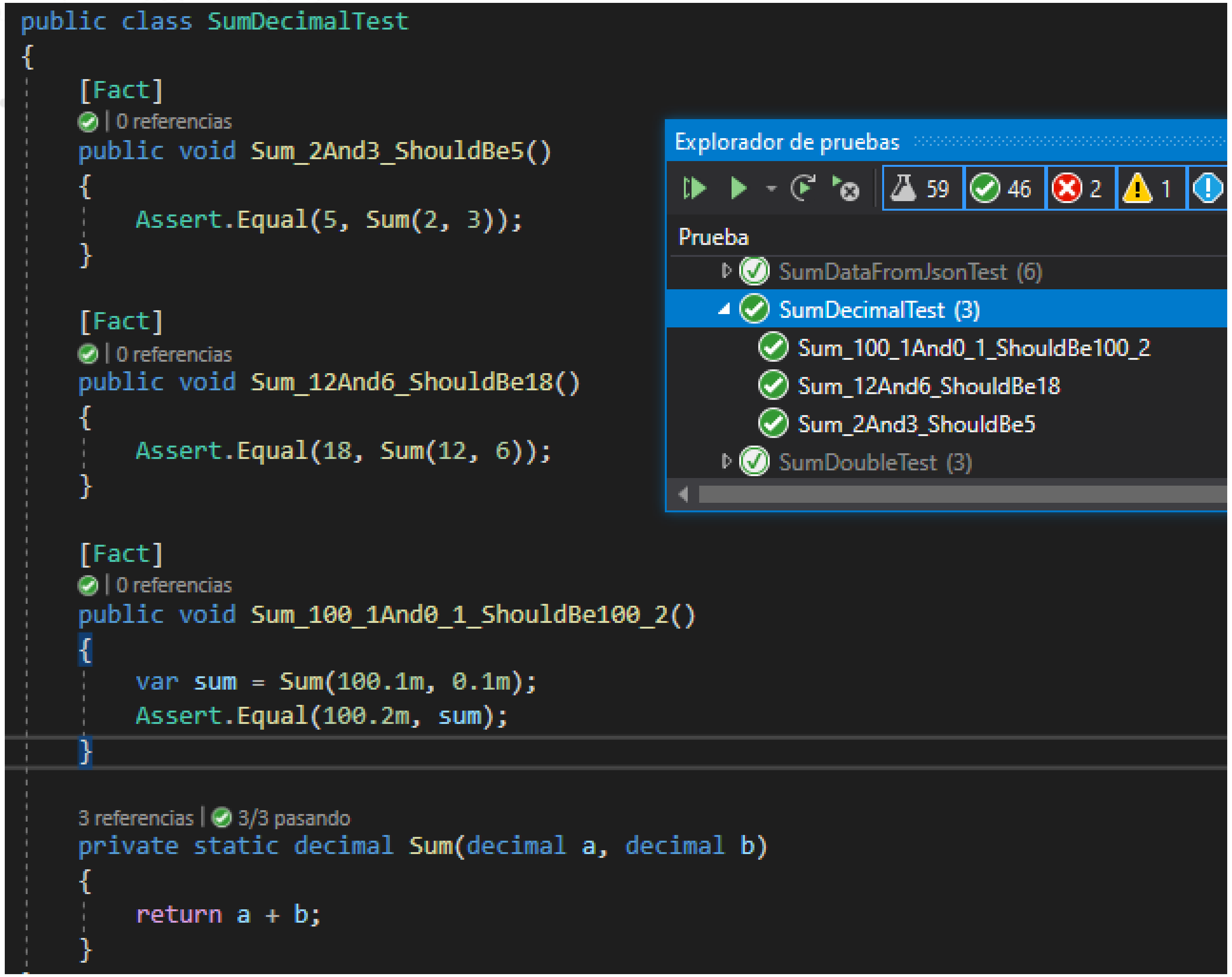
Prueba
- ✗ SumDoubleTest (4)
  - ✗ Sum_100_1And0_1_ShouldBe100
  - ✓ Sum_100_1And0_1_ShouldBe100W4DigitsPrecission
  - ✓ Sum_12And6_ShouldBe18
  - ✓ Sum_2And3_ShouldBe5
- ✓ SumInlineDataTest (6)

xUnit 101

Double Trouble



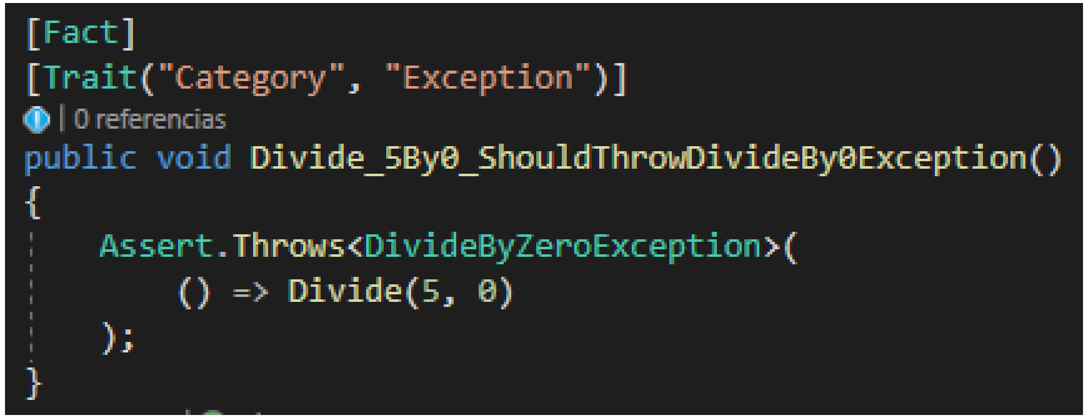
```
3 referencias | ✓ 3/3 pasando
private static double Sum(double a, double b)
{
    return (double)((decimal)a + (decimal)b);
}
```



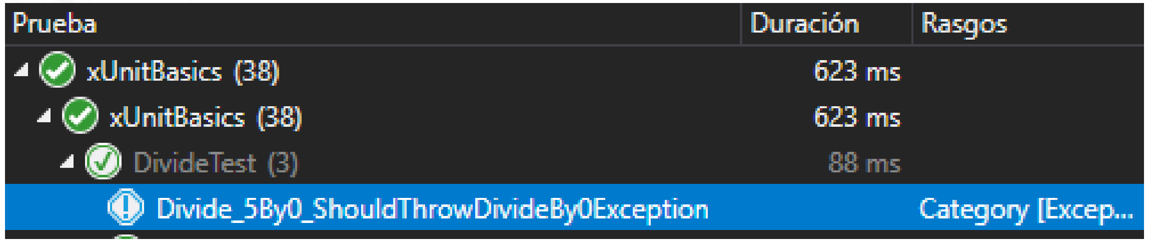
```
public class SumDecimalTest
{
    [Fact]
    ✓ | 0 referencias
    public void Sum_2And3_ShouldBe5()
    {
        Assert.Equal(5, Sum(2, 3));
    }

    [Fact]
    ✓ | 0 referencias
    public void Sum_12And6_ShouldBe18()
    {
        Assert.Equal(18, Sum(12, 6));
    }

    [Fact]
    ✓ | 0 referencias
    public void Sum_100_1And0_1_ShouldBe100_2()
    {
        var sum = Sum(100.1m, 0.1m);
        Assert.Equal(100.2m, sum);
    }

    3 referencias | ✓ 3/3 pasando
    private static decimal Sum(decimal a, decimal b)
    {
        return a + b;
    }
```

xUnit 101

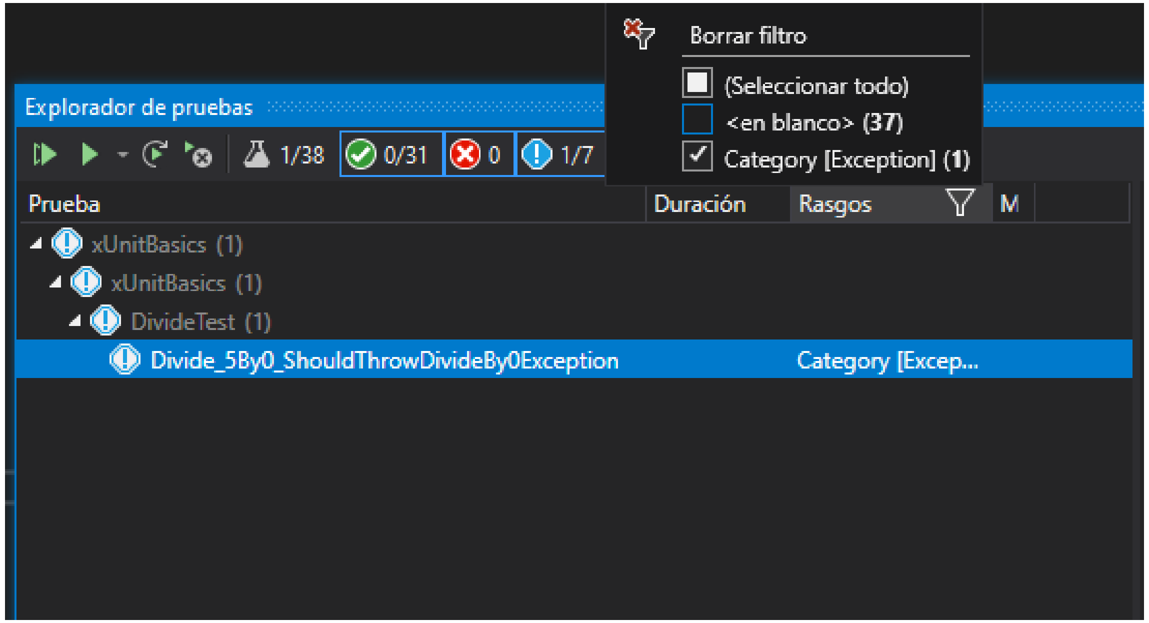
Traits - Categories



```
[Fact]
[Trait("Category", "Exception")]
0 referencias
public void Divide_5By0_ShouldThrowDivideBy0Exception()
{
    Assert.Throws<DivideByZeroException>(
        () => Divide(5, 0)
    );
}
```



Prueba	Duración	Rasgos
xUnitBasics (38)	623 ms	
xUnitBasics (38)	623 ms	
DivideTest (3)	88 ms	
Divide_5By0_ShouldThrowDivideBy0Exception		Category [Excep...



Explorador de pruebas

Borrar filtro

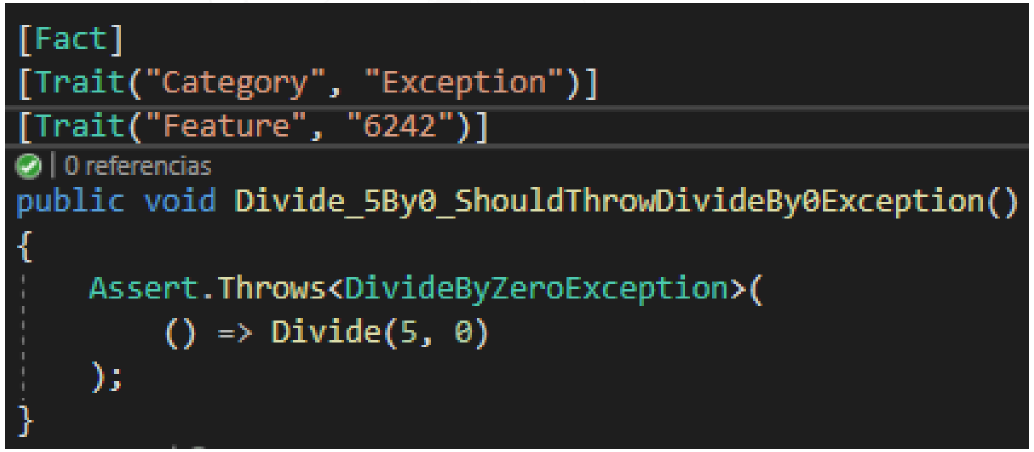
- □ (Seleccionar todo)
- □ <en blanco> (37)
- ☑ Category [Exception] (1)

Prueba	Duración	Rasgos	M
xUnitBasics (1)			
xUnitBasics (1)			
DivideTest (1)			
Divide_5By0_ShouldThrowDivideBy0Exception		Category [Excep...	

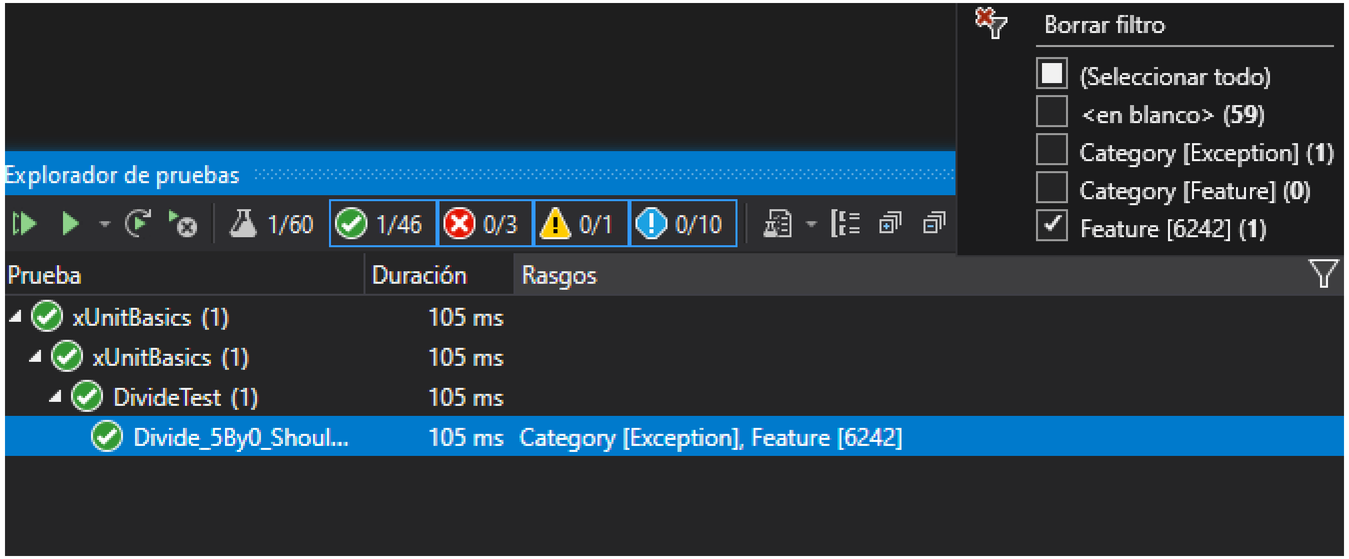
xUnit 101

Traits - Custom Properties

#somoshiberus



```
[Fact]
[Trait("Category", "Exception")]
[Trait("Feature", "6242")]
✓ | 0 referencias
public void Divide_5By0_ShouldThrowDivideBy0Exception()
{
    Assert.Throws<DivideByZeroException>(
        () => Divide(5, 0)
    );
}
```



Borrar filtro

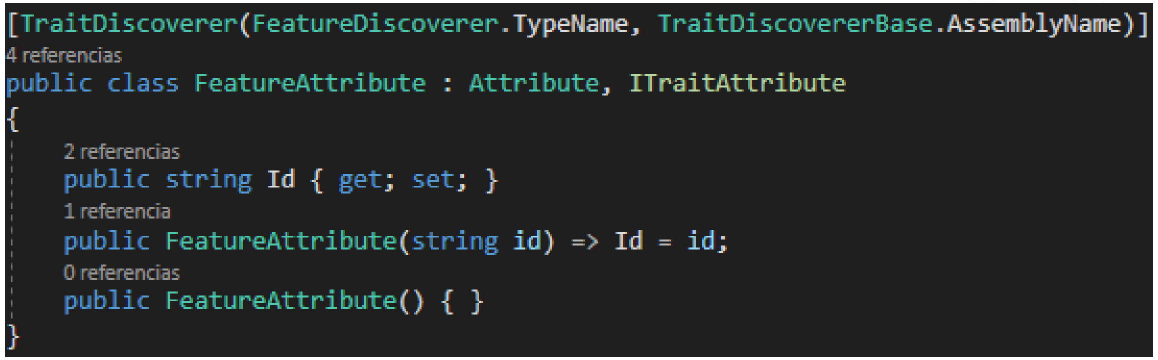
- ☐ (Seleccionar todo)
- ☐ <en blanco> (59)
- ☐ Category [Exception] (1)
- ☐ Category [Feature] (0)
- ☑ Feature [6242] (1)

Prueba	Duración	Rasgos
xUnitBasics (1)	105 ms	
xUnitBasics (1)	105 ms	
DivideTest (1)	105 ms	
Divide_5By0_Shoul...	105 ms	Category [Exception], Feature [6242]

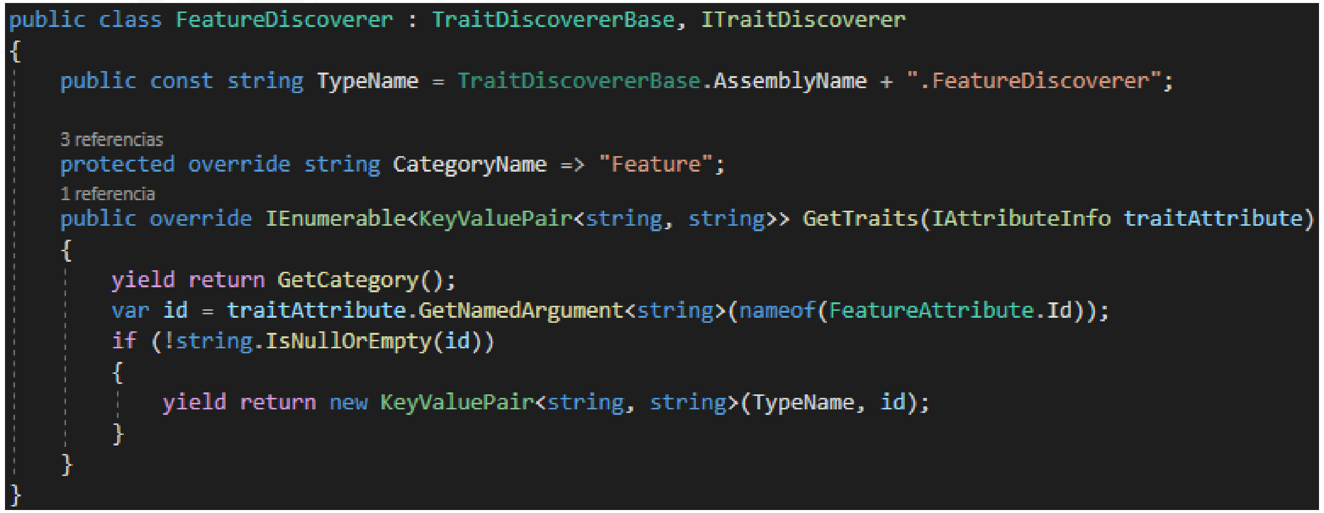
xUnit 101

Traits - Custom Properties

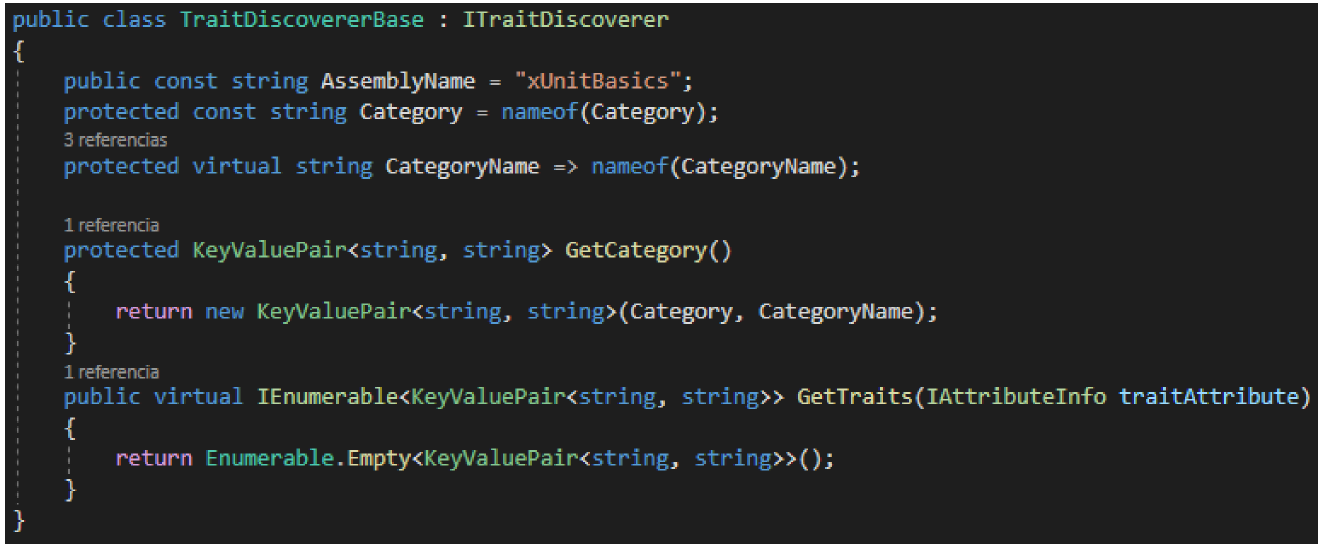
#somoshiberus



```
[TraitDiscoverer(FeatureDiscoverer.TypeName, TraitDiscovererBase.AssemblyName)]
4 referencias
public class FeatureAttribute : Attribute, ITraitAttribute
{
    2 referencias
    public string Id { get; set; }
    1 referencia
    public FeatureAttribute(string id) => Id = id;
    0 referencias
    public FeatureAttribute() { }
}
```



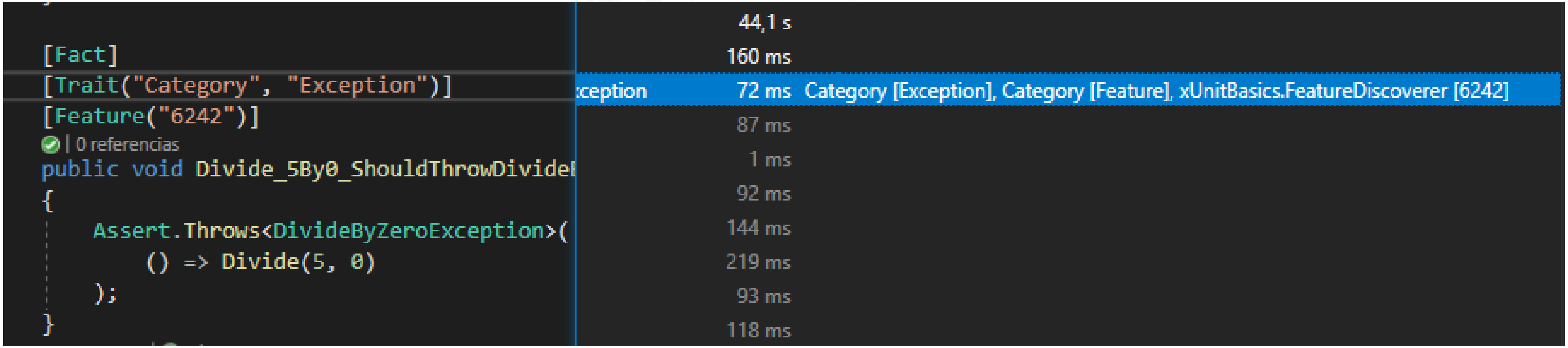
```
public class FeatureDiscoverer : TraitDiscovererBase, ITraitDiscoverer
{
    public const string TypeName = TraitDiscovererBase.AssemblyName + ".FeatureDiscoverer";

    3 referencias
    protected override string CategoryName => "Feature";
    1 referencia
    public override IEnumerable<KeyValuePair<string, string>> GetTraits(IAttributeInfo traitAttribute)
    {
        yield return GetCategory();
        var id = traitAttribute.GetNamedArgument<string>(nameof(FeatureAttribute.Id));
        if (!string.IsNullOrEmpty(id))
        {
            yield return new KeyValuePair<string, string>(TypeName, id);
        }
    }
}
```



```
public class TraitDiscovererBase : ITraitDiscoverer
{
    public const string AssemblyName = "xUnitBasics";
    protected const string Category = nameof(Category);
    3 referencias
    protected virtual string CategoryName => nameof(CategoryName);

    1 referencia
    protected KeyValuePair<string, string> GetCategory()
    {
        return new KeyValuePair<string, string>(Category, CategoryName);
    }
    1 referencia
    public virtual IEnumerable<KeyValuePair<string, string>> GetTraits(IAttributeInfo traitAttribute)
    {
        return Enumerable.Empty<KeyValuePair<string, string>>();
    }
}
```

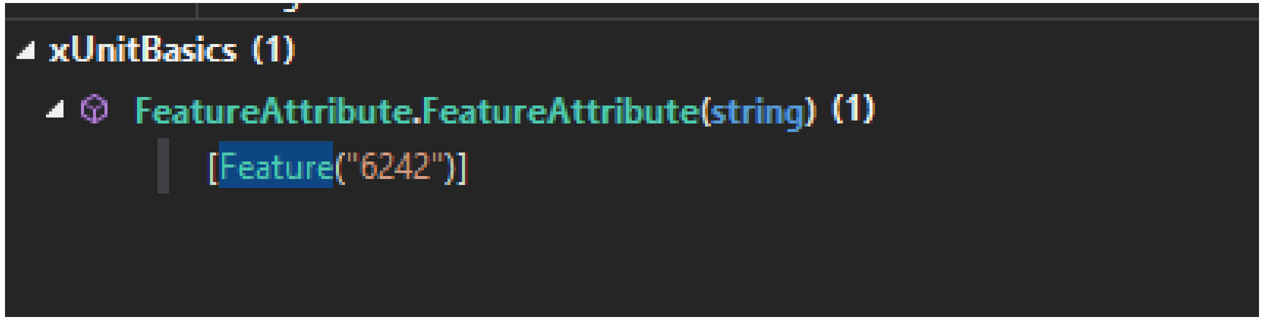
xUnit 101

Traits - Custom Properties



```
[Fact]
[Trait("Category", "Exception")]
[Feature("6242")]
✓ | 0 referencias
public void Divide_5By0_ShouldThrowDivideI
{
    Assert.Throws<DivideByZeroException>(
        () => Divide(5, 0)
    );
}
```

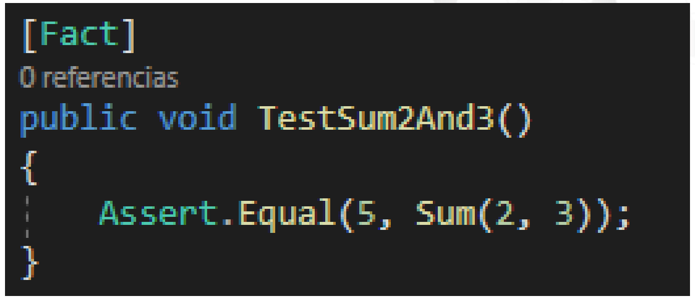
```
             44,1 s
             160 ms
xception      72 ms  Category [Exception], Category [Feature], xUnitBasics.FeatureDiscoverer [6242]
              87 ms
               1 ms
              92 ms
             144 ms
             219 ms
              93 ms
             118 ms
```



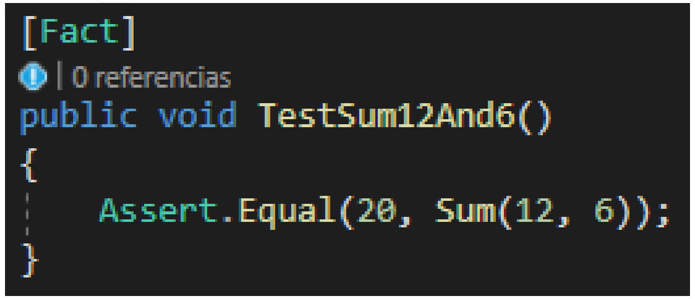
```
◢ xUnitBasics (1)
  ◢ FeatureAttribute.FeatureAttribute(string) (1)
        [Feature("6242")]
```

xUnit 101

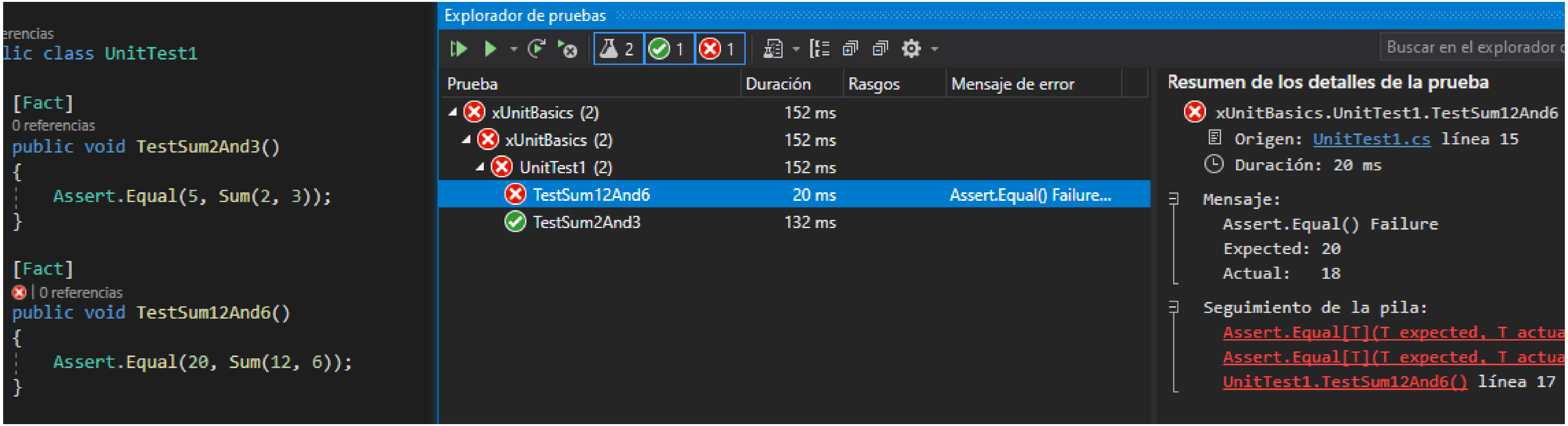
Theories - InlineData



```
[Fact]
0 referencias
public void TestSum2And3()
{
    Assert.Equal(5, Sum(2, 3));
}
```



```
[Fact]
| 0 referencias
public void TestSum12And6()
{
    Assert.Equal(20, Sum(12, 6));
}
```



```
erencias
lic class UnitTest1

    [Fact]
    0 referencias
    public void TestSum2And3()
    {
        Assert.Equal(5, Sum(2, 3));
    }

    [Fact]
    | 0 referencias
    public void TestSum12And6()
    {
        Assert.Equal(20, Sum(12, 6));
    }
```

Explorador de pruebas

Prueba	Duración	Rasgos	Mensaje de error
xUnitBasics (2)	152 ms		
xUnitBasics (2)	152 ms		
UnitTest1 (2)	152 ms		
TestSum12And6	20 ms		Assert.Equal() Failure...
TestSum2And3	132 ms		

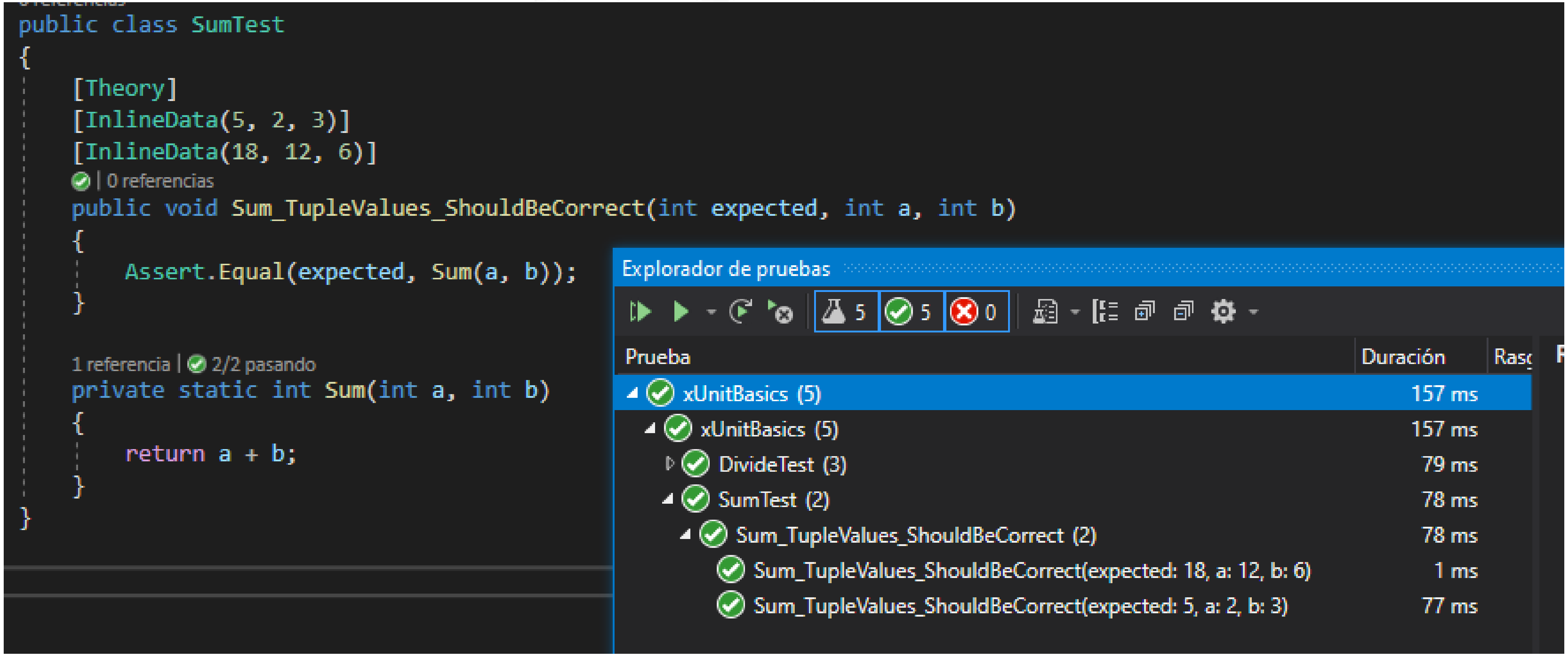
Resumen de los detalles de la prueba

```
xUnitBasics.UnitTest1.TestSum12And6
  Origen: UnitTest1.cs línea 15
  Duración: 20 ms

Mensaje:
  Assert.Equal() Failure
  Expected: 20
  Actual:   18

Seguimiento de la pila:
  Assert.Equal[T](T expected, T actua
  Assert.Equal[T](T expected, T actua
  UnitTest1.TestSum12And6() línea 17
```

Nafarroako Gobernua Gobierno de Navarra

xUnit 101

Theories - InlineData



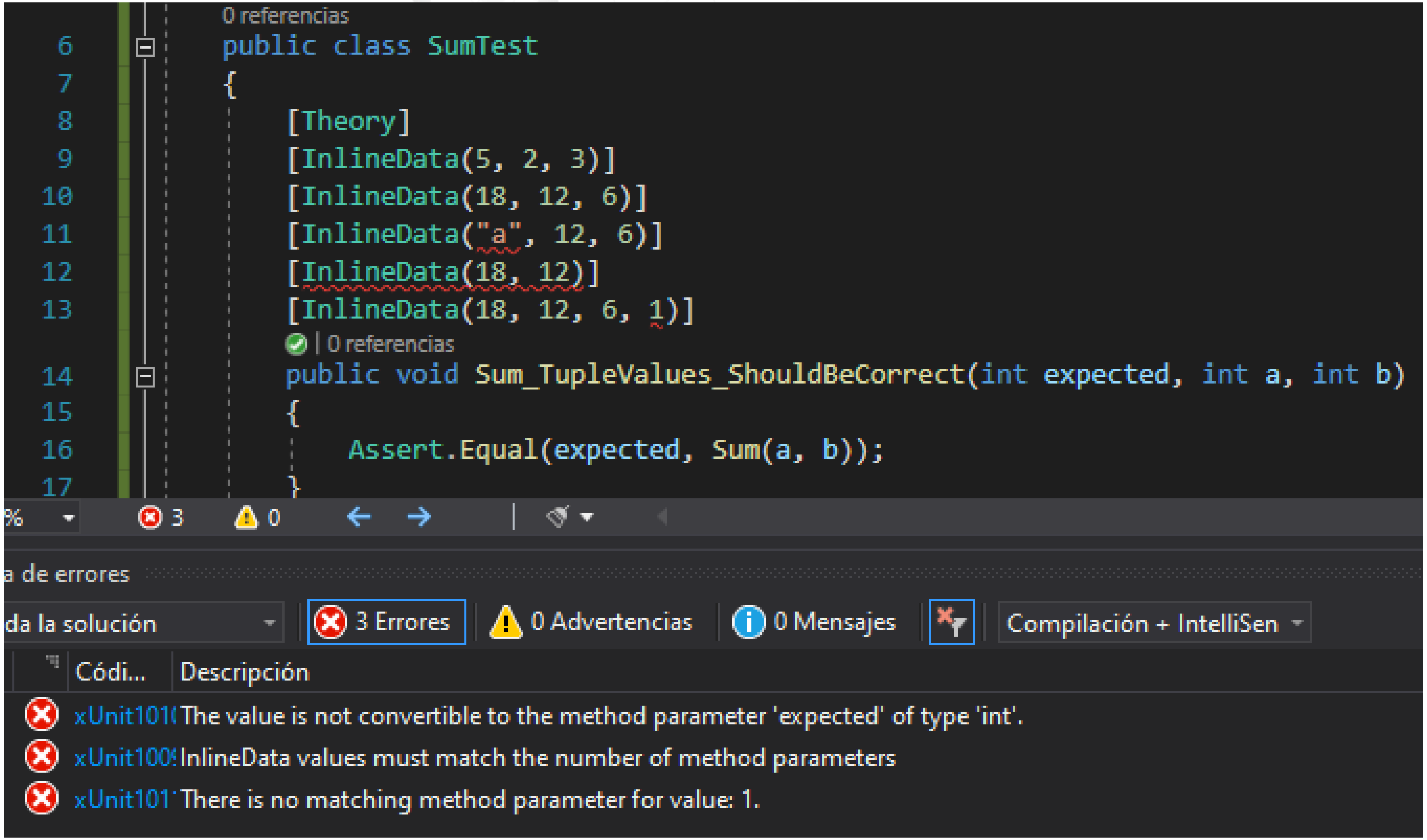
```
public class SumTest
{
    [Theory]
    [InlineData(5, 2, 3)]
    [InlineData(18, 12, 6)]
    ✓ | 0 referencias
    public void Sum_TupleValues_ShouldBeCorrect(int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }

    1 referencia | ✓ 2/2 pasando
    private static int Sum(int a, int b)
    {
        return a + b;
    }
}
```

Explorador de pruebas

```
Prueba                                                            Duración   Rasg
◢ ✓ xUnitBasics (5)                                                157 ms
  ◢ ✓ xUnitBasics (5)                                              157 ms
    ▷ ✓ DivideTest (3)                                              79 ms
    ◢ ✓ SumTest (2)                                                 78 ms
      ◢ ✓ Sum_TupleValues_ShouldBeCorrect (2)                       78 ms
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)   1 ms
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)    77 ms
```

xUnit 101

Theories - InlineData

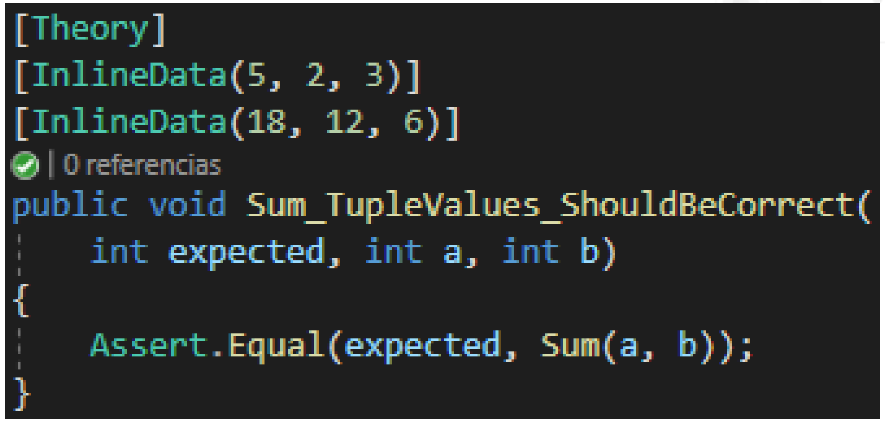


```
0 referencias
public class SumTest
{
    [Theory]
    [InlineData(5, 2, 3)]
    [InlineData(18, 12, 6)]
    [InlineData("a", 12, 6)]
    [InlineData(18, 12)]
    [InlineData(18, 12, 6, 1)]
    | 0 referencias
    public void Sum_TupleValues_ShouldBeCorrect(int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }
```

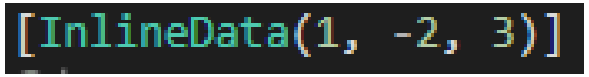
Códi...	Descripción
xUnit101(	The value is not convertible to the method parameter 'expected' of type 'int'.
xUnit100!	InlineData values must match the number of method parameters
xUnit101	There is no matching method parameter for value: 1.

xUnit 101

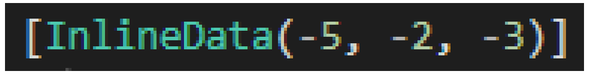
Theories - Boundary Value Analysis



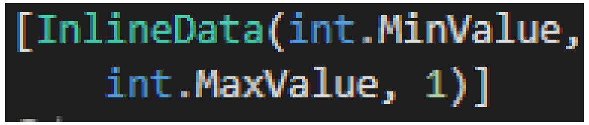
```
[Theory]
[InlineData(5, 2, 3)]
[InlineData(18, 12, 6)]
✓ | 0 referencias
public void Sum_TupleValues_ShouldBeCorrect(
    int expected, int a, int b)
{
    Assert.Equal(expected, Sum(a, b));
}
```



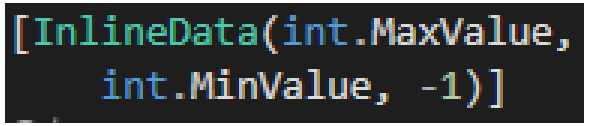
```
[InlineData(1, -2, 3)]
```



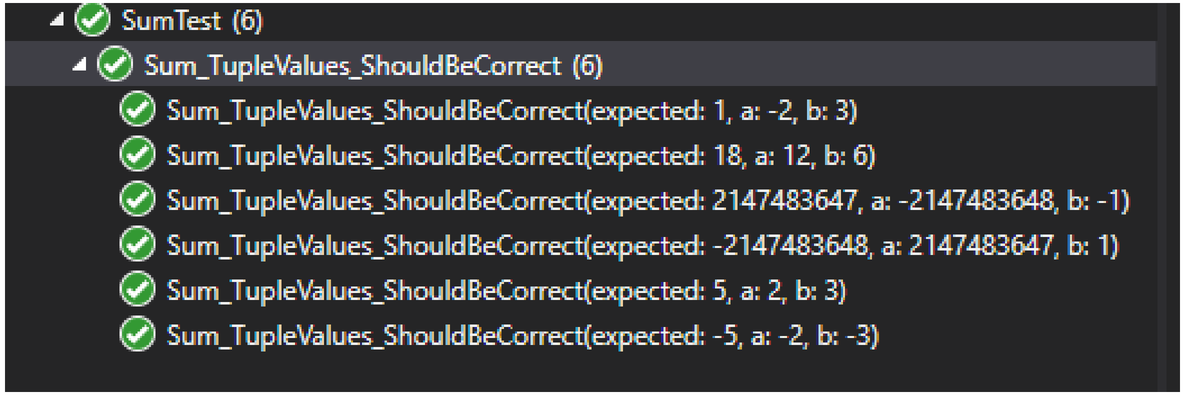
```
[InlineData(-5, -2, -3)]
```



```
[InlineData(int.MinValue,
    int.MaxValue, 1)]
```



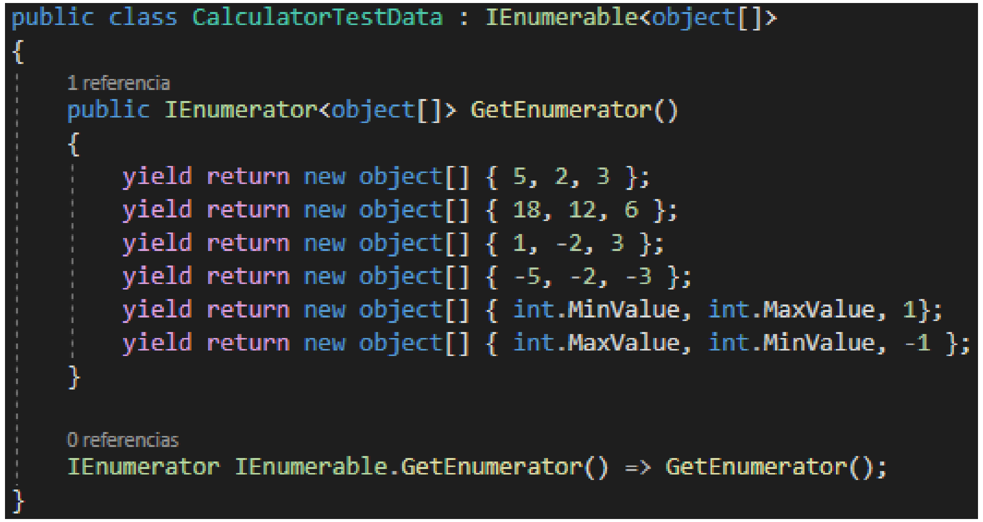
```
[InlineData(int.MaxValue,
    int.MinValue, -1)]
```



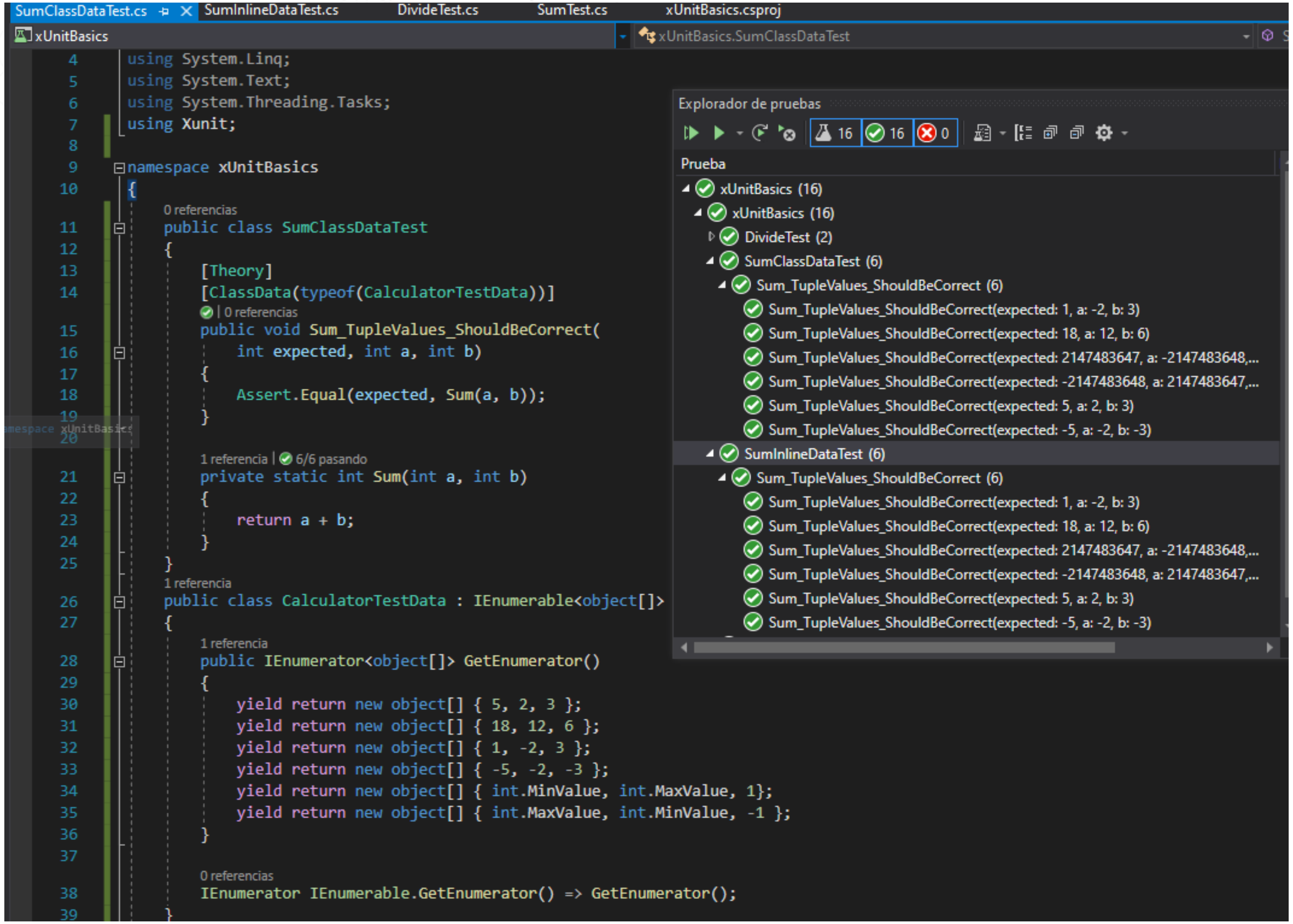
```
◢ ✓ SumTest (6)
  ◢ ✓ Sum_TupleValues_ShouldBeCorrect (6)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 1, a: -2, b: 3)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 2147483647, a: -2147483648, b: -1)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: -2147483648, a: 2147483647, b: 1)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: -5, a: -2, b: -3)
```

xUnit 101

Theories - ClassData



```csharp
public class CalculatorTestData : IEnumerable<object[]>
{
    1 referencia
    public IEnumerator<object[]> GetEnumerator()
    {
        yield return new object[] { 5, 2, 3 };
        yield return new object[] { 18, 12, 6 };
        yield return new object[] { 1, -2, 3 };
        yield return new object[] { -5, -2, -3 };
        yield return new object[] { int.MinValue, int.MaxValue, 1};
        yield return new object[] { int.MaxValue, int.MinValue, -1 };
    }

    0 referencias
    IEnumerator IEnumerable.GetEnumerator() => GetEnumerator();
}
```



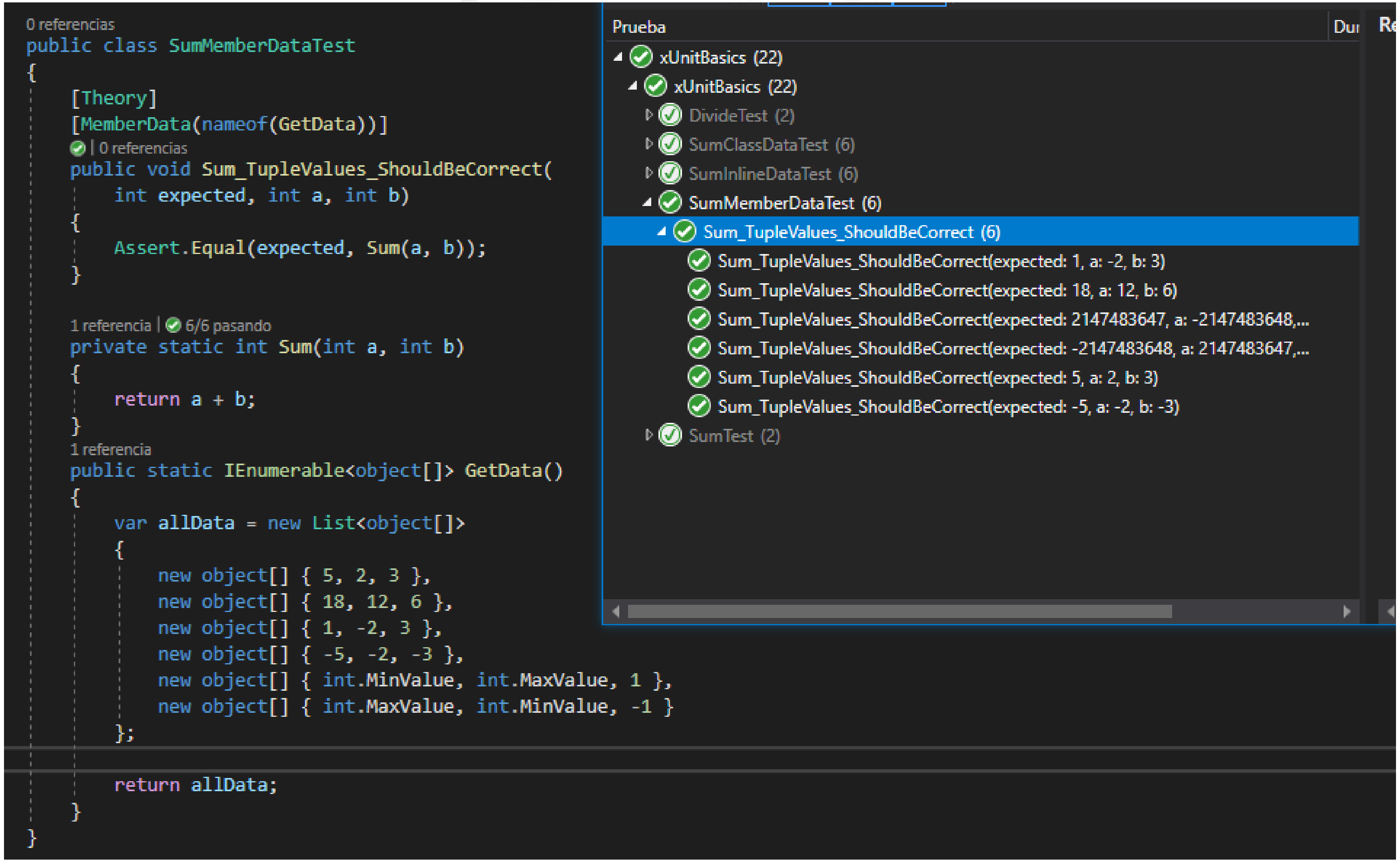
```csharp
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Xunit;

namespace xUnitBasics
{
    0 referencias
    public class SumClassDataTest
    {
        [Theory]
        [ClassData(typeof(CalculatorTestData))]
        ✓ | 0 referencias
        public void Sum_TupleValues_ShouldBeCorrect(
            int expected, int a, int b)
        {
            Assert.Equal(expected, Sum(a, b));
        }

        1 referencia | ✓ 6/6 pasando
        private static int Sum(int a, int b)
        {
            return a + b;
        }
    }
    1 referencia
    public class CalculatorTestData : IEnumerable<object[]>
    {
        1 referencia
        public IEnumerator<object[]> GetEnumerator()
        {
            yield return new object[] { 5, 2, 3 };
            yield return new object[] { 18, 12, 6 };
            yield return new object[] { 1, -2, 3 };
            yield return new object[] { -5, -2, -3 };
            yield return new object[] { int.MinValue, int.MaxValue, 1};
            yield return new object[] { int.MaxValue, int.MinValue, -1 };
        }

        0 referencias
        IEnumerator IEnumerable.GetEnumerator() => GetEnumerator();
    }
```

```
◢ ✓ xUnitBasics (16)
  ◢ ✓ xUnitBasics (16)
    ▷ ✓ DivideTest (2)
    ◢ ✓ SumClassDataTest (6)
      ◢ ✓ Sum_TupleValues_ShouldBeCorrect (6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 1, a: -2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 2147483647, a: -2147483648,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -2147483648, a: 2147483647,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -5, a: -2, b: -3)
    ◢ ✓ SumInlineDataTest (6)
      ◢ ✓ Sum_TupleValues_ShouldBeCorrect (6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 1, a: -2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 2147483647, a: -2147483648,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -2147483648, a: 2147483647,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -5, a: -2, b: -3)
```

https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/yield

xUnit 101

Theories - MemberData

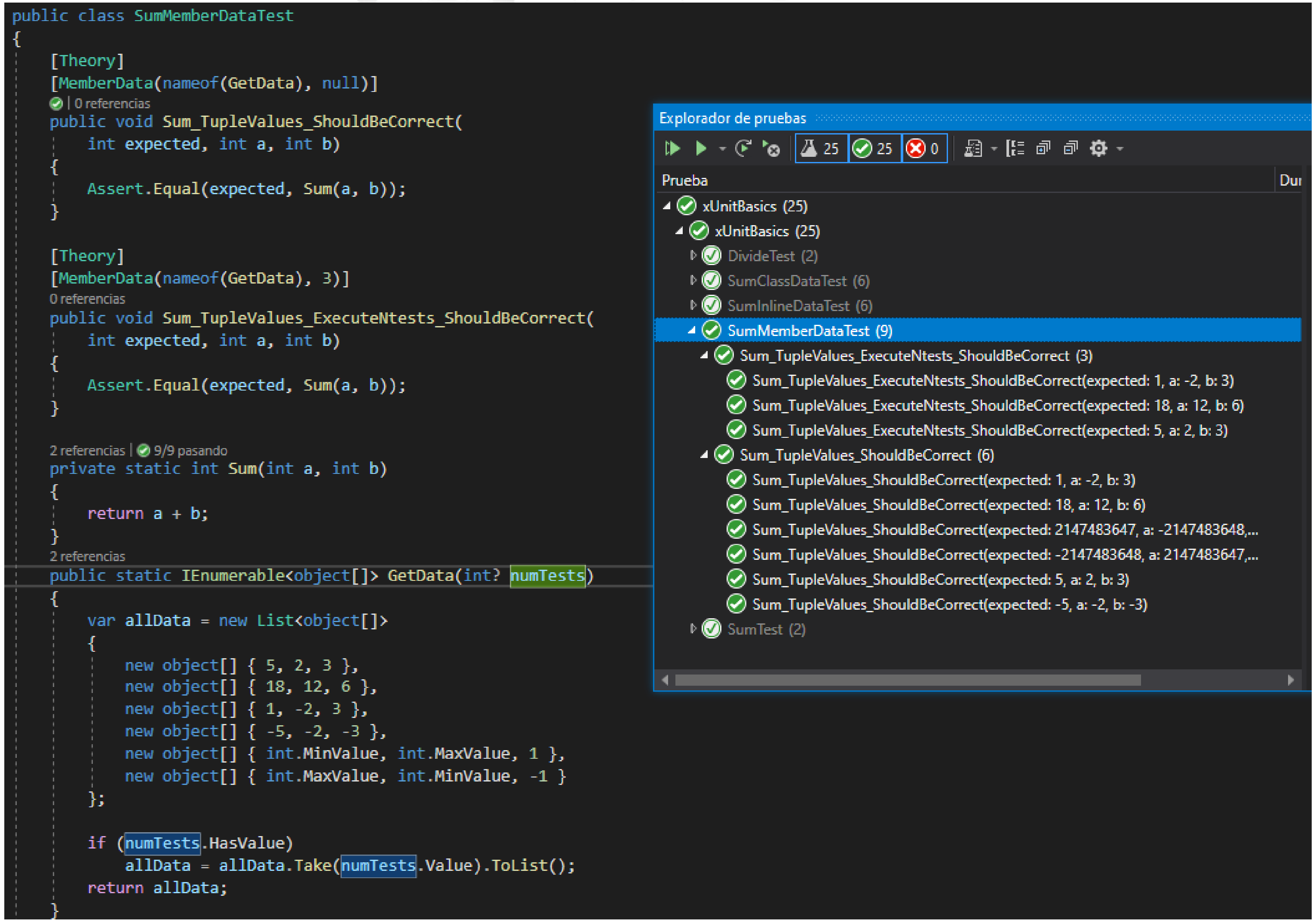


```
0 referencias
public class SumMemberDataTest
{
    [Theory]
    [MemberData(nameof(GetData))]
    ✓ | 0 referencias
    public void Sum_TupleValues_ShouldBeCorrect(
        int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }

    1 referencia | ✓ 6/6 pasando
    private static int Sum(int a, int b)
    {
        return a + b;
    }
    1 referencia
    public static IEnumerable<object[]> GetData()
    {
        var allData = new List<object[]>
        {
            new object[] { 5, 2, 3 },
            new object[] { 18, 12, 6 },
            new object[] { 1, -2, 3 },
            new object[] { -5, -2, -3 },
            new object[] { int.MinValue, int.MaxValue, 1 },
            new object[] { int.MaxValue, int.MinValue, -1 }
        };

        return allData;
    }
}
```

```
Prueba                                                                 Dur   Re
◢ ✓ xUnitBasics (22)
  ◢ ✓ xUnitBasics (22)
    ▷ ✓ DivideTest (2)
    ▷ ✓ SumClassDataTest (6)
    ▷ ✓ SumInlineDataTest (6)
    ◢ ✓ SumMemberDataTest (6)
      ◢ ✓ Sum_TupleValues_ShouldBeCorrect (6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 1, a: -2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 2147483647, a: -2147483648,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -2147483648, a: 2147483647,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -5, a: -2, b: -3)
    ▷ ✓ SumTest (2)
```

xUnit 101

Theories - MemberData



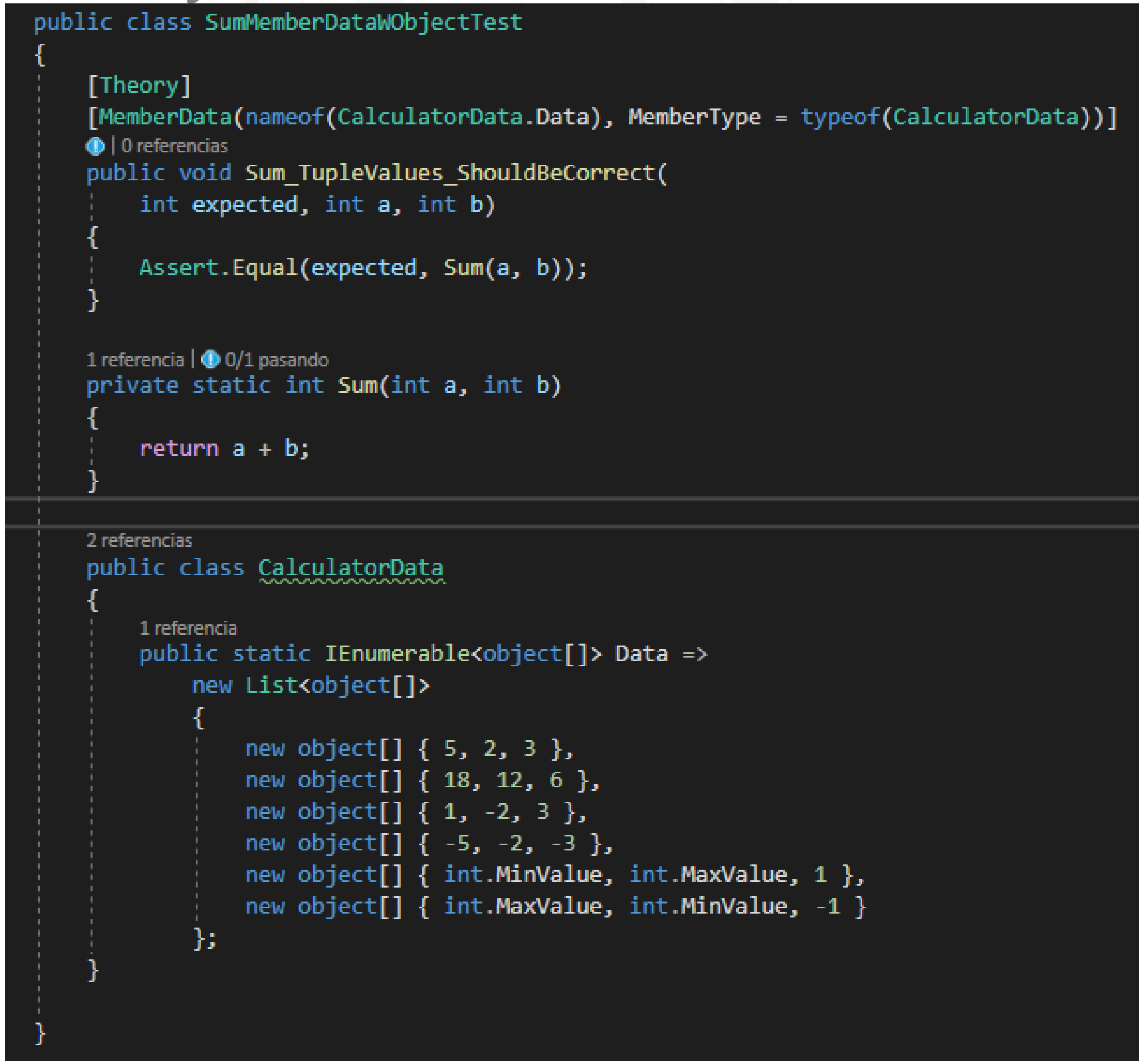
```
public class SumMemberDataTest
{
    [Theory]
    [MemberData(nameof(GetData), null)]
    ✓ | 0 referencias
    public void Sum_TupleValues_ShouldBeCorrect(
        int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }

    [Theory]
    [MemberData(nameof(GetData), 3)]
    0 referencias
    public void Sum_TupleValues_ExecuteNtests_ShouldBeCorrect(
        int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }

    2 referencias | ✓ 9/9 pasando
    private static int Sum(int a, int b)
    {
        return a + b;
    }
    2 referencias
    public static IEnumerable<object[]> GetData(int? numTests)
    {
        var allData = new List<object[]>
        {
            new object[] { 5, 2, 3 },
            new object[] { 18, 12, 6 },
            new object[] { 1, -2, 3 },
            new object[] { -5, -2, -3 },
            new object[] { int.MinValue, int.MaxValue, 1 },
            new object[] { int.MaxValue, int.MinValue, -1 }
        };

        if (numTests.HasValue)
            allData = allData.Take(numTests.Value).ToList();
        return allData;
    }
```

Explorador de pruebas

```
◢ ✓ xUnitBasics (25)
  ◢ ✓ xUnitBasics (25)
    ▷ ✓ DivideTest (2)
    ▷ ✓ SumClassDataTest (6)
    ▷ ✓ SumInlineDataTest (6)
    ◢ ✓ SumMemberDataTest (9)
      ◢ ✓ Sum_TupleValues_ExecuteNtests_ShouldBeCorrect (3)
          ✓ Sum_TupleValues_ExecuteNtests_ShouldBeCorrect(expected: 1, a: -2, b: 3)
          ✓ Sum_TupleValues_ExecuteNtests_ShouldBeCorrect(expected: 18, a: 12, b: 6)
          ✓ Sum_TupleValues_ExecuteNtests_ShouldBeCorrect(expected: 5, a: 2, b: 3)
      ◢ ✓ Sum_TupleValues_ShouldBeCorrect (6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 1, a: -2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 2147483647, a: -2147483648,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -2147483648, a: 2147483647,...
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)
          ✓ Sum_TupleValues_ShouldBeCorrect(expected: -5, a: -2, b: -3)
    ▷ ✓ SumTest (2)
```

xUnit 101

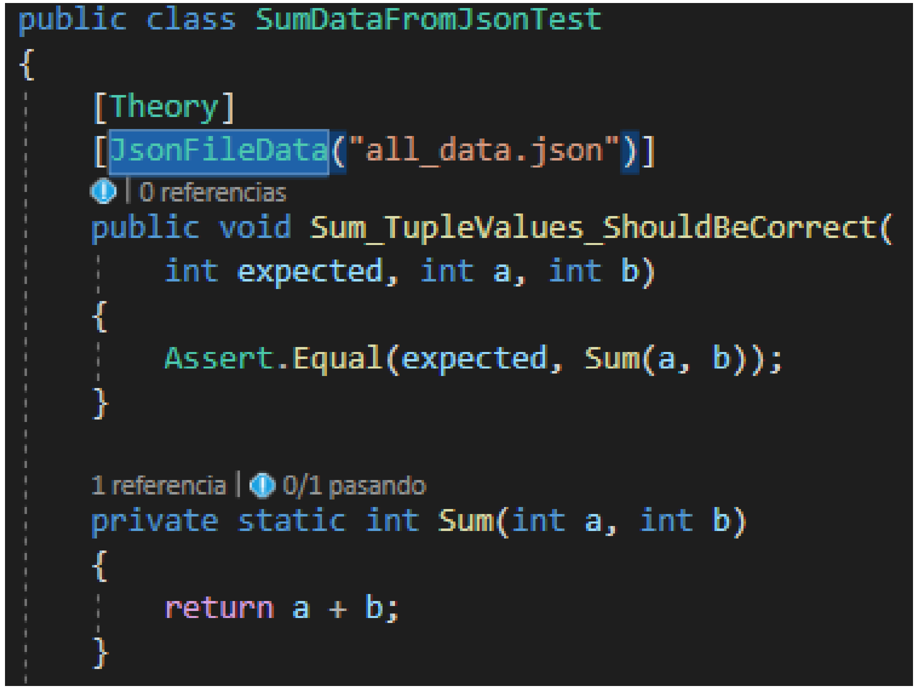
Theories – MemberData + Object



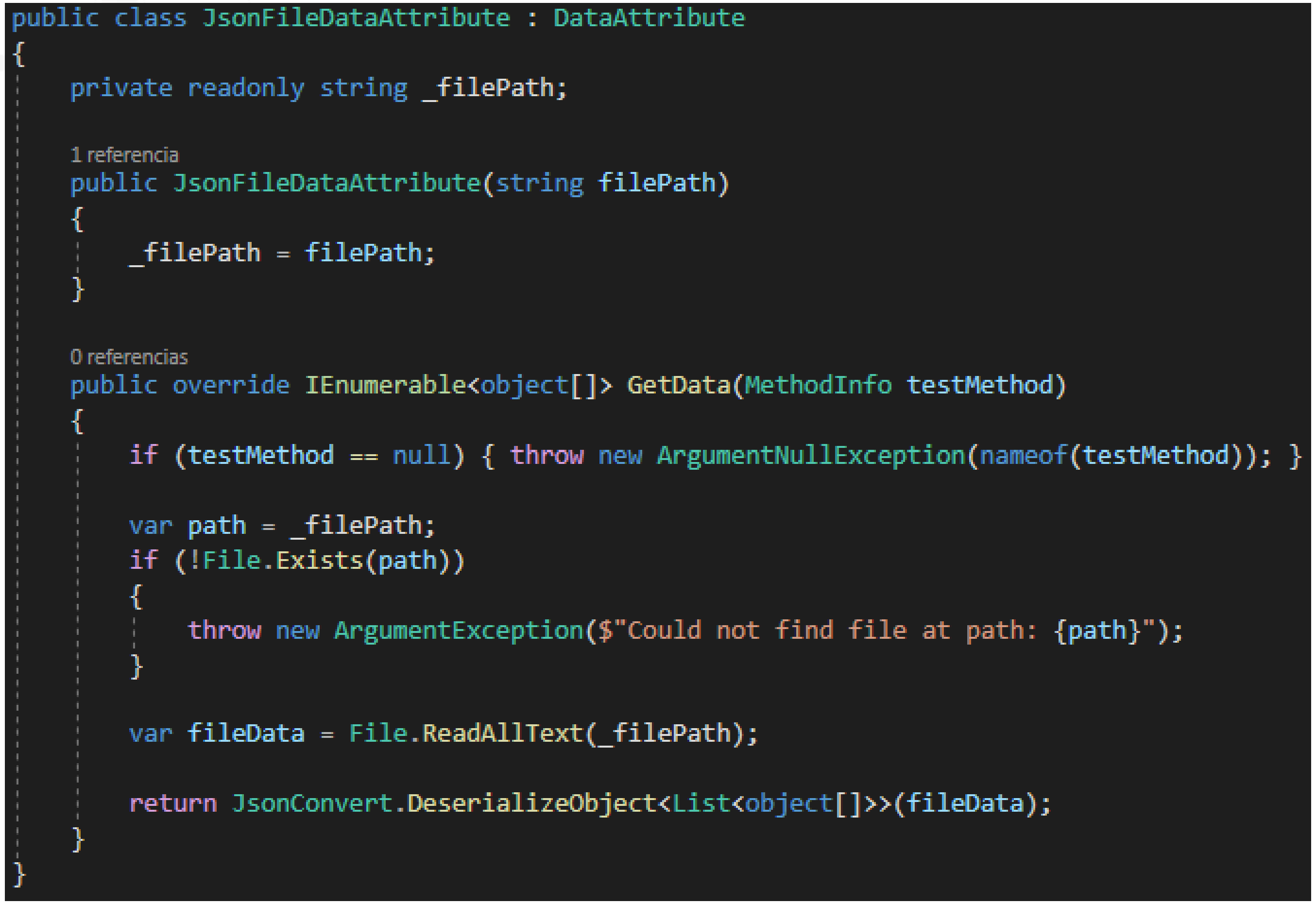
```
public class SumMemberDataWObjectTest
{
    [Theory]
    [MemberData(nameof(CalculatorData.Data), MemberType = typeof(CalculatorData))]
    | 0 referencias
    public void Sum_TupleValues_ShouldBeCorrect(
        int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }

    1 referencia | 0/1 pasando
    private static int Sum(int a, int b)
    {
        return a + b;
    }

    2 referencias
    public class CalculatorData
    {
        1 referencia
        public static IEnumerable<object[]> Data =>
            new List<object[]>
            {
                new object[] { 5, 2, 3 },
                new object[] { 18, 12, 6 },
                new object[] { 1, -2, 3 },
                new object[] { -5, -2, -3 },
                new object[] { int.MinValue, int.MaxValue, 1 },
                new object[] { int.MaxValue, int.MinValue, -1 }
            };
    }

}
```

xUnit 101

Theories – Custom data from files



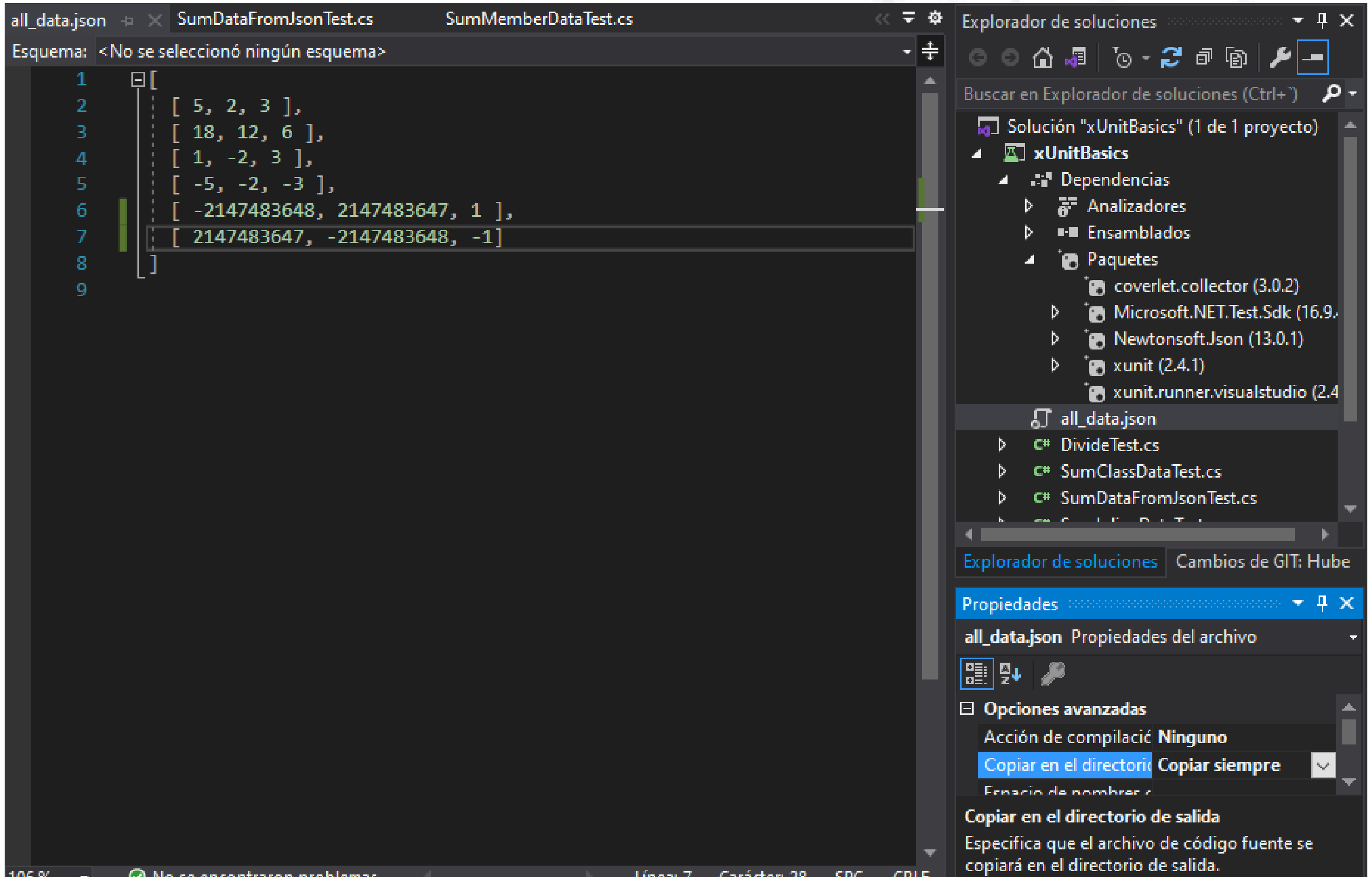
```
public class SumDataFromJsonTest
{
    [Theory]
    [JsonFileData("all_data.json")]
    ❗ | 0 referencias
    public void Sum_TupleValues_ShouldBeCorrect(
        int expected, int a, int b)
    {
        Assert.Equal(expected, Sum(a, b));
    }

    1 referencia | ❗ 0/1 pasando
    private static int Sum(int a, int b)
    {
        return a + b;
    }
```



```
public class JsonFileDataAttribute : DataAttribute
{
    private readonly string _filePath;

    1 referencia
    public JsonFileDataAttribute(string filePath)
    {
        _filePath = filePath;
    }

    0 referencias
    public override IEnumerable<object[]> GetData(MethodInfo testMethod)
    {
        if (testMethod == null) { throw new ArgumentNullException(nameof(testMethod)); }

        var path = _filePath;
        if (!File.Exists(path))
        {
            throw new ArgumentException($"Could not find file at path: {path}");
        }

        var fileData = File.ReadAllText(_filePath);

        return JsonConvert.DeserializeObject<List<object[]>>(fileData);
    }
}
```

xUnit 101

Theories – Custom data from files



```
[
  [ 5, 2, 3 ],
  [ 18, 12, 6 ],
  [ 1, -2, 3 ],
  [ -5, -2, -3 ],
  [ -2147483648, 2147483647, 1 ],
  [ 2147483647, -2147483648, -1]
]
```

106 %
No se encontraron problemas
Líneas: 7
Carácter: 28
SPC
CRLF
Explorador de soluciones
Buscar en Explorador de soluciones (Ctrl+´)

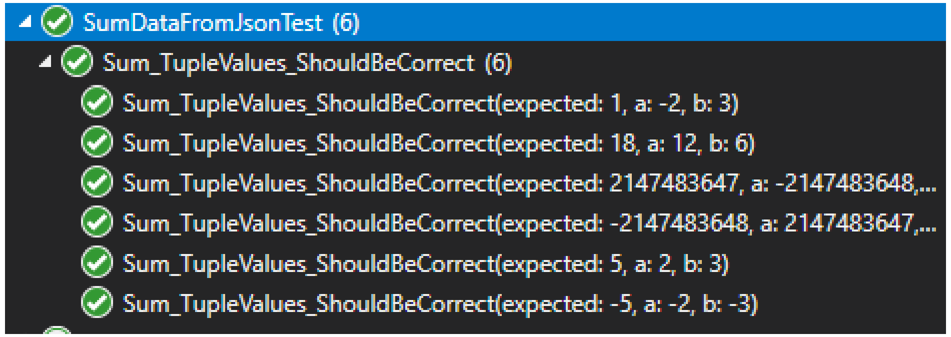
Solución "xUnitBasics" (1 de 1 proyecto)
- xUnitBasics
  - Dependencias
    - Analizadores
    - Ensamblados
    - Paquetes
      - coverlet.collector (3.0.2)
      - Microsoft.NET.Test.Sdk (16.9.
      - Newtonsoft.Json (13.0.1)
      - xunit (2.4.1)
      - xunit.runner.visualstudio (2.4
  - all_data.json
  - DivideTest.cs
  - SumClassDataTest.cs
  - SumDataFromJsonTest.cs

all_data.json Propiedades del archivo

Opciones avanzadas

Acción de compilació	Ninguno
Copiar en el directori	Copiar siempre
Espacio de nombres c	

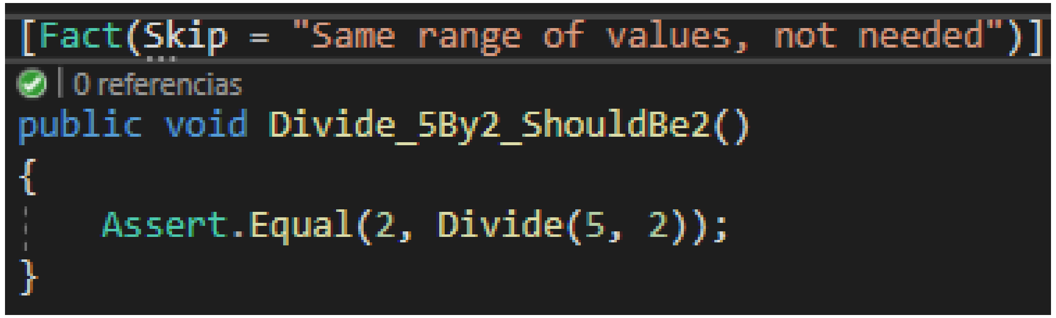
Copiar en el directorio de salida
Especifica que el archivo de código fuente se copiará en el directorio de salida.



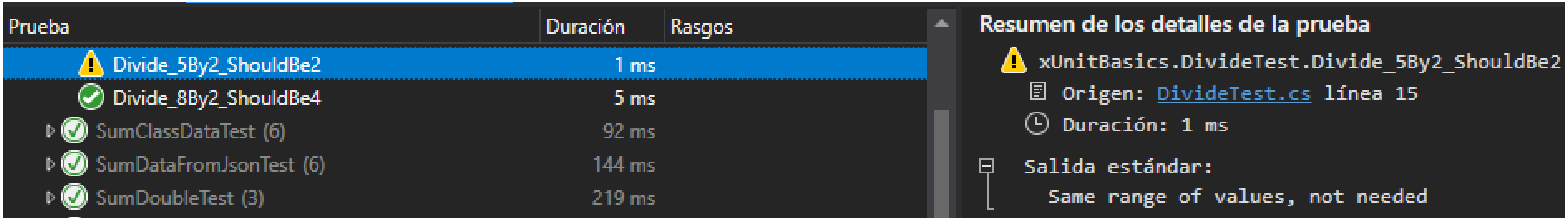
```
◢ ✓ SumDataFromJsonTest (6)
  ◢ ✓ Sum_TupleValues_ShouldBeCorrect (6)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 1, a: -2, b: 3)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 18, a: 12, b: 6)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 2147483647, a: -2147483648,...
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: -2147483648, a: 2147483647,...
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: 5, a: 2, b: 3)
      ✓ Sum_TupleValues_ShouldBeCorrect(expected: -5, a: -2, b: -3)
```

xUnit 101

Skip Tests



```
[Fact(Skip = "Same range of values, not needed")]
0 referencias
public void Divide_5By2_ShouldBe2()
{
    Assert.Equal(2, Divide(5, 2));
}
```



```
Prueba                              Duración   Rasgos
  ⚠ Divide_5By2_ShouldBe2              1 ms
  ✓ Divide_8By2_ShouldBe4              5 ms
▷ ✓ SumClassDataTest (6)              92 ms
▷ ✓ SumDataFromJsonTest (6)          144 ms
▷ ✓ SumDoubleTest (3)                219 ms
```

Resumen de los detalles de la prueba

```
⚠ xUnitBasics.DivideTest.Divide_5By2_ShouldBe2
    Origen: DivideTest.cs línea 15
    Duración: 1 ms

  Salida estándar:
    Same range of values, not needed
```

hiberus© TECNOLOGIA

La compañía hiperespecializada en las TIC

www.hiberus.com