

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования «Рязанский государственный радиотехнический
университет имени В.Ф. Уткина»
Рязанский станкостроительный колледж

Отчёт о практической работе

Unit Test 1-4

«МДК 05.03»

Выполнила:

Студентка группы ИСП-32

Фролова Е.О.

Проверил:

Родин Е.Н.

Цель работы: Основная цель данной работы заключалась в освоении принципов модульного тестирования (Unit Testing) и приобретении практических навыков написания unit-тестов для обеспечения надежности и качества программного кода.

Ход работы:

Unit Test 1.

- 1) Создали консольное приложение:

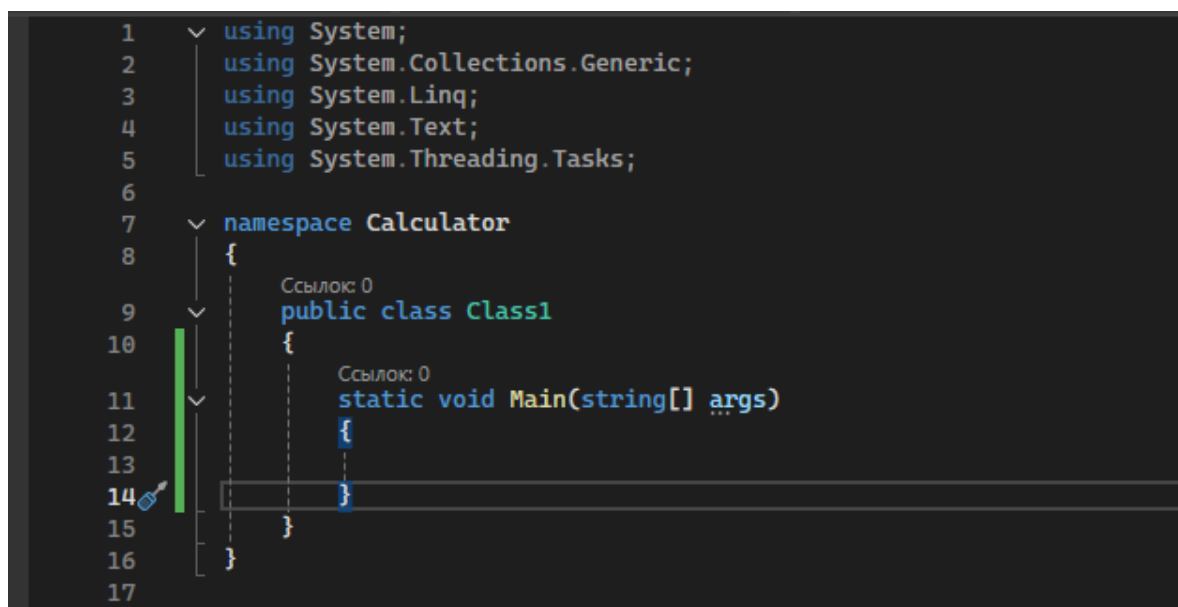


Рисунок 1 – Консольное приложение

- 2) Добавили класс, в котором будут производиться математические операции:

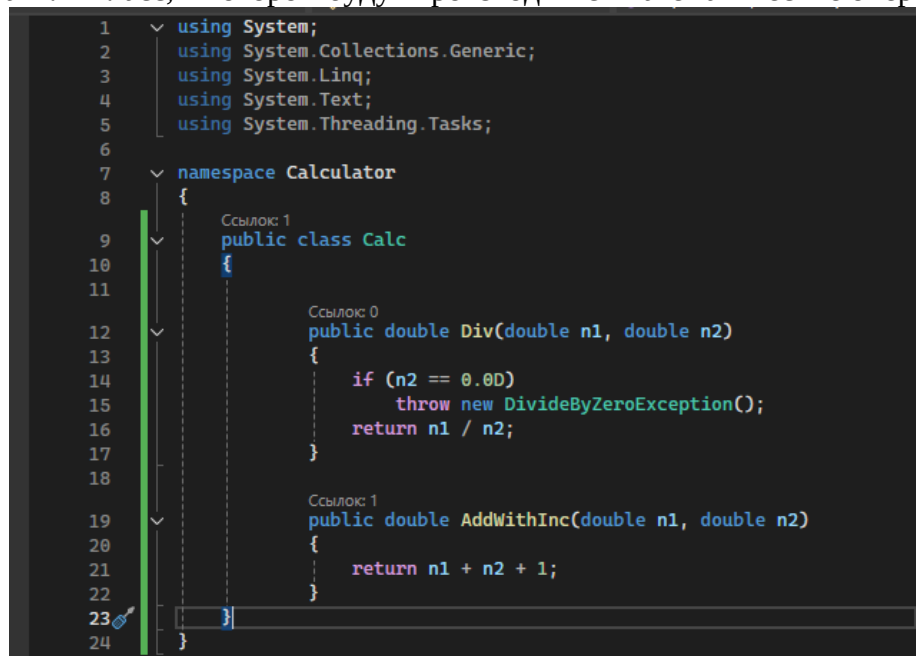
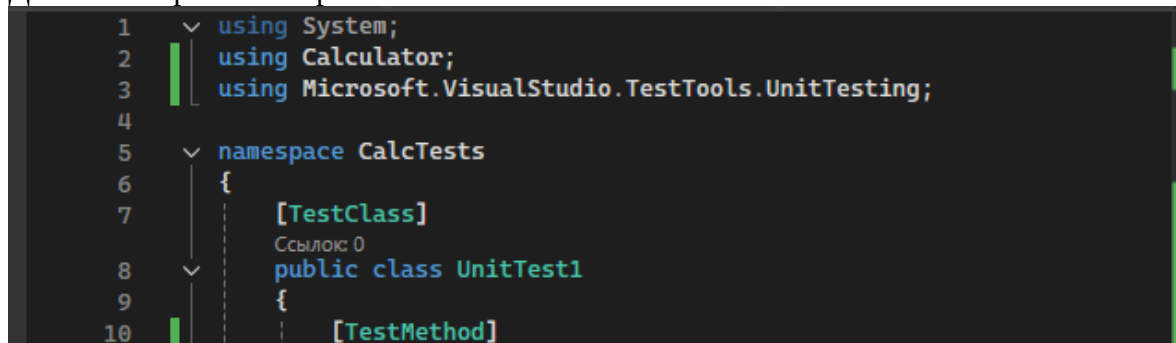


Рисунок 2 – Добавление класса

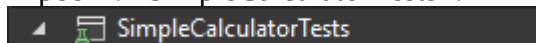
- 3) Добавили в решение проект тестов:



```
1  using System;
2  using Calculator;
3  using Microsoft.VisualStudio.TestTools.UnitTesting;
4
5  namespace CalcTests
6  {
7      [TestClass]
8      public class UnitTest1
9      {
10         [TestMethod]
```

Рисунок 3 – Добавили в решение проект тестов

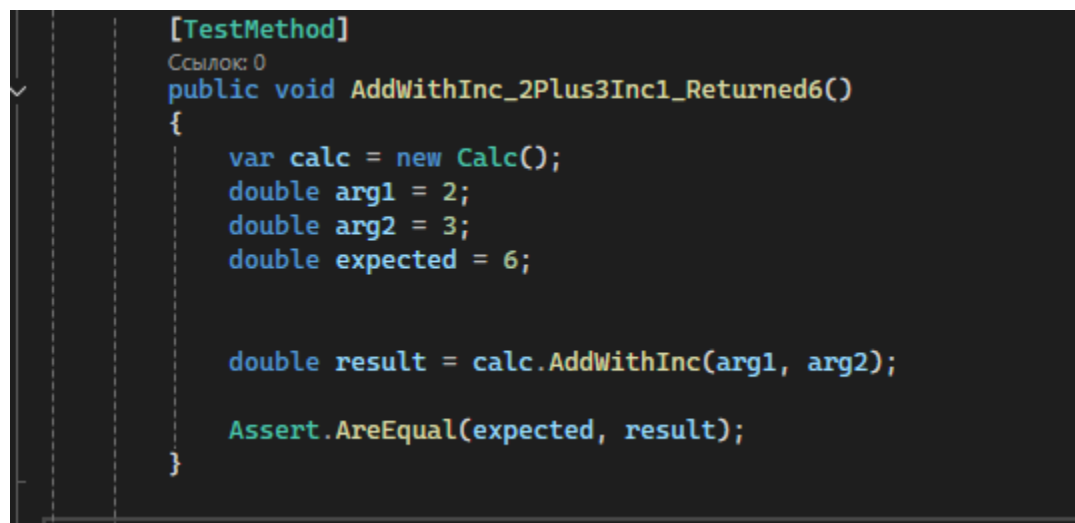
- 4) Переименовали наш проект: «SimpleCalculatorTests»:



SimpleCalculatorTests

Рисунок 4 – Переименованный проект

- 5) Добавили в проект тест для проверки метода AddWithInc



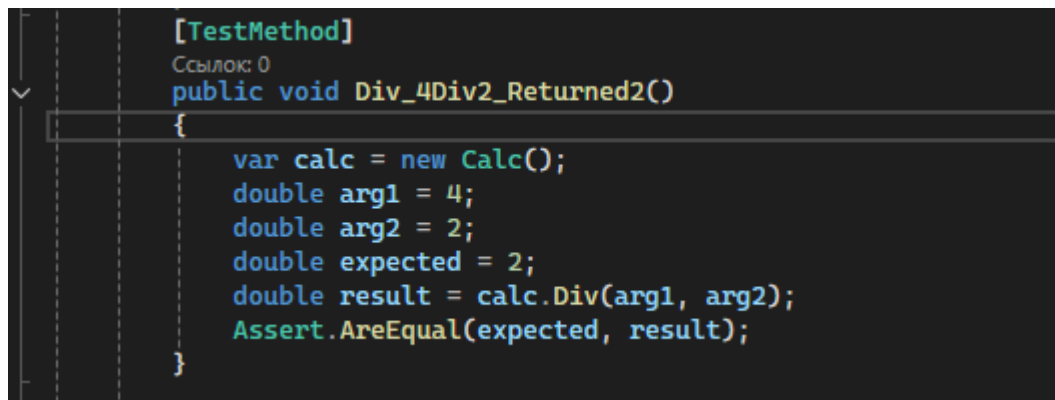
```
[TestMethod]
Ссылка: 0
public void AddWithInc_2Plus3Inc1_Returned6()
{
    var calc = new Calc();
    double arg1 = 2;
    double arg2 = 3;
    double expected = 6;

    double result = calc.AddWithInc(arg1, arg2);

    Assert.AreEqual(expected, result);
}
```

Рисунок 5 – Добавили метода AddWithInc

- 6) Проверили метод Div



```
[TestMethod]
Ссылка: 0
public void Div_4Div2_Returned2()
{
    var calc = new Calc();
    double arg1 = 4;
    double arg2 = 2;
    double expected = 2;
    double result = calc.Div(arg1, arg2);
    Assert.AreEqual(expected, result);
}
```

Рисунок 6 – Добавили метода Div

7) Следующий тест будет проверять операцию деления на нуль в методе Div.

```
[TestMethod]
[ExpectedException(typeof(DivideByZeroException),
    "Oh my god, we can't divison on zero")]
Ссылка: 0
public void Div_4Div0_ZeroDivException()
{
    var calc = new Calc();
    double arg1 = 4;
    double arg2 = 0;
    double result = calc.Div(arg1, arg2);
}
```

Рисунок 7 - Тест операции деления на нуль в методе Div.

8) Запустили тест:

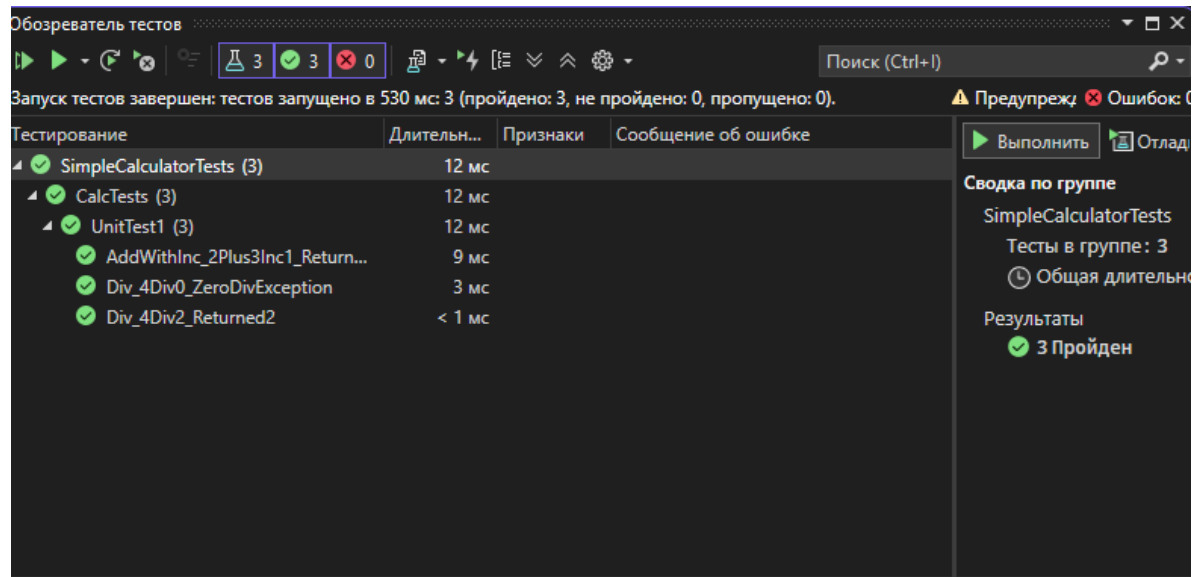


Рисунок 8 – Запуск теста

Unit Test 2.

- 1) Создали класс, который будет содержать методы для тестирования:

```
public class Calculator
{
    Ссылка: 0
    public int Add(int a, int b)
    {
        return a + b;
    }

    Ссылка: 0
    public int Subtract(int a, int b)
    {
        return a - b;
    }
}
```

Рисунок 1 - Класс, который содержит методы для тестирования

- 2) Создали новый проект в Visual Studio, выбрав шаблон MSTest Test Project.

```
using Calculator2;

namespace CalculatorTests
{
    [TestClass]
    Ссылка: 0
    public class CalculatorTests
    {
        [TestMethod]
```

Рисунок 2 – Новый проект MSTest Test Project.

3) Добавили ссылку на проект с классом Calculator2.

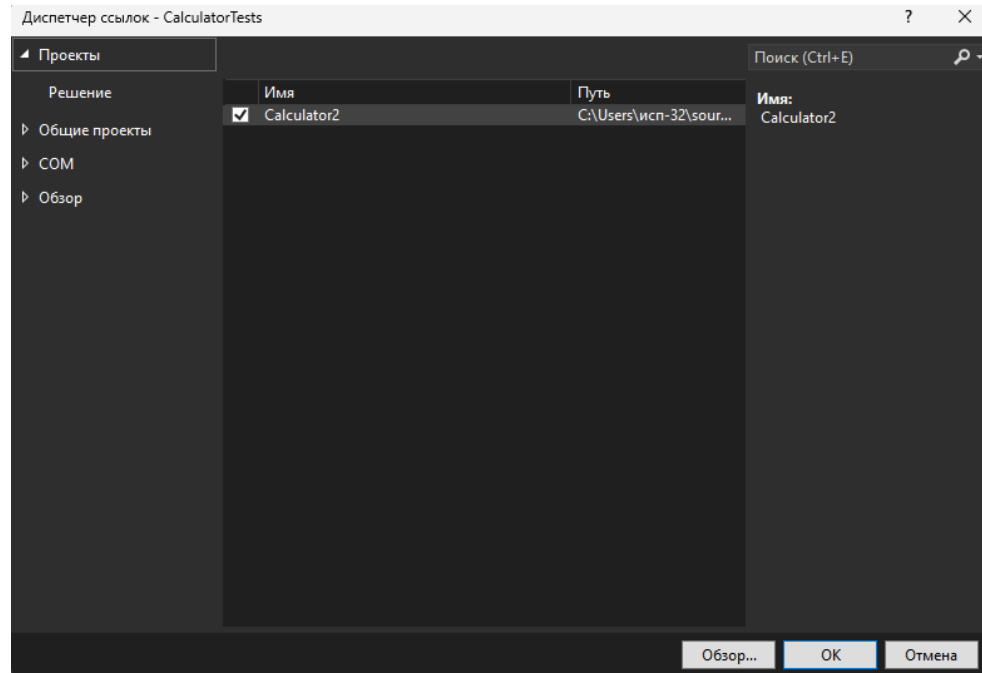


Рисунок 3 – Ссылка на проект

4) Написали тесты для методов Add и Subtract:

```
1  using Calculator2;
2
3  namespace CalculatorTests
4  {
5      [TestClass]
6      public class CalculatorTests
7      {
8          [TestMethod]
9          public void Add_WhenCalled_ReturnsSumOfArguments()
10         {
11             var calculator = new Calculator();
12             var result = calculator.Add(2, 3);
13             Assert.AreEqual(5, result);
14         }
15
16         [TestMethod]
17         public void Subtract_WhenCalled_ReturnsSumOfArguments()
18         {
19             var calculator = new Calculator();
20             var result = calculator.Subtract(5, 3);
21             Assert.AreEqual(2, result);
22         }
23     }
24 }
25
26
27
28
29
30
31
```

Рисунок 4 - Тесты для методов Add и Subtract

5) Запустили тесты;

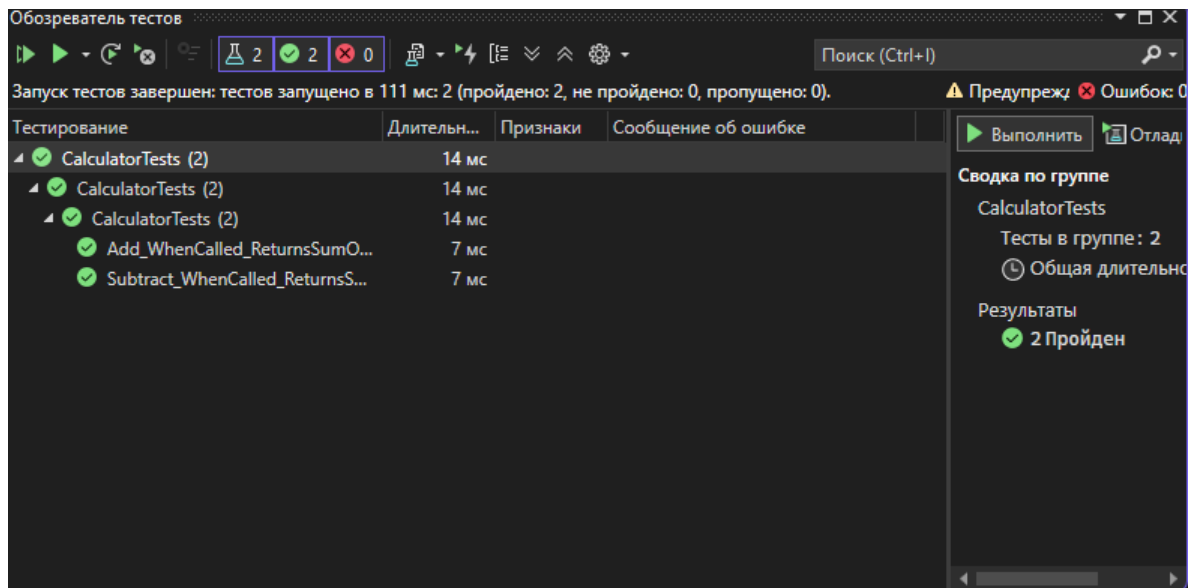
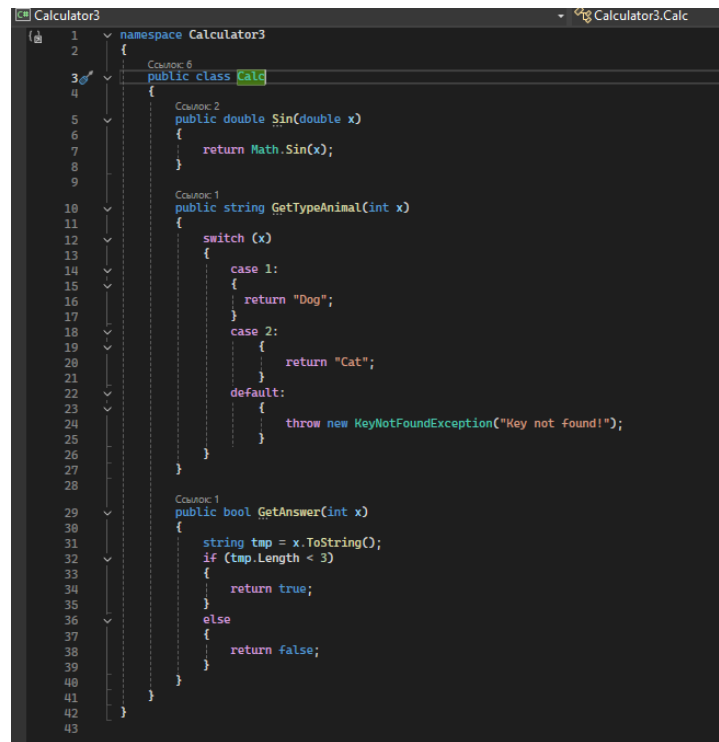


Рисунок 5 – Запуск тестов

Unit Test 3.

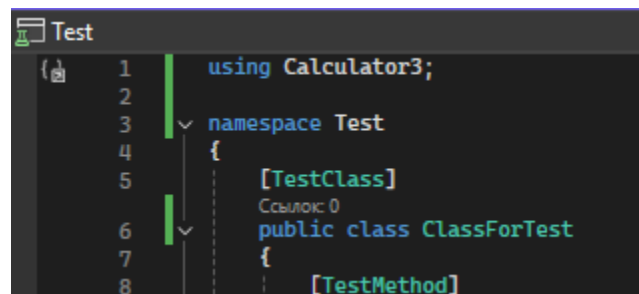
- 1) Создали класс, который будет содержать методы для тестирования:



```
1 namespace Calculator3
2 {
3     public class Calc
4     {
5         public double Sin(double x)
6         {
7             return Math.Sin(x);
8         }
9
10        public string GetTypeAnimal(int x)
11        {
12            switch (x)
13            {
14                case 1:
15                {
16                    return "Dog";
17                }
18                case 2:
19                {
20                    return "Cat";
21                }
22                default:
23                {
24                    throw new KeyNotFoundException("Key not found!");
25                }
26            }
27        }
28
29        public bool GetAnswer(int x)
30        {
31            string tmp = x.ToString();
32            if (tmp.Length < 3)
33            {
34                return true;
35            }
36            else
37            {
38                return false;
39            }
40        }
41    }
42 }
43
```

Рисунок 1 - Класс, который содержит методы для тестирования

- 2) Создали новый проект в Visual Studio, выбрав шаблон MSTest Test Project



```
1 using Calculator3;
2
3 namespace Test
4 {
5     [TestClass]
6     public class ClassForTest
7     {
8         [TestMethod]
9     }
10 }
```

Рисунок 2 - Новый проект в Visual Studio MSTest Test Project

3) Добавили ссылку на проект с классом Calculator3.

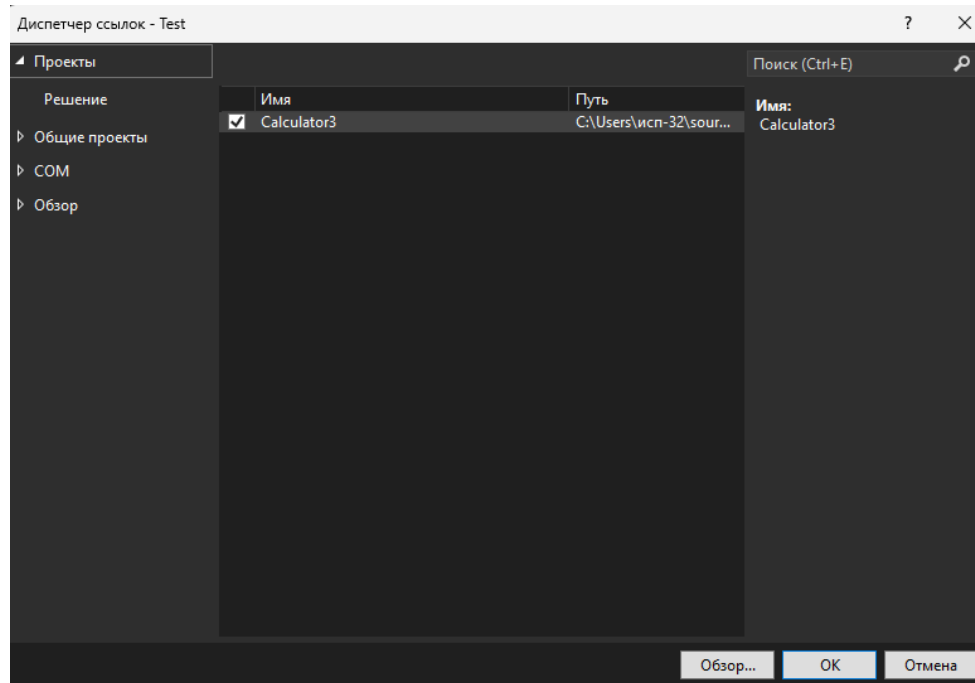


Рисунок 3 – Ссылка на проект

4) Написали тесты для методов

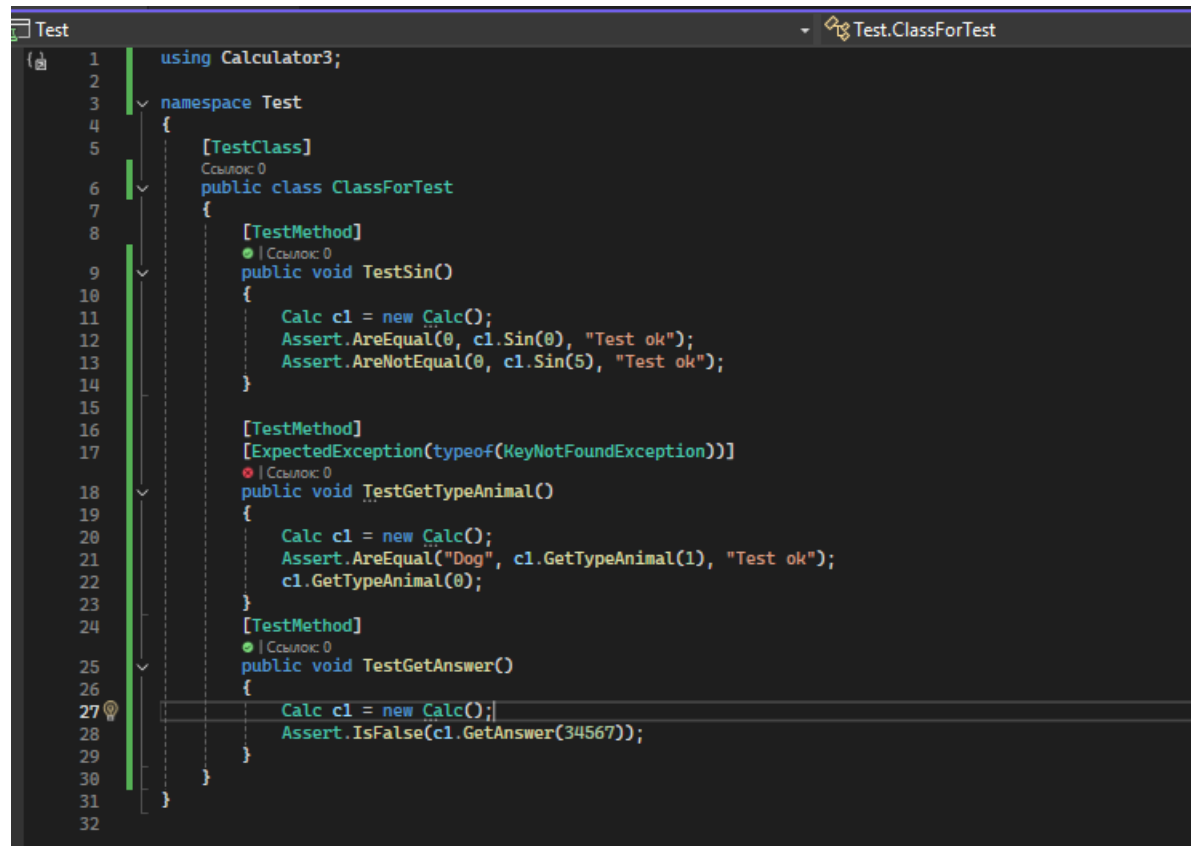


Рисунок 4 – Тесты для методов

5) Запустили тесты

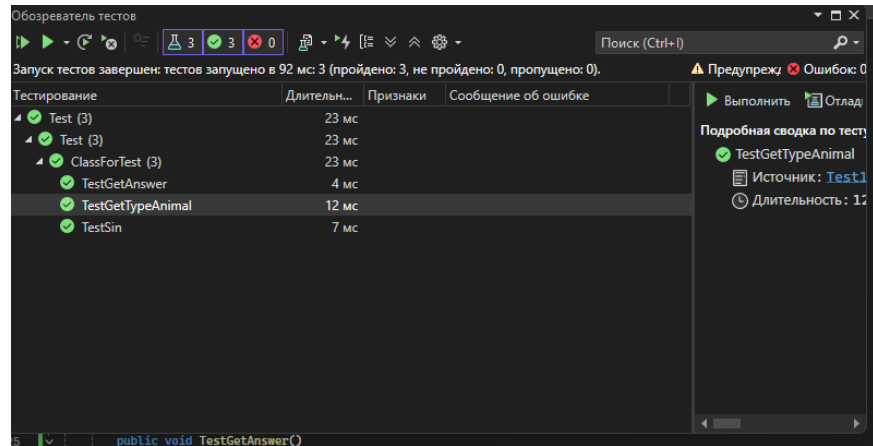


Рисунок 5 – Запуск тестов

6) Провели анализ

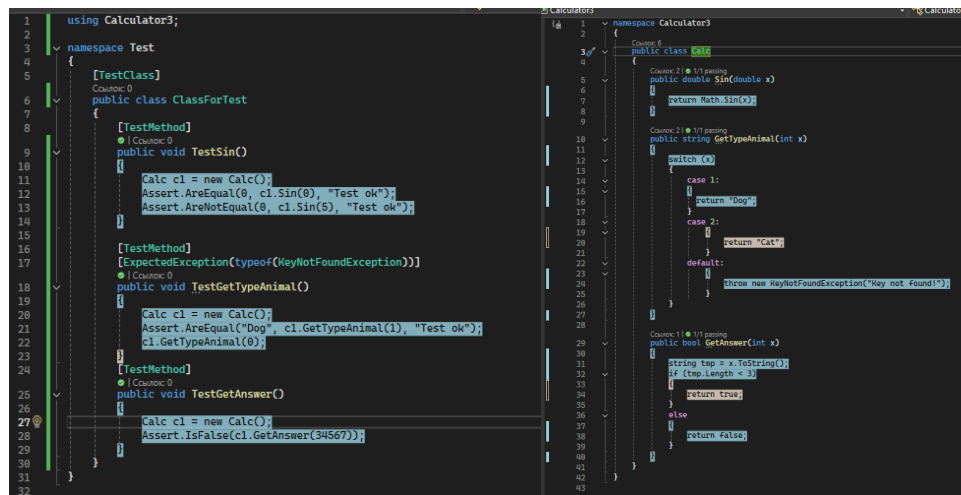
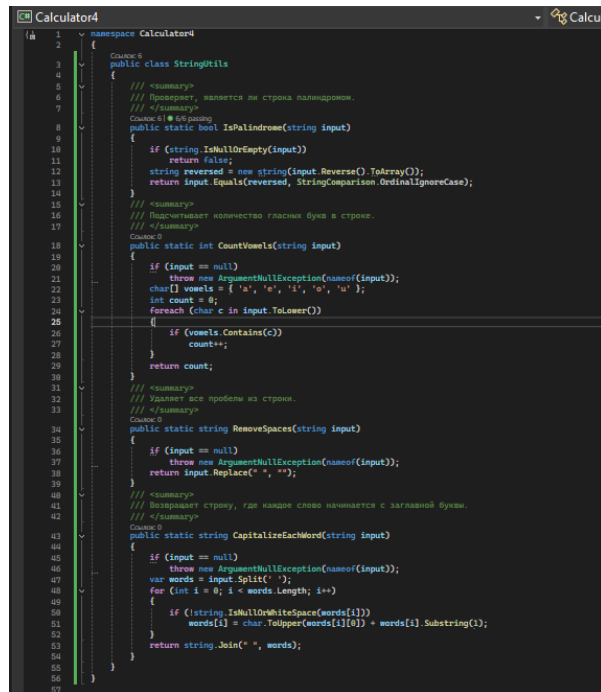


Рисунок 6, 7 – Анализ

Unit Test 4.

- 1) Класс StringUtils. Этот класс содержит набор методов для работы со строками:



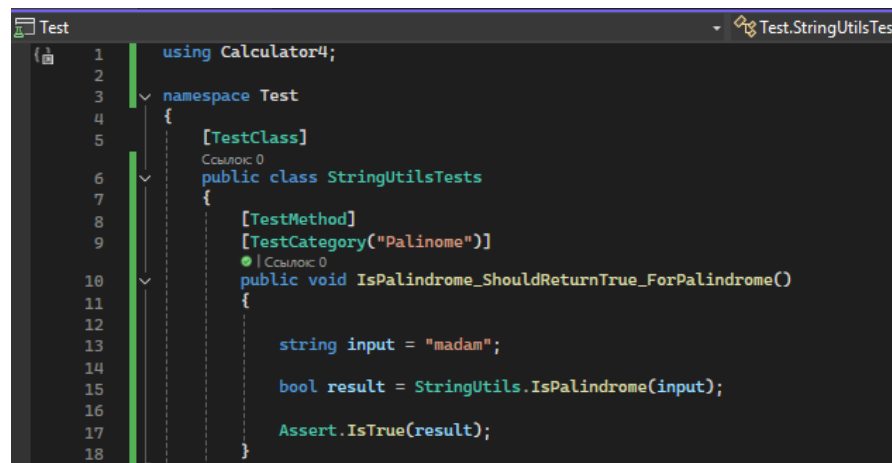
```
1 namespace Calculator4
2 {
3     class StringUtils
4     {
5         /// <summary>
6         /// Проверяет, является ли строка палиндромом.
7         /// </summary>
8         public static bool IsPalindrome(string input)
9         {
10             if (string.IsNullOrEmpty(input))
11                 return false;
12             string reversed = new string(input.Reverse().ToArray());
13             return input.Equals(reversed, StringComparison.OrdinalIgnoreCase);
14         }
15         /// <summary>
16         /// Подсчитывает количество разных букв в строке.
17         /// </summary>
18         public static int CountVowels(string input)
19         {
20             if (input == null)
21                 throw new ArgumentNullException(nameof(input));
22             char[] vowels = { 'a', 'e', 'i', 'o', 'u' };
23             int count = 0;
24             foreach (char c in input.ToLower())
25             {
26                 if (vowels.Contains(c))
27                     count++;
28             }
29             return count;
30         }
31         /// <summary>
32         /// Удаляет все пробелы из строки.
33         /// </summary>
34         public static string RemoveSpaces(string input)
35         {
36             if (input == null)
37                 throw new ArgumentNullException(nameof(input));
38             return input.Replace(" ", "");
39         }
40         /// <summary>
41         /// Возвращает строку, где каждое слово начинается с заглавной буквы.
42         /// </summary>
43         public static string CapitalizeEachWord(string input)
44         {
45             if (input == null)
46                 throw new ArgumentNullException(nameof(input));
47             var words = input.Split(' ');
48             for (int i = 0; i < words.Length; i++)
49             {
50                 if (!string.IsNullOrWhiteSpace(words[i]))
51                     words[i] = char.ToUpper(words[i][0]) + words[i].Substring(1);
52             }
53             return string.Join(" ", words);
54         }
55     }
56 }
```

Рисунок 1 - Класс StringUtils

- 2) Написали тесты для метода IsPalindrome.

Тесты должны проверять следующие случаи:

- Строка-палиндром.



```
1 using Calculator4;
2
3 namespace Test
4 {
5     [TestClass]
6     class StringUtilsTests
7     {
8         [TestMethod]
9         [TestCategory("Palinome")]
10         public void IsPalindrome_ShouldReturnTrue_ForPalindrome()
11         {
12             string input = "madam";
13             bool result = StringUtils.IsPalindrome(input);
14             Assert.IsTrue(result);
15         }
16     }
17 }
```

Рисунок 2 – Строка палиндром

3) Строка не является палиндромом.

```
[TestMethod]
[TestCategory("NonPalinome")]
● | Ссылка: 0
public void IsPalindrome_ShouldReturnFalse_ForNonPalindrome()
{
    string input = "hello";

    bool result = StringUtils.IsPalindrome(input);

    Assert.IsFalse(result);
}
```

Рисунок 3 – Строка не палиндром

4) Пустая строка

```
[TestMethod]
[TestCategory("EmptyString")]
● | Ссылка: 0
public void IsPalindrome_ShouldReturnFalse_ForEmptyString()
{
    string input = "";

    bool result = StringUtils.IsPalindrome(input);

    Assert.IsFalse(result);
}
```

Рисунок 4 – Пустая строка

5) Строка с пробелами

```
[TestMethod]
[TestCategory("PalindromeWithSpaces")]
● | Ссылка: 0
public void IsPalindrome_ShouldReturnTrue_ForPalindromeWithSpaces()
{
    string input = "race car".Replace(" ", "");

    bool result = StringUtils.IsPalindrome(input);

    Assert.IsTrue(result);
}
```

Рисунок 5 – Строка с пробелами

6) Числовая строка

```
[TestMethod]
[TestCategory("NumericPalindrome")]
● | Ссылка: 0
public void IsPalindrome_ShouldReturnTrue_ForNumericPalindrome()
{
    string input = "12321";

    bool result = StringUtils.IsPalindrome(input);

    Assert.IsTrue(result);
}
```

Рисунок 6 – Числовая строка

7) Выполнение тестов

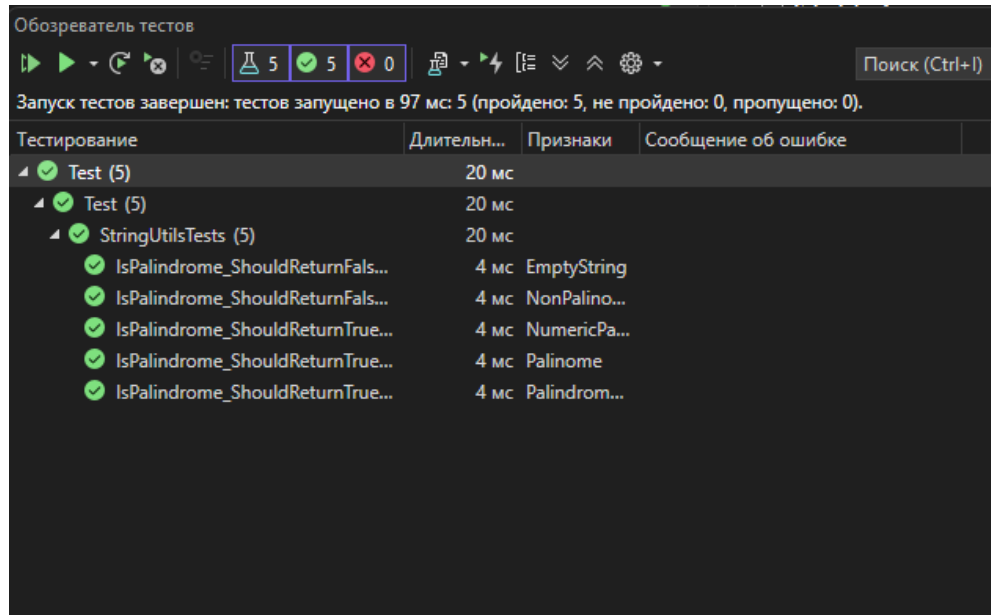


Рисунок 7 – Выполненные тесты

Задание 2: Напиши тесты для метода CountVowels.

Тесты должны проверять следующие случаи:

1) В строке есть гласные буквы.

```
namespace Test2
{
    [TestClass]
    Ссылка 0
    public class StringUtilsTests
    {
        [TestMethod]
        [TestCategory("CorrectCount")]
        Ссылка 0
        public void CountVowels_WithVowels_ReturnsCorrectCount()
        {
            string input = "hello";
            int expected = 2; // 'e' и 'o'

            int result = StringUtils.CountVowels(input);

            // Assert
            Assert.AreEqual(expected, result);
        }
    }
}
```

Рисунок 1 – в строке есть гласные буквы

2) В строке нет гласных букв.

```
[TestMethod]
[TestCategory("ReturnsZero")]
● | Ссылка: 0
public void CountVowels_NoVowels_ReturnsZero()
{
    string input = "bcdfg"; // Нет гласных
    int expected = 0;

    int result = StringUtils.CountVowels(input);

    Assert.AreEqual(expected, result);
}
```

Рисунок 2 – нет гласных

3) Пустая строка.

```
[TestMethod]
[TestCategory("EmptyString")]
● | Ссылка: 0
public void CountVowels_EmptyString_ReturnsZero()
{
    string input = "";
    int expected = 0;

    int result = StringUtils.CountVowels(input);

    Assert.AreEqual(expected, result);
}
```

Рисунок 3 – пустая строка

4) Сочетание больших и маленьких букв.

```
[TestMethod]
[TestCategory("MixedCaseVowels")]
● | Ссылка: 0
public void CountVowels_MixedCaseVowels_ReturnsCorrectCount()
{
    string input = "HeLLo";
    int expected = 2;

    int result = StringUtils.CountVowels(input);

    Assert.AreEqual(expected, result);
}
```

Рисунок 4 - Сочетание больших и маленьких букв

5) Специальные символы.

```
[TestMethod]
[TestCategory("SpecialCharacters")]
public void CountVowels_SpecialCharacters_ReturnsCorrectCount()
{
    string input = "h@e#l$l%o^";
    int expected = 2; // 'e' и 'o'

    int result = StringUtils.CountVowels(input);

    Assert.AreEqual(expected, result);
}
```

Рисунок 5 – Специальные символы

6) Выполнение тестов

Test2 (5)	35 мс	
Test2 (5)	35 мс	
StringUtilsTests (5)	35 мс	
CountVowels_EmptyString_Retu...	6 мс	EmptyString
CountVowels_MixedCaseVowels...	7 мс	MixedCase...
CountVowels_NoVowels_Return...	7 мс	ReturnsZero
CountVowels_SpecialCharacters...	8 мс	SpecialCha...
CountVowels_WithVowels_Retu...	7 мс	CorrectCo...

Рисунок 6 – Тесты

Задание 3: Напиши тесты для метода RemoveSpaces.

Тесты должны проверять следующие случаи:

1) Строка с пробелами.

```
public class StringUtilsTests
{
    [TestMethod]
    [TestCategory("InputWithSpaces")]
    public void RemoveSpaces_InputWithSpaces_ReturnsStringWithoutSpaces()
    {
        string input = "Hello World";

        string result = StringUtils.RemoveSpaces(input);

        Assert.AreEqual("HelloWorld", result);
    }
}
```

Рисунок 1 – Строка с пробелами

2) Пустая строка.

```
[TestMethod]
[TestCategory("EmptyString")]
// Ссылка: 0
public void RemoveSpaces_EmptyString_ReturnsEmptyString()
{
    string input = "";

    string result = StringUtils.RemoveSpaces(input);

    Assert.AreEqual("", result);
}
```

Рисунок 2 – Пустая строка

3) Строка без пробелов.

```
[TestMethod]
[TestCategory("InputWithoutSpaces")]
// Ссылка: 0
public void RemoveSpaces_InputWithoutSpaces_ReturnsSameString()
{
    string input = "HelloWorld";

    string result = StringUtils.RemoveSpaces(input);

    Assert.AreEqual("HelloWorld", result);
}
```

Рисунок 3 – Строка без пробелов

4) Пробелы в начале и конце строки.

```
[TestMethod]
[TestCategory("StringWithLeadingAndTrailingSpaces")]
// Ссылка: 0
public void RemoveSpaces_StringWithLeadingAndTrailingSpaces_ReturnsStringWithoutSpaces()
{
    string input = "  Hello World  ";

    string result = StringUtils.RemoveSpaces(input);

    Assert.AreEqual("HelloWorld", result);
}
```

Рисунок 4 – Пробелы в начале и в конце строки

5) Выполнение тестов

test3 (4)	20 мс
test3 (4)	20 мс
StringUtilsTests (4)	20 мс
RemoveSpaces_EmptyString_Re...	5 мс EmptyString
RemoveSpaces_InputWithoutSp...	5 мс InputWith...
RemoveSpaces_InputWithSpace...	5 мс InputWith...
RemoveSpaces_StringWithLeadi...	5 мс StringWith...

Заключение

В ходе выполнения работы были успешно освоены ключевые аспекты модульного тестирования (Unit Testing), что позволило углубить понимание принципов обеспечения качества кода и автоматизации тестирования.