

**Пермский институт (филиал) федерального
государственного бюджетного образовательного
учреждения высшего образования
«Российский экономический университет имени Г.В.
Плеханова»**

Кафедра информационных технологий и программирования.

Сопроводительная записка к практической работе №1.
Тема работы: «Средства тестирования Visual Studio-2022».

Работу выполнила:
Александрович Валерия
Николаевна
Группа: ИПс-11
Преподаватель: Берестов
Дмитрий Борисович

Пермь 2026

Оглавление.

Введение.....	3
1. Создание проекта для тестирования.	4
2. Создание проекта модульного теста.	11
3. Создание тестового класса.	15
4. Создание метода теста.	17
5. Сборка и запуск теста.	19
6. Исправление кода и повторный запуск тестов.	22
7. Создание и запуск новых методов теста.	23
8. Рефакторинг тестируемого кода.....	25
9. Рефакторинг тестовых методов.....	27
10. Повторное тестирование, переписывание и анализ.	28
Заключение.	30

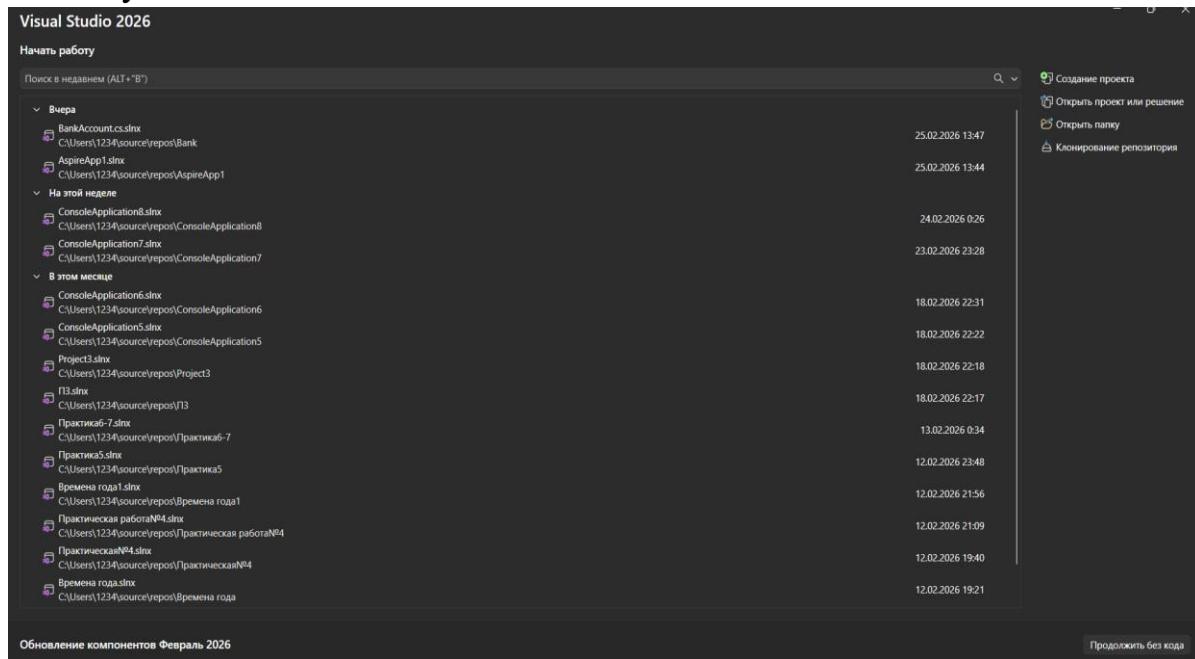
Введение.

Выполняя практическую работу №1 по теме «Средства тестирования Visual Studio-2022», я ознакомилась и изучила учебное пособие Средства тестирования Visual Studio.

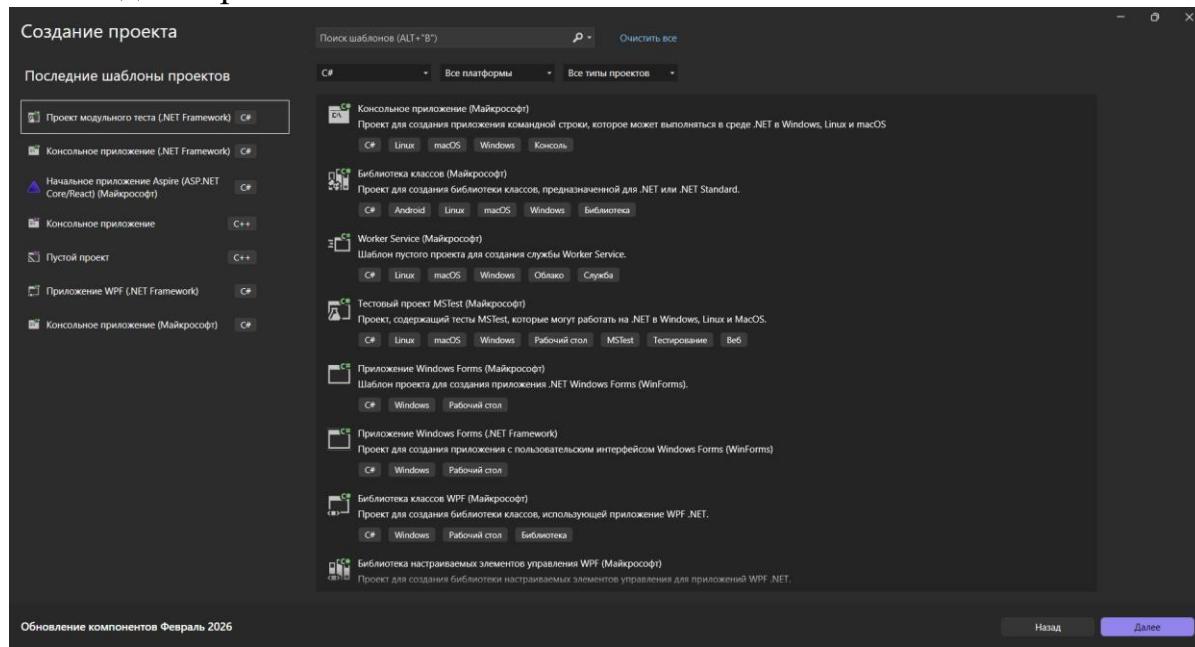
Практическая работа №1 была выполнена по руководству "Средства тестирования Visual Studio-2022", по стр. 158 -170.
(<https://cloud.mail.ru/public/JaXA/BUKbRzZoN>).

1. Создание проекта для тестирования.

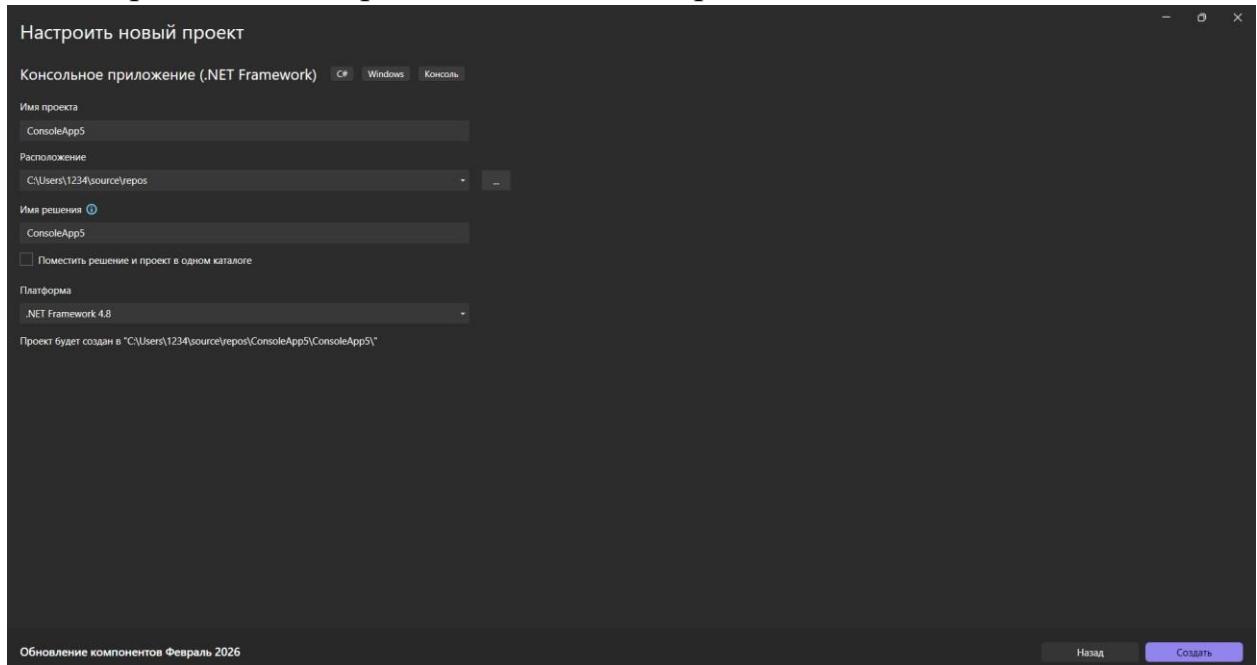
1. Я запустила Visual Studio.



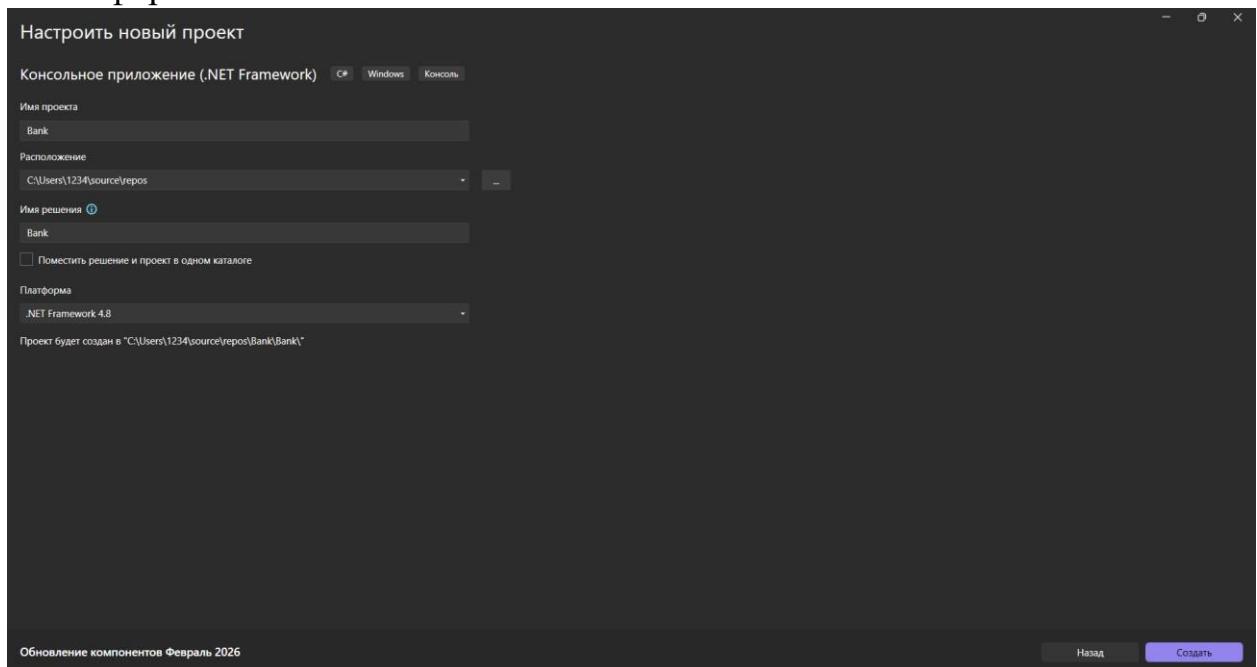
2. Я создала проект.



3. Я выбрала шаблон проекта Консольное приложение на C# для .NET Core.



4. Я назвала проект Bank и выбрали рекомендуемую версию целевой платформы.



5. Я заменила содержимое файла Program.cs следующими кодом на C#, который определяет класс BankAccount. Код:

```
using System;
namespace BankAccountNS
{
    /// <summary>
```

```

/// Bank account demo class.
/// </summary>
public class BankAccount
{
    private readonly string m_customerName;
    private double m_balance;      private
BankAccount() { }

    public BankAccount(string customerName, double balance)
    {
        m_customerName = customerName;
        m_balance = balance;
    }

    public string CustomerName
    {
        get { return m_customerName; }
    }

    public double Balance
    {
        get { return m_balance; }
    }

    public void Debit(double amount)
    {
        if (amount > m_balance)
        {
            throw new ArgumentOutOfRangeException("amount");
        }
        if (amount < 0)
        {
            throw new ArgumentOutOfRangeException("amount");
        }
        m_balance += amount; // intentionally incorrect code
    }

    public void Credit(double amount)
    {
        if (amount < 0)
        {
            throw new ArgumentOutOfRangeException("amount");
        }
        m_balance += amount;
    }

    public static void Main()

```

```

    {
        BankAccount ba = new BankAccount("Mr. Bryan Walton",
            11.99);
        ba.Credit(5.77);
        ba.Debit(11.22);
        Console.WriteLine("Current
balance is ${0}", ba.Balance);
    }
}

```

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

- File Menu:** Файл, Правка, Вид, Git, Проект, Сборка, Опладка, Тест, Средства, Расширения, Окно, Справка.
- Toolbar:** Contains icons for opening files, saving, running, and debugging.
- Status Bar:** Стр: 53, Симв: 6, Пробелы, CRLF, UTF-8 with BOM.

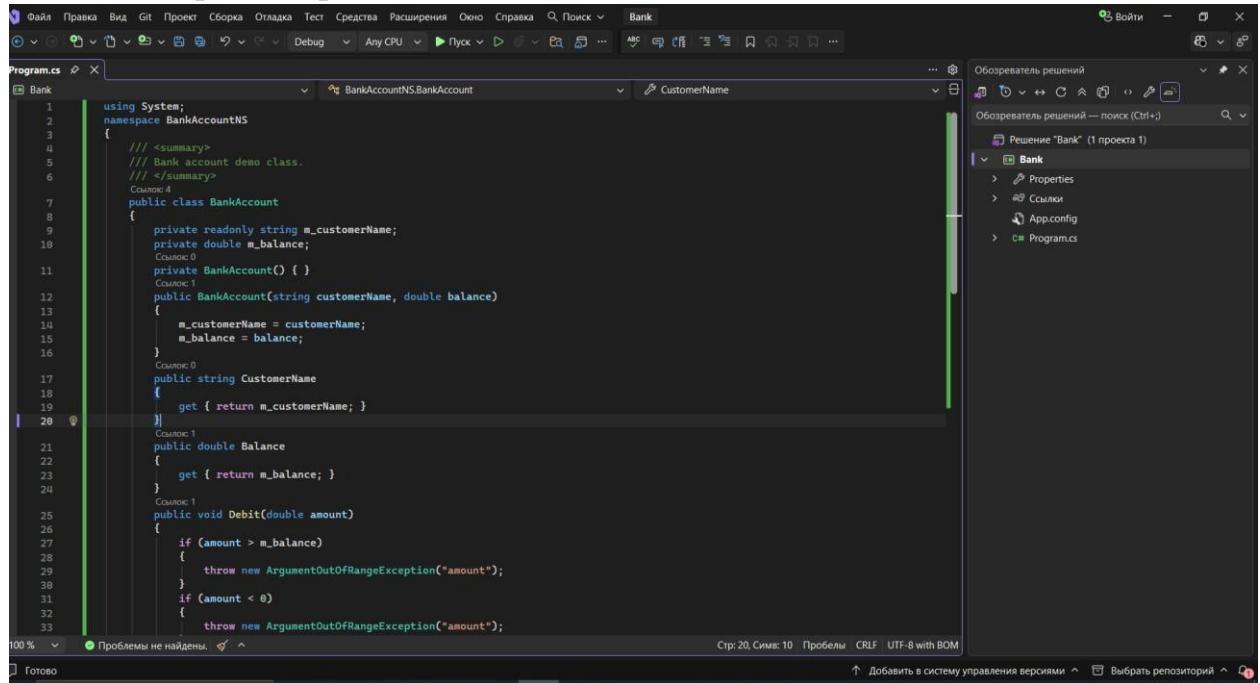
```

using System;
namespace BankAccountNS
{
    /// <summary>
    /// Bank account demo class.
    /// </summary>
    public class BankAccount
    {
        private readonly string _customerName;
        private double _balance;
        public BankAccount()
        {
            _customerName = "Customer";
            _balance = 0;
        }
        public BankAccount(string customerName, double balance)
        {
            _customerName = customerName;
            _balance = balance;
        }
        public string CustomerName
        {
            get { return _customerName; }
        }
        public double Balance
        {
            get { return _balance; }
        }
        public void Debit(double amount)
        {
            if (amount > _balance)
            {
                throw new ArgumentOutOfRangeException("amount");
            }
            if (amount < 0)
            {
                throw new ArgumentOutOfRangeException("amount");
            }
            _balance -= amount;
        }
        public void Credit(double amount)
        {
            _balance += amount;
        }
    }
}

```

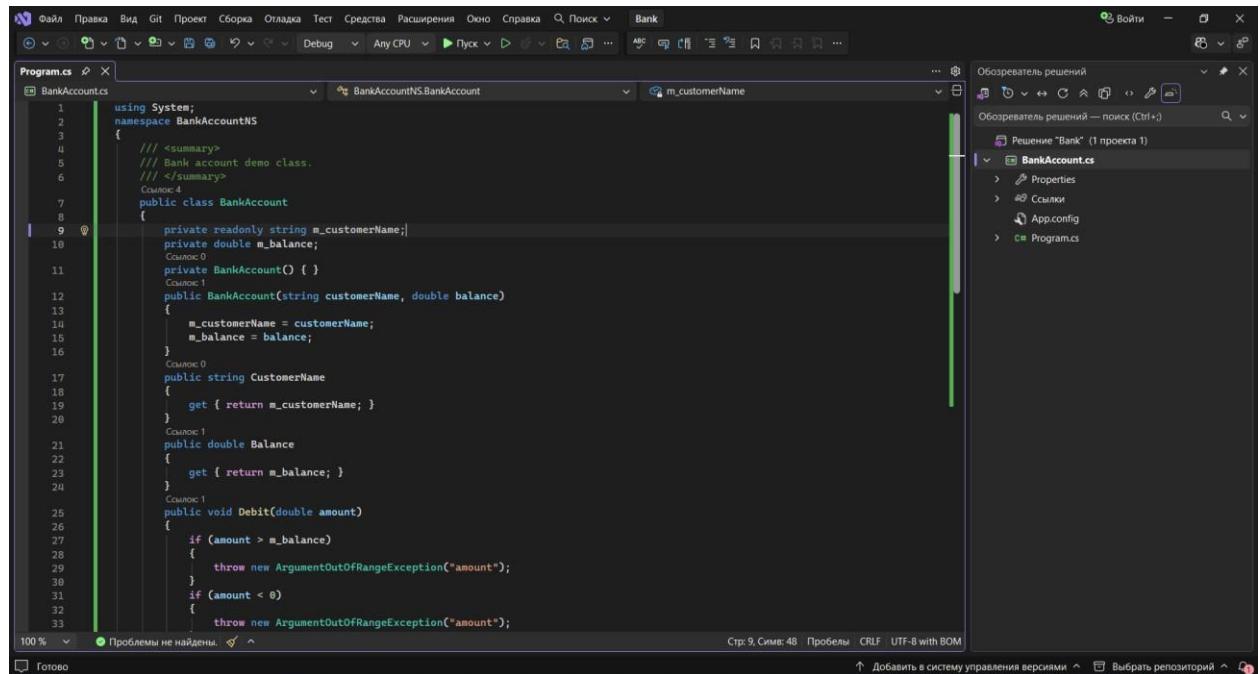
6. Я переименовала файл в BankAccount.cs.

Для этого я зашла в Обозреватель решений, нажала на Bank правой кнопки мыши и выбрала Переименовать.



```
Файл Правка Вид Git Проект Сборка Отладка Тест Средства Расширения Окно Справка ⌂ Поиск ... Bank

Program.cs < x | Bank
1 using System;
2 namespace BankAccountNS
3 {
4     /// <summary>
5     /// Bank account demo class.
6     /// </summary>
Ссылка 4
7     public class BankAccount
8     {
9         private readonly string m_customerName;
10        private double m_balance;
Ссылка 0
11        private BankAccount() { }
Ссылка 1
12        public BankAccount(string customerName, double balance)
13        {
14            m_customerName = customerName;
15            m_balance = balance;
16        }
Ссылка 0
17        public string CustomerName
18        {
19            get { return m_customerName; }
20        }
Ссылка 1
21        public double Balance
22        {
23            get { return m_balance; }
24        }
Ссылка 1
25        public void Debit(double amount)
26        {
27            if (amount > m_balance)
28            {
29                throw new ArgumentOutOfRangeException("amount");
30            }
31            if (amount < 0)
32            {
33                throw new ArgumentOutOfRangeException("amount");
}
Стр: 20, Симв: 10 Проблемы CRLF UTF-8 with BOM
100% < Проблемы не найдены. ⌂ ^ Готово ↑ Добавить в систему управления версиями ^ Выбрать репозиторий ^
```



```
Файл Правка Вид Git Проект Сборка Отладка Тест Средства Расширения Окно Справка ⌂ Поиск ... Bank

BankAccount.cs < x | Bank
1 using System;
2 namespace BankAccountNS
3 {
4     /// <summary>
5     /// Bank account demo class.
6     /// </summary>
Ссылка 4
7     public class BankAccount
8     {
9         private readonly string m_customerName;
10        private double m_balance;
Ссылка 0
11        private BankAccount() { }
Ссылка 1
12        public BankAccount(string customerName, double balance)
13        {
14            m_customerName = customerName;
15            m_balance = balance;
16        }
Ссылка 0
17        public string CustomerName
18        {
19            get { return m_customerName; }
20        }
Ссылка 1
21        public double Balance
22        {
23            get { return m_balance; }
24        }
Ссылка 1
25        public void Debit(double amount)
26        {
27            if (amount > m_balance)
28            {
29                throw new ArgumentOutOfRangeException("amount");
30            }
31            if (amount < 0)
32            {
33                throw new ArgumentOutOfRangeException("amount");
}
Стр: 9, Симв: 48 Проблемы CRLF UTF-8 with BOM
100% < Проблемы не найдены. ⌂ ^ Готово ↑ Добавить в систему управления версиями ^ Выбрать репозиторий ^
```

7. Использовала клавиши CTRL +SHIFT + В для того, чтобы произошла сборка (вывод был).

The screenshot shows the Visual Studio IDE interface. On the left, the code editor displays `BankAccount.cs` with the following content:

```
1  using System;
2  namespace BankAccountNS
3  {
4      /// <summary>
5      /// Bank account demo class.
6      /// </summary>
7      public class BankAccount
8      {
9          private readonly string m_customerName;
10         private double m_balance;
11         private BankAccount() { }
12         public BankAccount(string customerName, double balance)
13         {
14             m_customerName = customerName;
15             m_balance = balance;
16         }
17         public string CustomerName
18         {
19             get { return m_customerName; }
20         }
21         public double Balance
22         {
23             get { return m_balance; }
24         }
25     }
```

The right side of the interface includes the Solution Explorer showing the project structure:

- Решение "Bank" (1 проекта)
- BankAccount.cs
- Properties
- Ссылки
- App.config
- Program.cs

At the bottom, the Output window displays the build logs:

```
Показать выходные данные из: Сборка
Сборка началась в 15:46...
1>----- Сборка началась: проект: BankAccount.cs, Конфигурация: Debug Any CPU -----
1> BankAccount.cs -> C:\Users\1234\source\repos\Bank\Bank\bin\Debug\Bank.exe
===== Сборка: успешно выполнено - 1 , со сбоями - 0, в актуальном состоянии - 0, пропущено - 0 =====
===== Сборка завершена в 15:46 и заняло 01,278 с =====
```

Результат работы:

```
using System;
namespace BankAccountNS
{
    /// <summary>
    /// Bank account demo class.
    /// </summary>
    [ComVisible(4)]
    public class BankAccount
    {
        private readonly string _customerName;
        private double _balance;
        public BankAccount()
        {
            _customerName = "Customer";
            _balance = 0;
        }
        public BankAccount(string customerName, double balance)
        {
            _customerName = customerName;
            _balance = balance;
        }
        public string CustomerName
        {
            get { return _customerName; }
        }
        public double Balance
        {
            get { return _balance; }
        }
    }
}
```

Вывод

Показать выходные данные из: Сборка

Сборка началась в 14:24...

Сборка: успешно выполнено - 0 , со сбоями - 0, в актуальном состоянии - 3, пропущено - 0

Сборка завершена в 14:24 и заняло 00.172 с

```
        if (amount < 0)
        {
            throw new ArgumentOutOfRangeException("amount");
        }
        if (amount < 0)
        {
            throw new ArgumentOutOfRangeException("amount");
        }
        _balance += amount; // intentionally incorrect code
    }
    [ComVisible(1)]
    public void Credit(double amount)
    {
        if (amount < 0)
        {
            throw new ArgumentOutOfRangeException("amount");
        }
        _balance += amount;
    }
    [ComVisible(0)]
    public static void Main()
    {
        BankAccount ba = new BankAccount("Mr. Bryan Walton",
            11.99);
        ba.Credit(5.77);
        ba.Debit(11.22);
        Console.WriteLine("Current balance is ${0}", ba.Balance);
    }
}
```

Консоль отладки Microsoft Visual Studio

Current balance is \$28,98

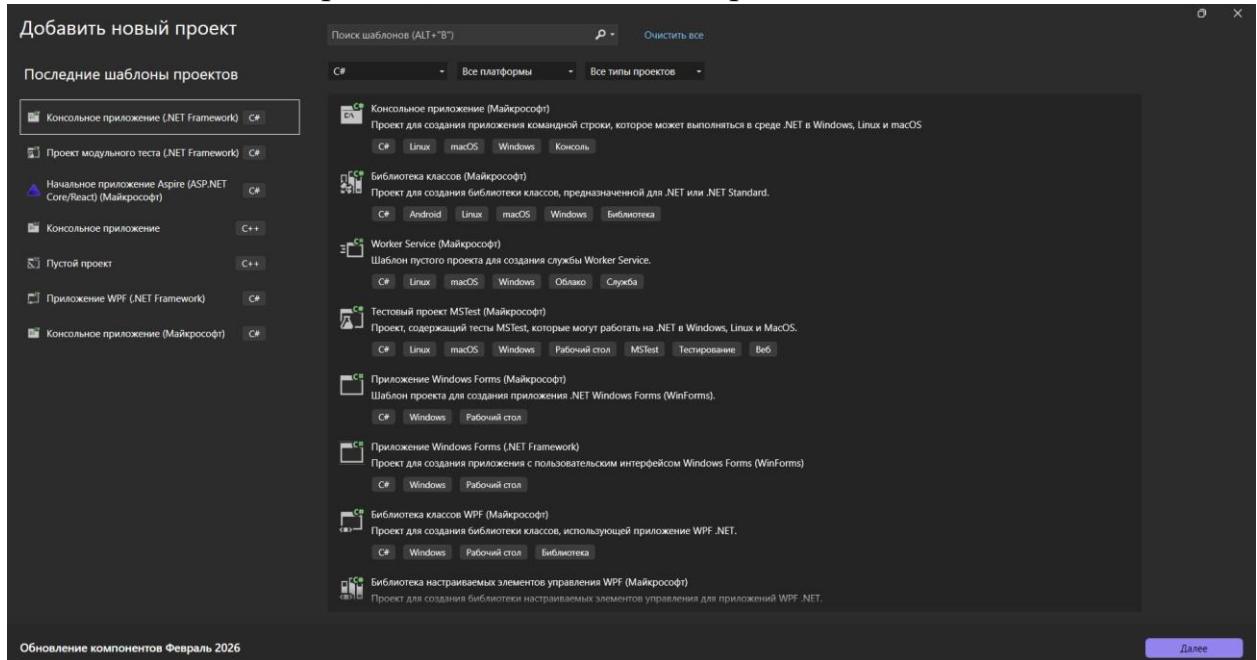
C:\Users\1234\source\repos\Bank\Bank\bin\Debug\Bank.exe (процесс 8892) завершил работу с кодом 0 (0x0).

Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка".

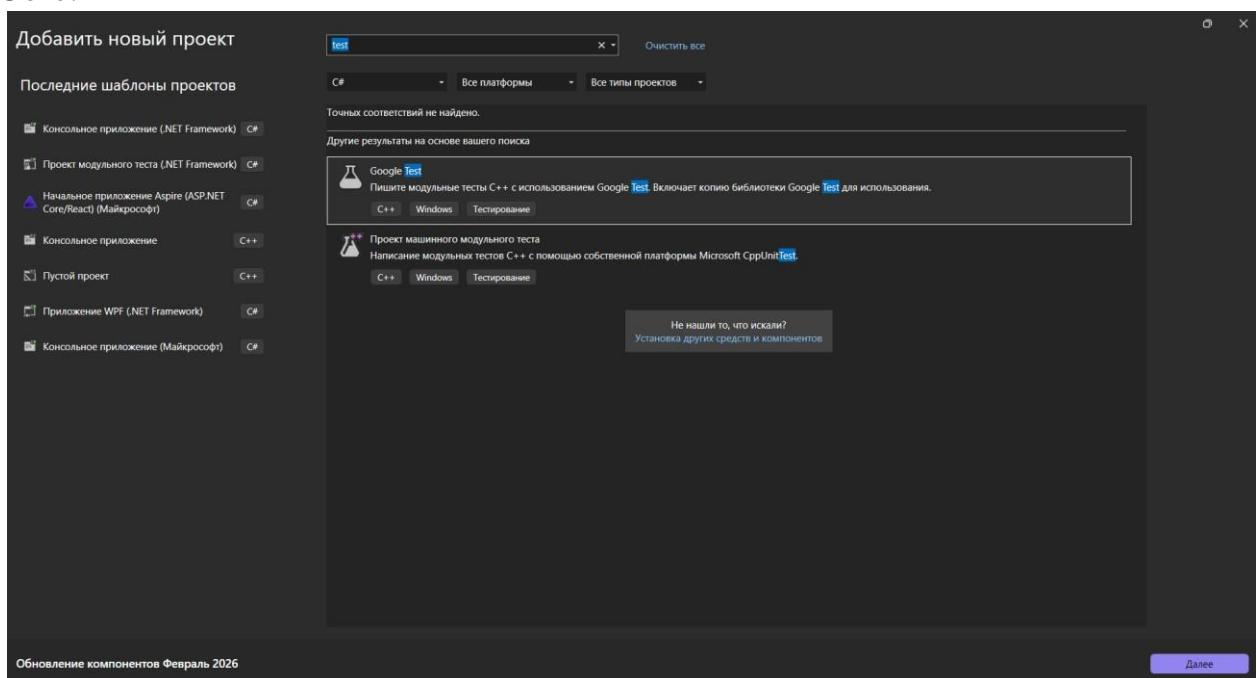
Нажмите любую клавишу, чтобы закрыть это окно.

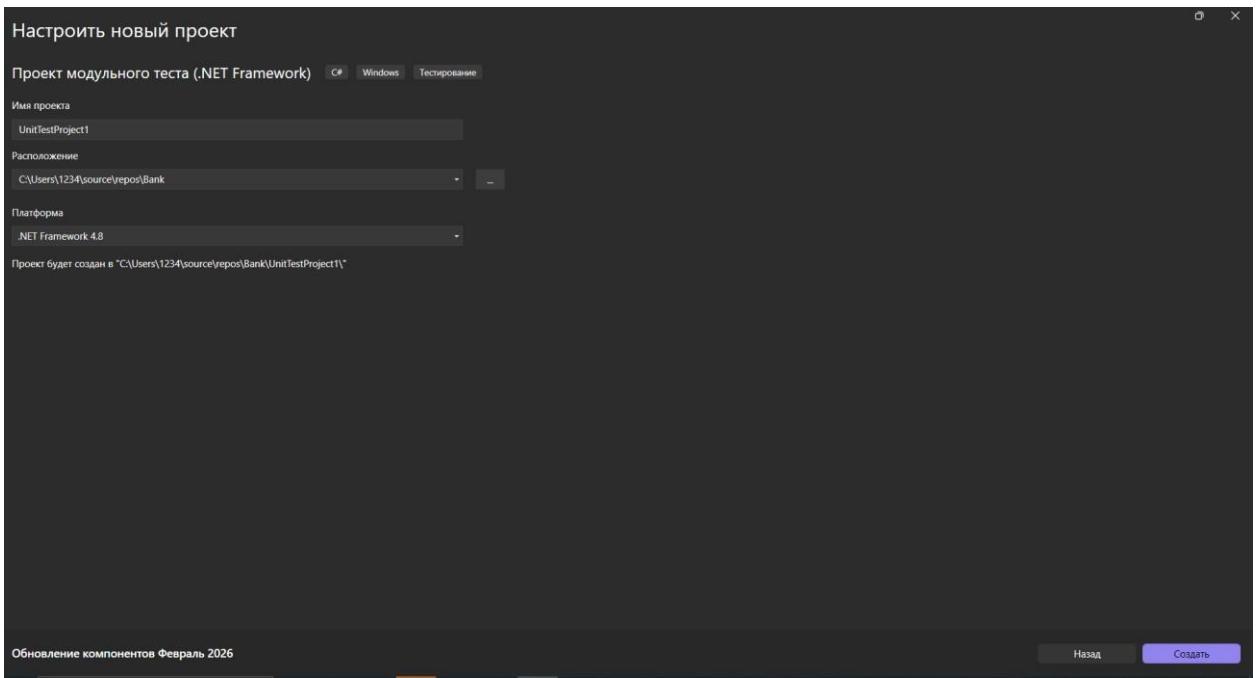
2. Создание проекта модульного теста.

1. В меню Файл выбрала Добавить>Создать проект.

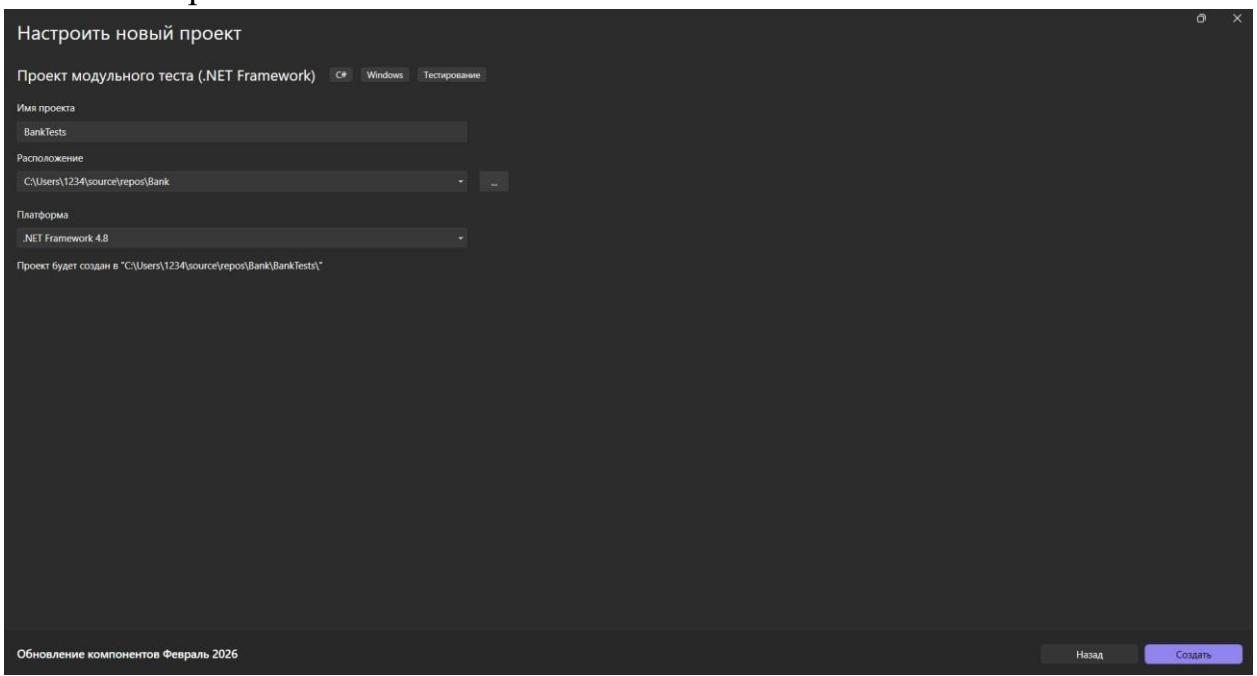


2. Ввела test в поле поиска, выбрала C# в качестве языка, затем выбрала Проект модульного теста MSTest (.NET Core) для C# в качестве шаблона .NET Core.

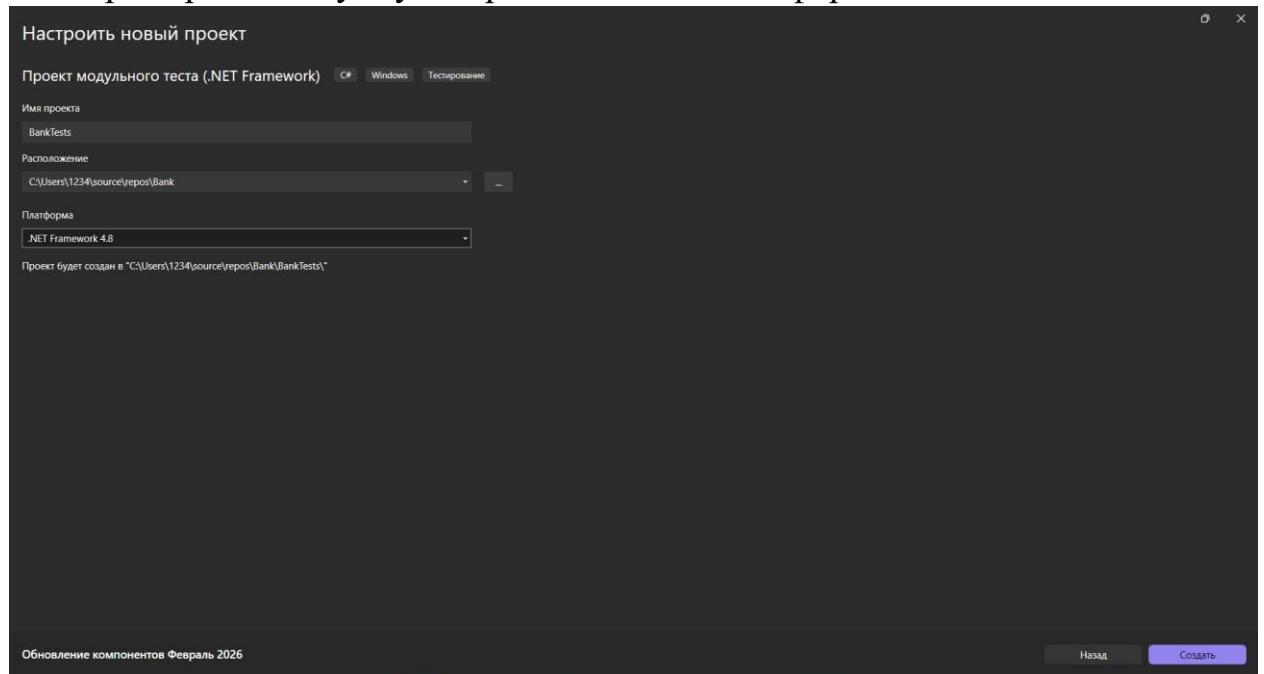




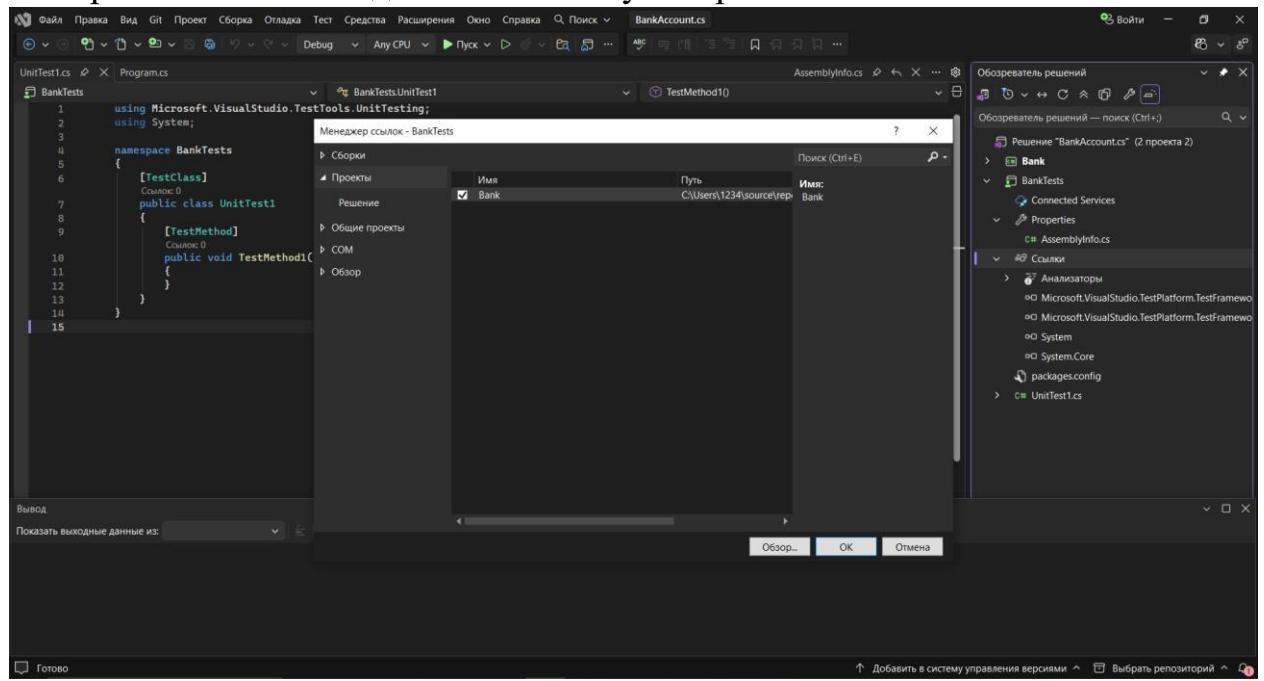
3. Назвала проект BankTests.



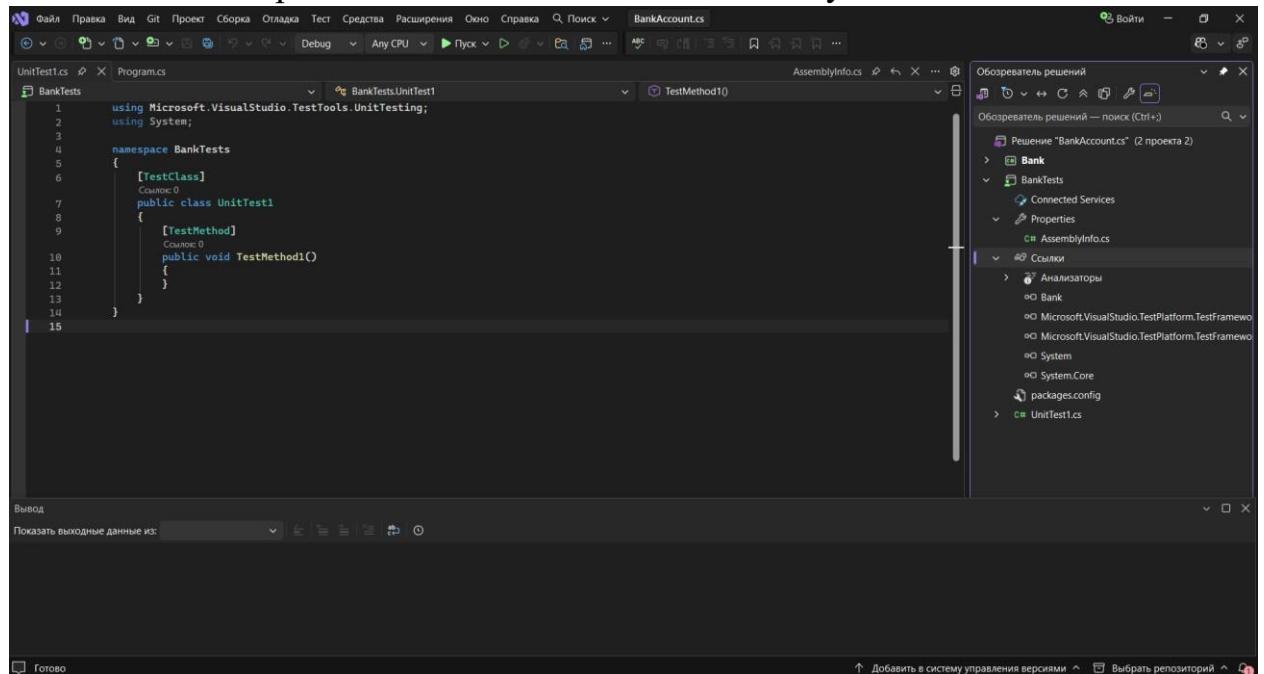
4. Выбрала рекомендуемую версию целевой платформы.



5. В проекте BankTests добавила ссылку на проект Банк.

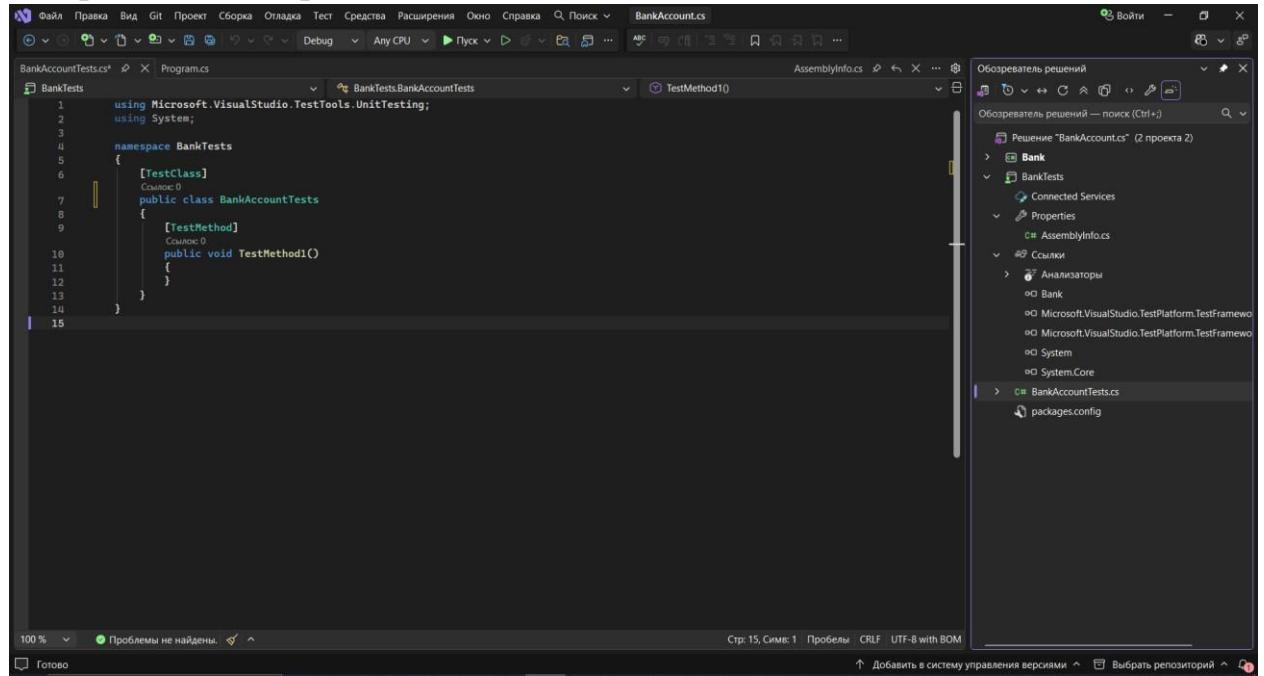


6. В диалоговом окне Диспетчер ссылок развернула Проекты, выбрала Решение и выбрала элемент Банк и нажала кнопку ОК.

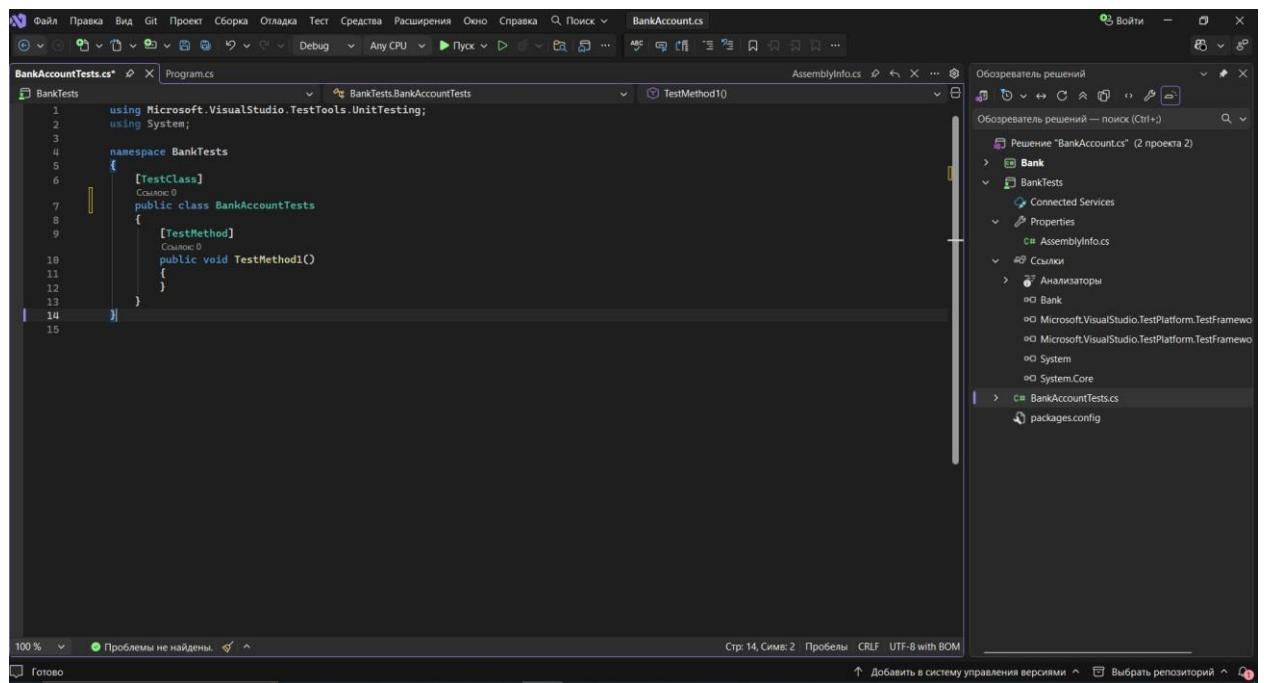


3. Создание тестового класса.

1. Переименовала файл в BankAccountTests.cs.



The screenshot shows the Visual Studio IDE interface. The left pane displays the code for `BankAccountTests.cs`:1 using Microsoft.VisualStudio.TestTools.UnitTesting;
2 using System;
3
4 namespace BankTests
5 {
6 [TestClass]
7 public class BankAccountTests
8 {
9 [TestMethod]
10 public void TestMethod1()
11 {
12 }
13 }
14}
15The right pane shows the Solution Explorer with the project structure:Решение "BankAccount.cs" (2 проекта)
> Bank
 > BankTests
 > Connected Services
 > Properties
 > AssemblyInfo.cs
 > Ссылки
 > Анализаторы
 > Bank
 > Microsoft.VisualStudio.TestTools.TestPlatform.TestFramework
 > Microsoft.VisualStudio.TestTools.TestPlatform.TestFramework
 > System
 > System.Core
> BankAccountTests.cs
 > packages.config



The screenshot shows the Visual Studio IDE interface. The left pane displays the code for `BankAccountTests.cs` (note the file name has been changed from the previous screenshot):1 using Microsoft.VisualStudio.TestTools.UnitTesting;
2 using System;
3
4 namespace BankTests
5 {
6 [TestClass]
7 public class BankAccountTests
8 {
9 [TestMethod]
10 public void TestMethod1()
11 {
12 }
13 }
14}
15The right pane shows the Solution Explorer with the project structure:Решение "BankAccount.cs" (2 проекта)
> Bank
 > BankTests
 > Connected Services
 > Properties
 > AssemblyInfo.cs
 > Ссылки
 > Анализаторы
 > Bank
 > Microsoft.VisualStudio.TestTools.TestPlatform.TestFramework
 > Microsoft.VisualStudio.TestTools.TestPlatform.TestFramework
 > System
 > System.Core
> BankAccountTests.cs
 > packages.config

2. Добавила оператор using.

The screenshot shows the Visual Studio IDE interface. On the left, there are two code editors: one for 'BankAccountTests.cs*' and another for 'Program.cs'. The 'BankAccountTests.cs*' editor contains the following C# code:

```
1  using Microsoft.VisualStudio.TestTools.UnitTesting;
2  using System;
3  using BankAccountNS;
4
5  namespace BankTests
6  {
7      [TestClass]
8      public class BankAccountTests
9      {
10          [TestMethod]
11          public void TestMethod1()
12          {
13          }
14      }
15  }
```

The 'Program.cs' editor is currently empty. On the right side of the interface, the 'Solution Explorer' window is open, showing the project structure for 'Bank'. The solution contains two projects: 'Bank' and 'BankTests'. The 'Bank' project includes 'AssemblyInfo.cs', 'Properties', 'Ссылки' (References), 'Анализаторы' (Analyzers), and 'System'. The 'BankTests' project includes 'BankAccountTests.cs' and 'packages.config'. The status bar at the bottom of the screen displays the message 'Проблемы не найдены.' (No problems found.)

4. Создание метода теста.

1. Ввела код.

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System;
using BankAccountNS;

namespace BankTests
{
    [TestClass]
    public class BankAccountTests
    {
        [TestMethod]
        public void Debit_WithValidAmount_UpdatesBalance()
        {
            // Arrange
            double beginningBalance = 11.99;
            double debitAmount = 4.55;
            double expected = 7.44;
            BankAccount account = new BankAccount("Mr. Bryan Walton",
beginningBalance);

            // Act
            account.Debit(debitAmount);

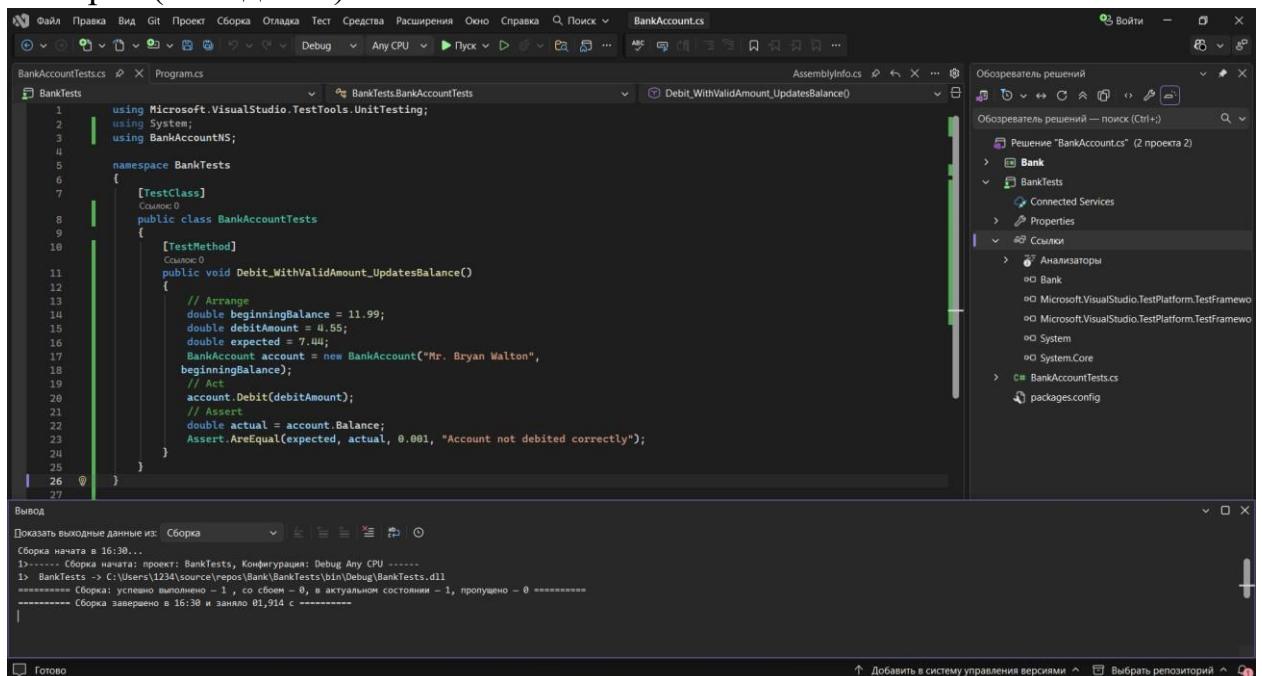
            // Assert
            double actual = account.Balance;
            Assert.AreEqual(expected, actual, 0.001, "Account not debited correctly");
        }
    }
}
```

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

- Menu Bar:** Файл, Правка, Вид, Git, Проект, Сборка, Отладка, Тест, Средства, Расширения, Окно, Справка, Поиск.
- Toolbars:** Debug, Any CPU, Пуск.
- Code Editor:** BankAccountTests.cs (highlighted), Program.cs, AssemblyInfo.cs.
- Solution Explorer:** Обозреватель решений (Solution Explorer). It shows the solution "BankAccount.cs" (2 projects) with the following structure:
 - Bank
 - BankTests
 - Connected Services
 - Properties
 - Ссылки
 - Анализаторы
 - Bank
 - Microsoft.VisualStudio.TestTools.UnitTesting
 - Microsoft.VisualStudio.TestTools.UnitTesting.TestFramework
 - System
 - System.Core
 - BankAccountTests.cs
 - packages.config
- Status Bar:** 100%, Проблемы не найдены, Стр. 26 Симв. 3 Проблемы CRLF UTF-8 with BOM, Добавить в систему управления версиями, Выбрать репозиторий.

5. Сборка и запуск теста.

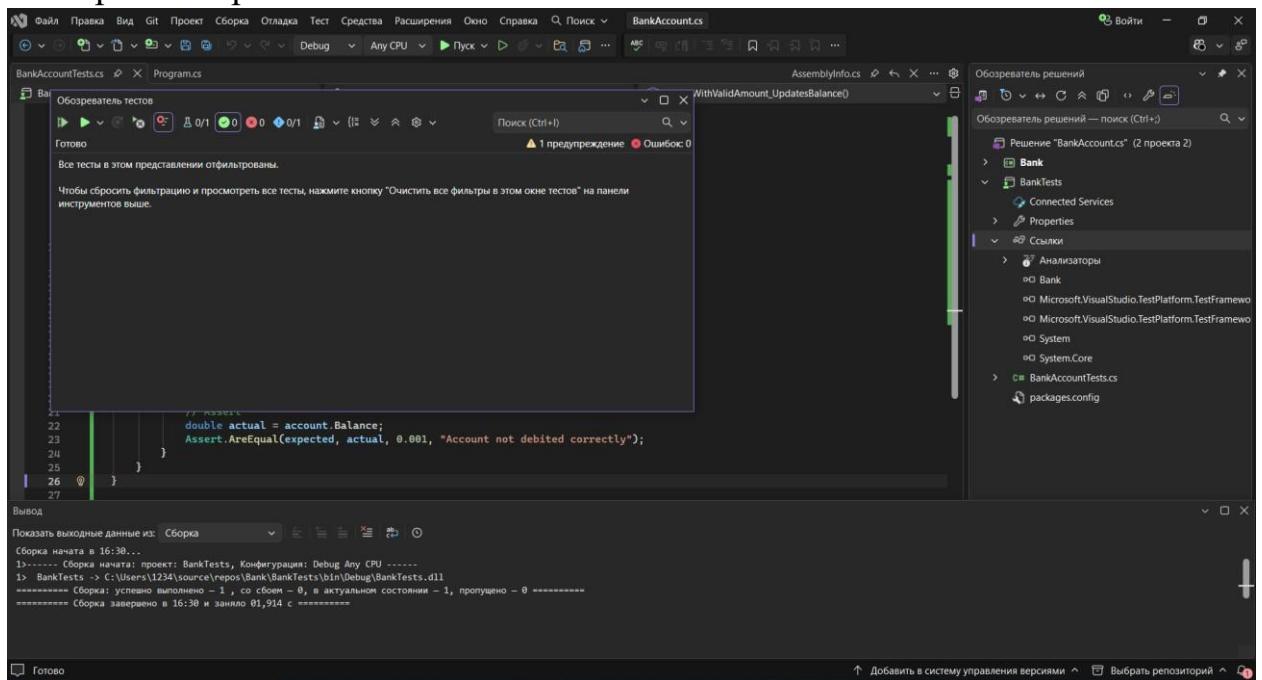
1. Использовала клавиши CTRL +SHIFT + B для того, чтобы произошла сборка (вывод был).



The screenshot shows the Visual Studio interface with the 'BankTests' project selected. The code editor displays a test class 'BankAccountTests' with a single test method 'Debit_WithValidAmount_UpdatesBalance'. The 'Output' window at the bottom shows the build log:

```
Показать выходные данные из: Сборка
Сборка началась в 16:38...
1>----- Сборка начата: проект: BankTests, Конфигурация: Debug Any CPU -----
1> BankTests -> C:\Users\1234\source\repos\Bank\BankTests\bin\Debug\BankTests.dll
===== Сборка: успешно выполнено - 1 , со ошибкой - 0 , в актуальном состоянии - 1, пропущено - 0 ======
===== Сборка завершена в 16:38 и заняло 01,914 с ======
```

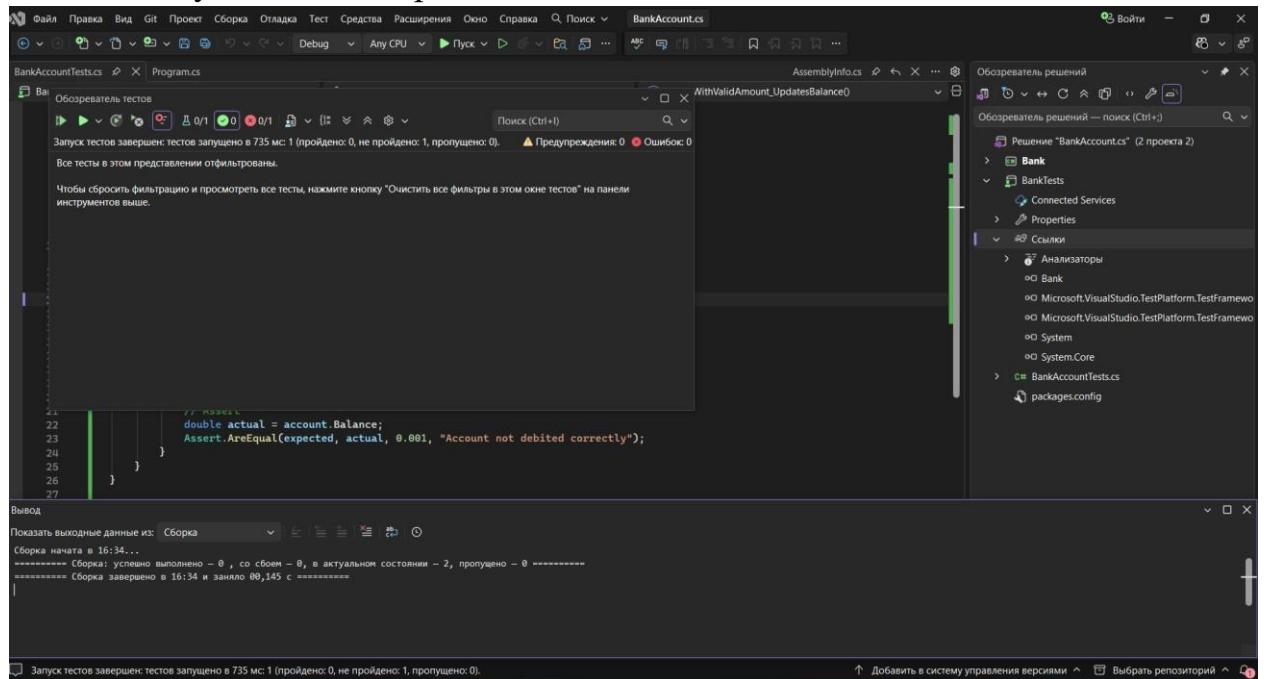
2. Открыла Обозреватель тестов, выбрав Тест>Windows>Обозреватель тестов в верхней строке меню.



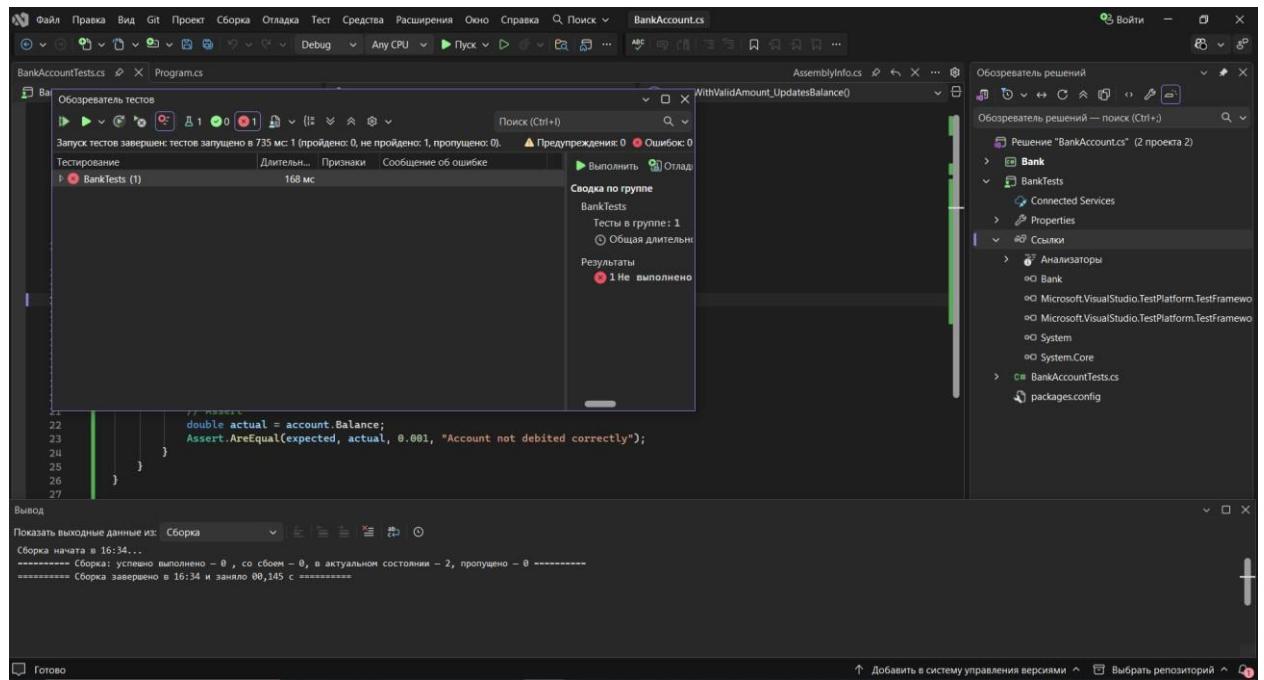
The screenshot shows the Visual Studio interface with the 'Test Explorer' window open. The 'BankTests' project is selected in the 'Solution Explorer' on the right. The 'Test Explorer' window displays the test results for the 'BankAccountTests' class, showing 1 test run and 1 warning. The code editor below shows the same test code as in the previous screenshot. The 'Output' window at the bottom shows the build log:

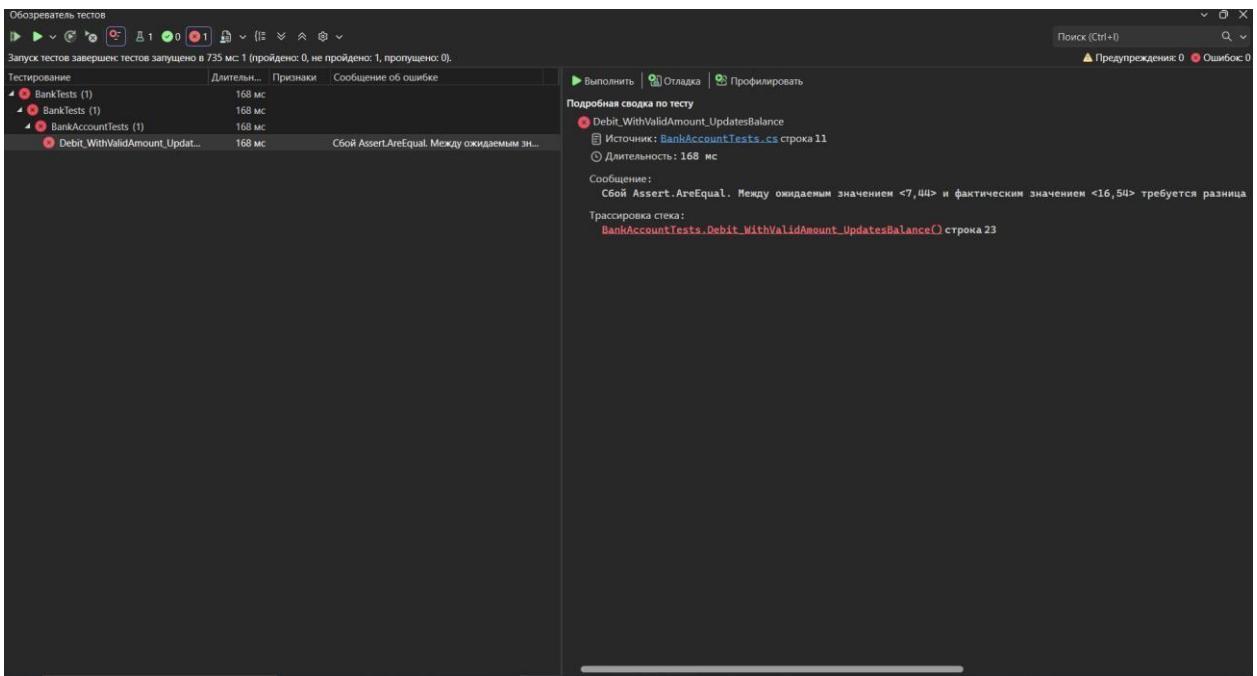
```
Показать выходные данные из: Сборка
Сборка началась в 16:38...
1>----- Сборка начата: проект: BankTests, Конфигурация: Debug Any CPU -----
1> BankTests -> C:\Users\1234\source\repos\Bank\BankTests\bin\Debug\BankTests.dll
===== Сборка: успешно выполнено - 1 , со ошибкой - 0 , в актуальном состоянии - 1, пропущено - 0 ======
===== Сборка завершена в 16:38 и заняло 01,914 с ======
```

3. Выбрала Запустить все, чтобы выполнить тест.
В данном случае тест не пройден.



4. Выбрала этот метод в обозревателе тестов для просмотра сведений в нижней части окна.

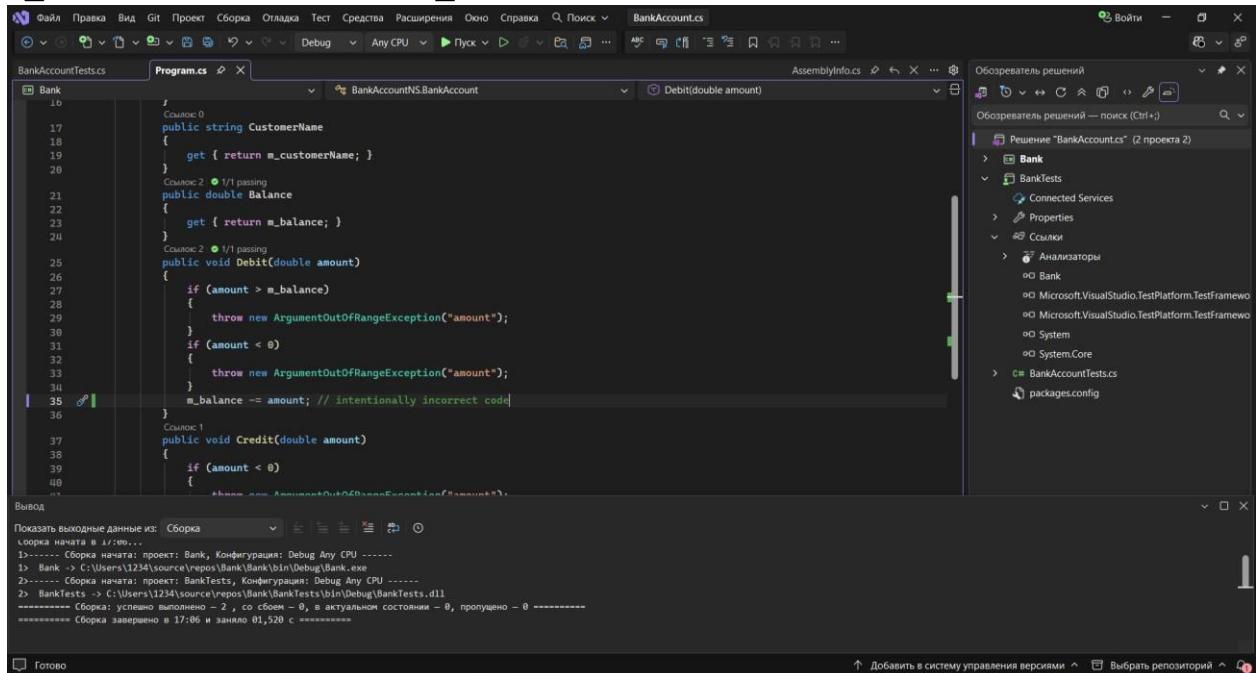




6. Исправление кода и повторный запуск тестов.

Исправление ошибки:

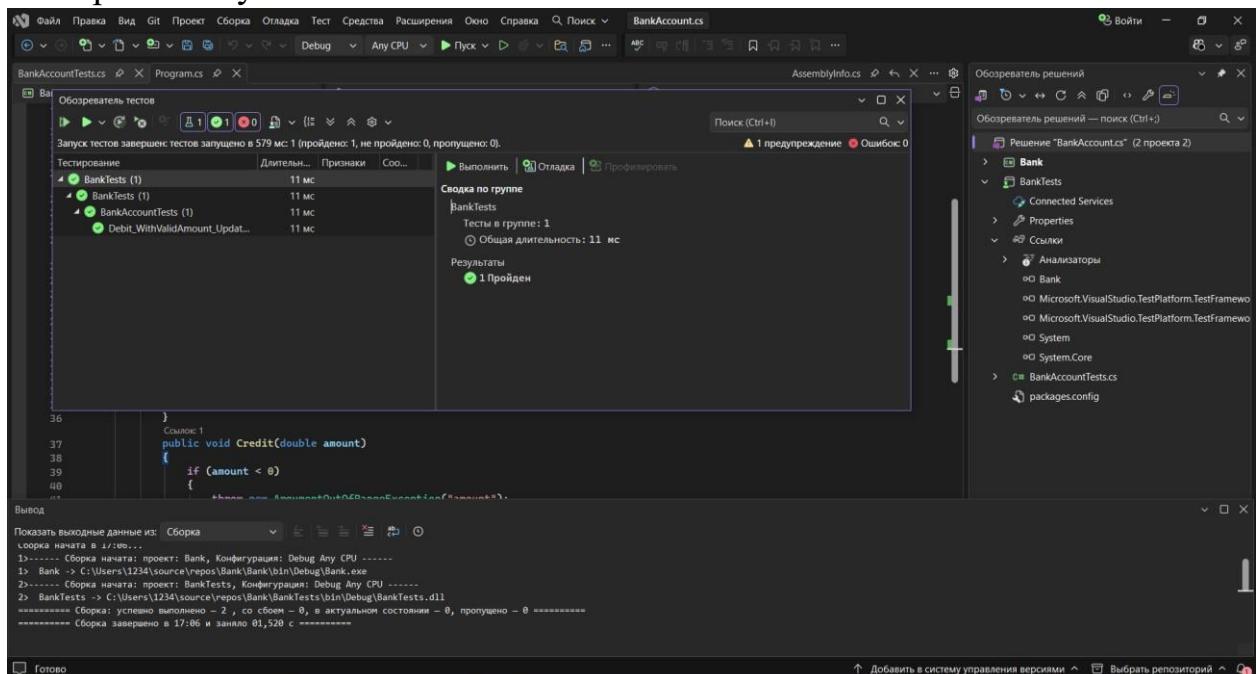
В файле BankAccount.cs была проведена замена строки с:
m_balance += amount; на: m_balance -= amount;



```
16     public string CustomerName
17     {
18         get { return _customerName; }
19     }
20
21     public double Balance
22     {
23         get { return m_balance; }
24     }
25
26     public void Debit(double amount)
27     {
28         if (amount > m_balance)
29         {
30             throw new ArgumentOutOfRangeException("amount");
31         }
32         if (amount < 0)
33         {
34             throw new ArgumentOutOfRangeException("amount");
35         }
36         m_balance -= amount; // intentionally incorrect code
37     }
38
39     public void Credit(double amount)
40     {
41         if (amount < 0)
42         {
43             throw new ArgumentOutOfRangeException("amount");
44         }
45     }

```

Повторный запуск теста.



Группа	Тесты	Статус
BankTests (1)	11 мс	Проведено
BankTests (1)	11 мс	Проведено
BankAccountTests (1)	11 мс	Проведено
Debit_WithValidAmount_Update...	11 мс	Проведено

7. Создание и запуск новых методов теста.

1. Ввод кода.

```
[TestMethod]
```

```
public void
```

```
Debit_WhenAmountIsLessThanZero_ShouldThrowArgumentOutOfRangeException()
```

```
{
```

```
    // Arrange
```

```
    double beginningBalance = 11.99;
```

```
    double debitAmount = -100.00;
```

```
    BankAccount account = new BankAccount("Mr. Bryan Walton",  
beginningBalance);
```

```
    // Act and assert
```

```
    Assert.ThrowsException<System.ArgumentOutOfRangeException>(() =>  
account.Debit(debitAmount));
```

```
}
```

The screenshot shows the Microsoft Visual Studio interface. The code editor window displays the file `BankAccountTests.cs` with the following content:

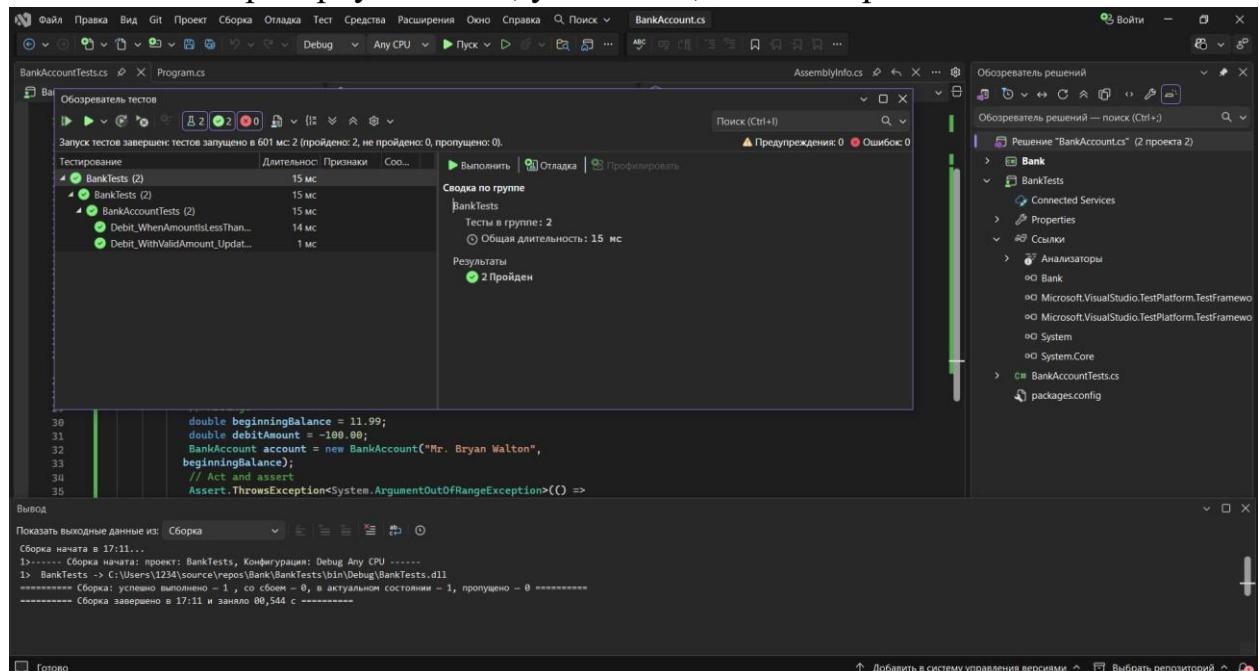
```
[TestClass]
public class BankAccountTests
{
    [TestMethod]
    public void Debit_WithValidAmount_UpdatesBalance()
    {
        // Arrange
        double beginningBalance = 11.99;
        double debitAmount = 4.55;
        double expected = 7.44;
        BankAccount account = new BankAccount("Mr. Bryan Walton",
beginningBalance);
        // Act
        account.Debit(debitAmount);
        // Assert
        double actual = account.Balance;
        Assert.AreEqual(expected, actual, 0.001, "Account not debited correctly");
    }

    [TestMethod]
    public void Debit_WhenAmountIsLessThanZero_ShouldThrowArgumentOutOfRangeException()
    {
        // Arrange
        double beginningBalance = 11.99;
        double debitAmount = -100.00;
        BankAccount account = new BankAccount("Mr. Bryan Walton",
beginningBalance);
        // Act and assert
        Assert.ThrowsException<System.ArgumentOutOfRangeException>(() =>
account.Debit(debitAmount));
    }
}
```

The Solution Explorer on the right shows the project structure:

- Решение "BankAccount.cs" (2 проекта)
- Bank
- BankTests
- Connected Services
- Properties
- Ссылки
- Анализаторы
- Bank
- Microsoft.VisualStudio.TestTools.TestPlatform.TestFramework
- System
- System.Core
- BankAccountTests.cs
- packages.config

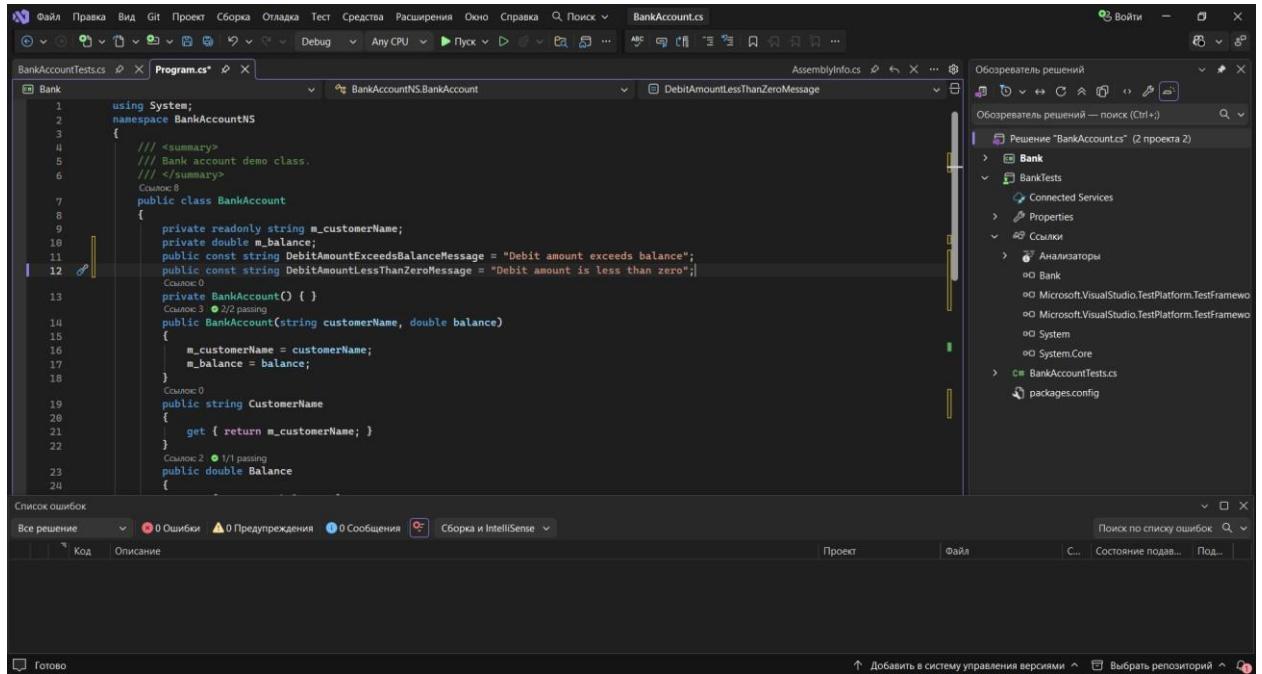
2. Выполнила проверку 2 тестов, убедилась, что они пройдены и нет ошибок.



8. Рефакторинг тестируемого кода.

1) Добавила это в тестируемый класс (BankAccount): `public const string DebitAmountExceedsBalanceMessage = "Debit amount exceeds balance";`

`public const string DebitAmountLessThanZeroMessage = "Debit amount is less than zero";`



2) Затем изменим два условных оператора в методе Debit : `if (amount > m_balance)`

```
{  
    throw new System.ArgumentOutOfRangeException("amount", amount,  
        DebitAmountExceedsBalanceMessage);  
}  
if (amount < 0)  
{  
    throw new System.ArgumentOutOfRangeException("amount", amount,  
        DebitAmountLessThanZeroMessage);  
}
```

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the code for `BankAccount.cs` in the `Bank` project. The code implements a bank account with methods for debiting and crediting, both of which validate the amount against the balance. The Solution Explorer on the right shows the `Bank` project containing `BankTests`, `Properties`, and `BankAccountTests.cs`. The Error List at the bottom indicates 0 errors, 0 warnings, and 0 messages.

```
19 public string CustomerName
20 {
21     get { return _customerName; }
22 }
23 // Ссылка 2: 1/1 passing
24 public double Balance
25 {
26     get { return _balance; }
27 }
28 // Ссылка 3: 2/2 passing
29 public void Debit(double amount)
30 {
31     if (amount > _balance)
32     {
33         throw new System.ArgumentOutOfRangeException("amount", amount,
34             DebitAmountExceedsBalanceMessage);
35     }
36     if (amount < 0)
37     {
38         throw new System.ArgumentOutOfRangeException("amount", amount,
39             DebitAmountLessThanZeroMessage);
40     }
41 }
42 public void Credit(double amount)
43 {
44     if (amount < 0)
45     {
46         throw new System.ArgumentOutOfRangeException("amount", amount,
47             CreditAmountLessThanZeroMessage);
48     }
49 }
```

9. Рефакторинг тестовых методов.

1. Ввод кода.

[TestMethod]

public void

Debit_WhenAmountIsMoreThanBalance_ShouldThrowArgumentOutOfRangeException()

{

// Arrange

double beginningBalance = 11.99;

double debitAmount = 20.0;

BankAccount account = new BankAccount("Mr. Bryan Walton",
beginningBalance);

// Act

try

{

account.Debit(debitAmount);

}

catch (System.ArgumentOutOfRangeException e)

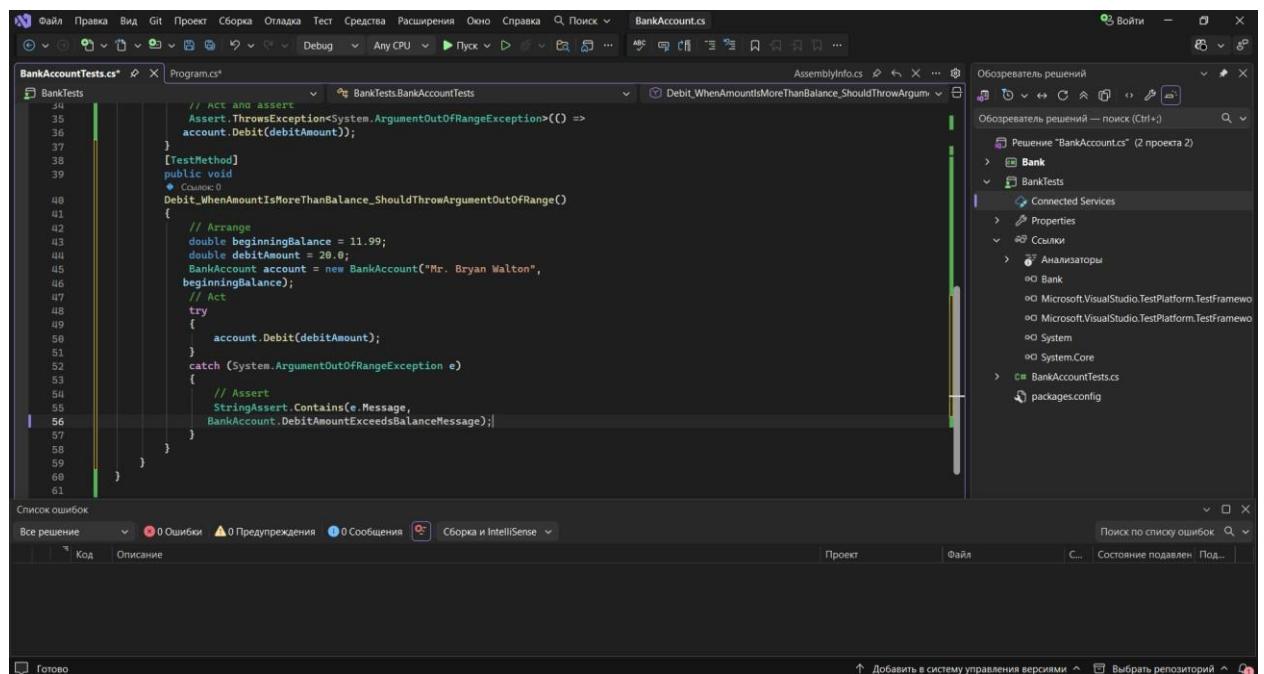
{

// Assert

StringAssert.Contains(e.Message,

BankAccount.DebitAmountExceedsBalanceMessage);

}



10. Повторное тестирование, переписывание и анализ.

1) Редактирование кода.

The screenshot shows the Microsoft Visual Studio interface. In the center, the code editor displays a file named `BankAccountTests.cs` containing C# test code. The code defines a test method `Debit_WhenAmountIsMoreThanBalance_ShouldThrowArgumentOutOfRangeException()` that attempts to debit an account with an amount greater than its balance and catches the resulting exception. The Solution Explorer on the right shows a project structure for "Bank" and "BankTests" with various files like `AssemblyInfo.cs`, `BankAccount.cs`, and `BankAccountTests.cs`.

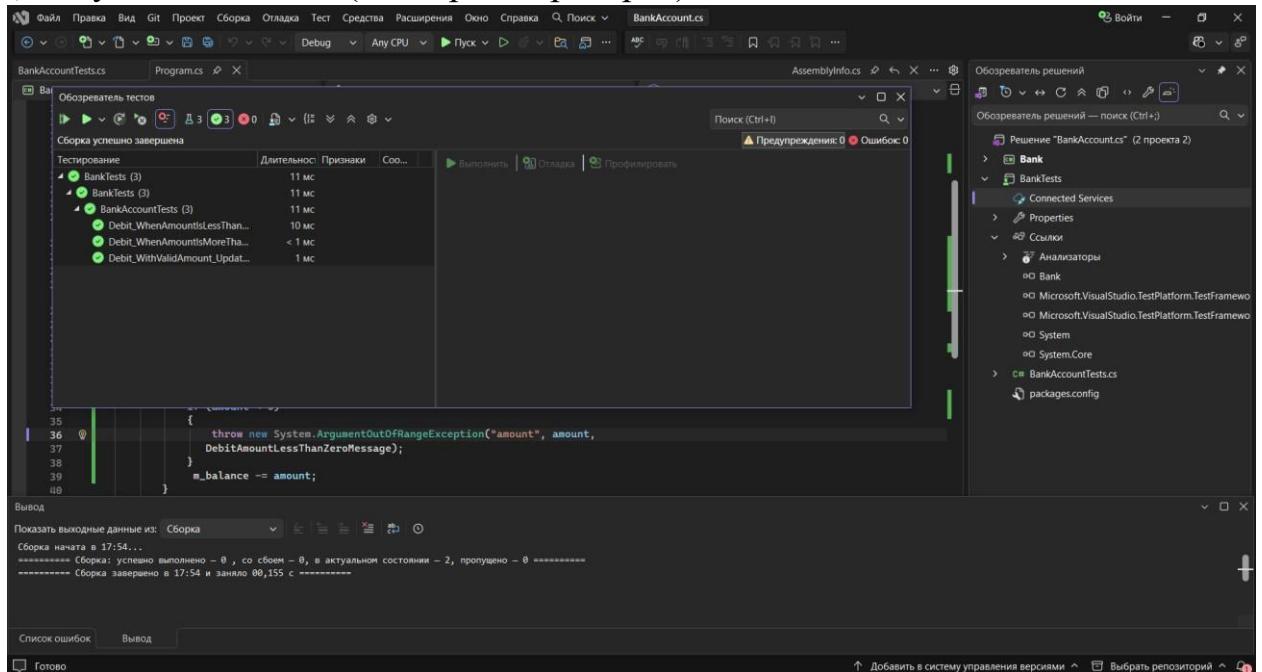
2) Проверка тестов.

Обнаружение ошибки в одном из тестов.

The screenshot shows the Test Explorer window in Visual Studio. It lists three tests: `BankTests (1)`, `BankTests (1)`, and `BankAccountTests (1)`. The third test, `Debit_WithValidAmount_UpdatesBalance`, has failed. A tooltip provides detailed information about the failure: "Сбой Assert.AreEqual. Между ожидаемым значением <7,44> и фактическим значением <11,99> требуется разница". The status bar at the bottom shows the date and time as 26.02.2026 17:47.

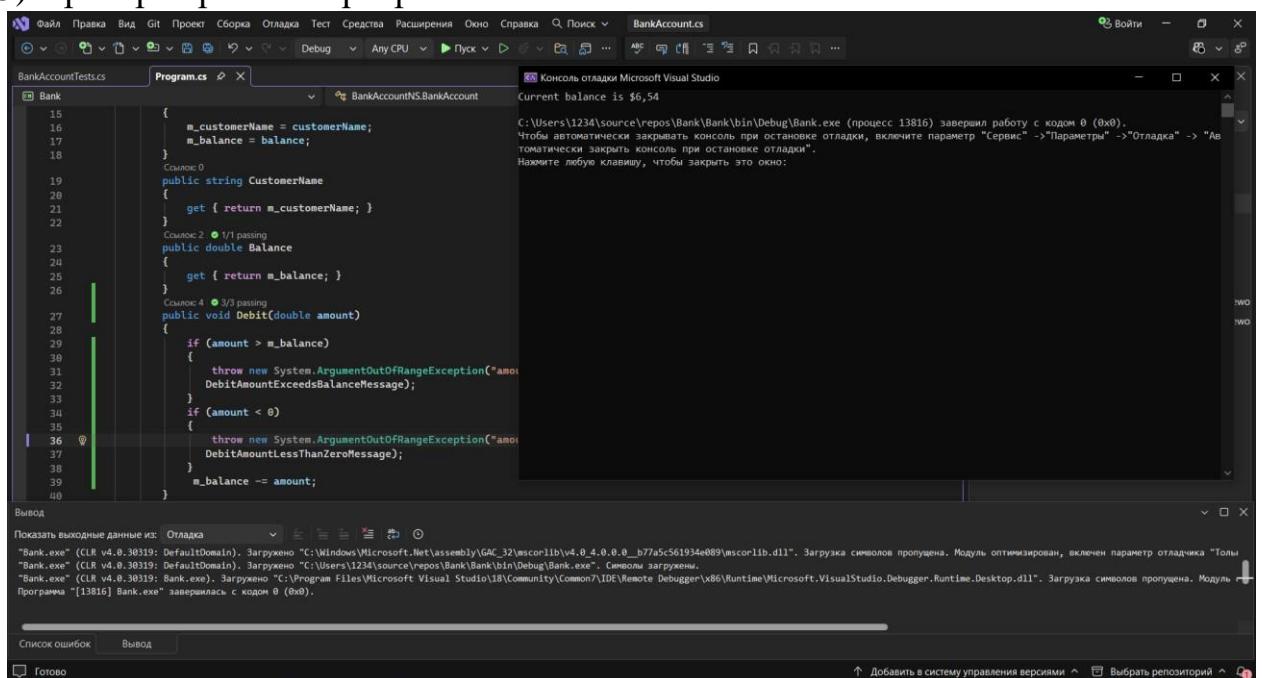
Добавление строки в код для устранения ошибки: `m_balance -= amount;`

3) Запуск всех тестов (повторная проверка).



4) Анализирование вывода, ошибка устранена. Все тесты выполнены правильно.

5) Проверка работы программы.



Заключение.

Выполняя практическую работу №1 по теме «Средства тестирования Visual Studio-2022», я освоила работу со средой тестирования. Научилась создавать и запускать модульные тесты, и выполнять рефакторинг. Таким образом, я смогла усовершенствовать тестовый код, что привело к созданию более надёжного информативного метода теста. Но самое главное, что в результате был улучшен тестируемый код.

