**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.**

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.

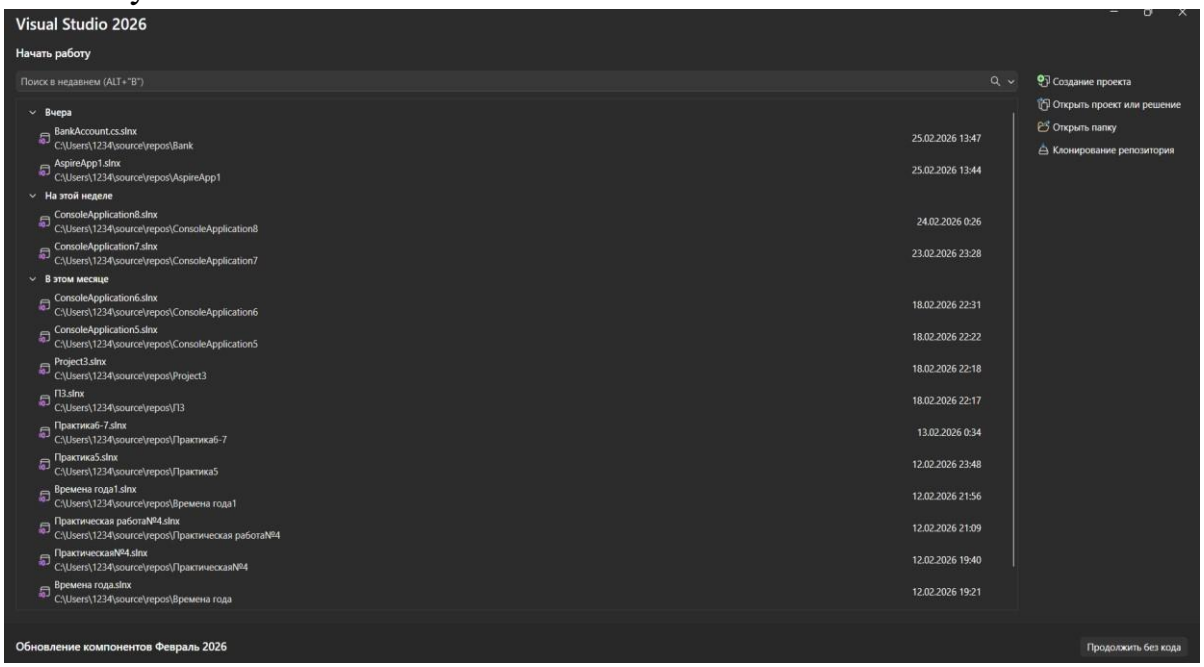


Рисунок 1

2. Я создала проект.

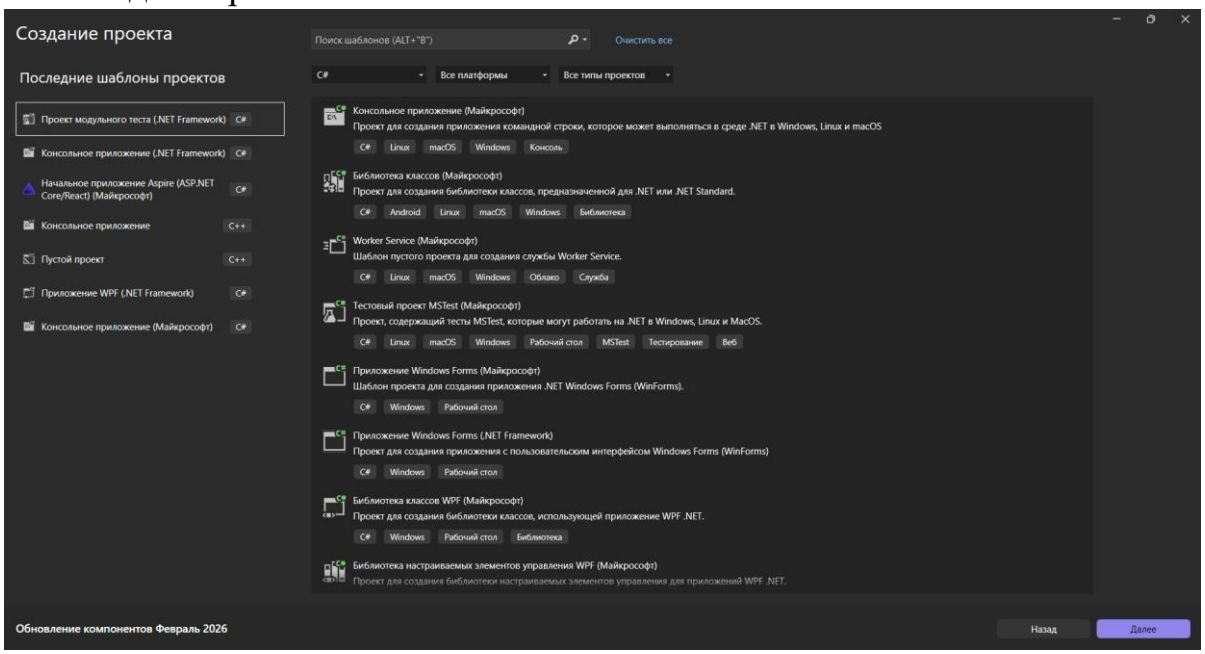


Рисунок 2

3. Я выбрала шаблон проекта Консольное приложение на C#.

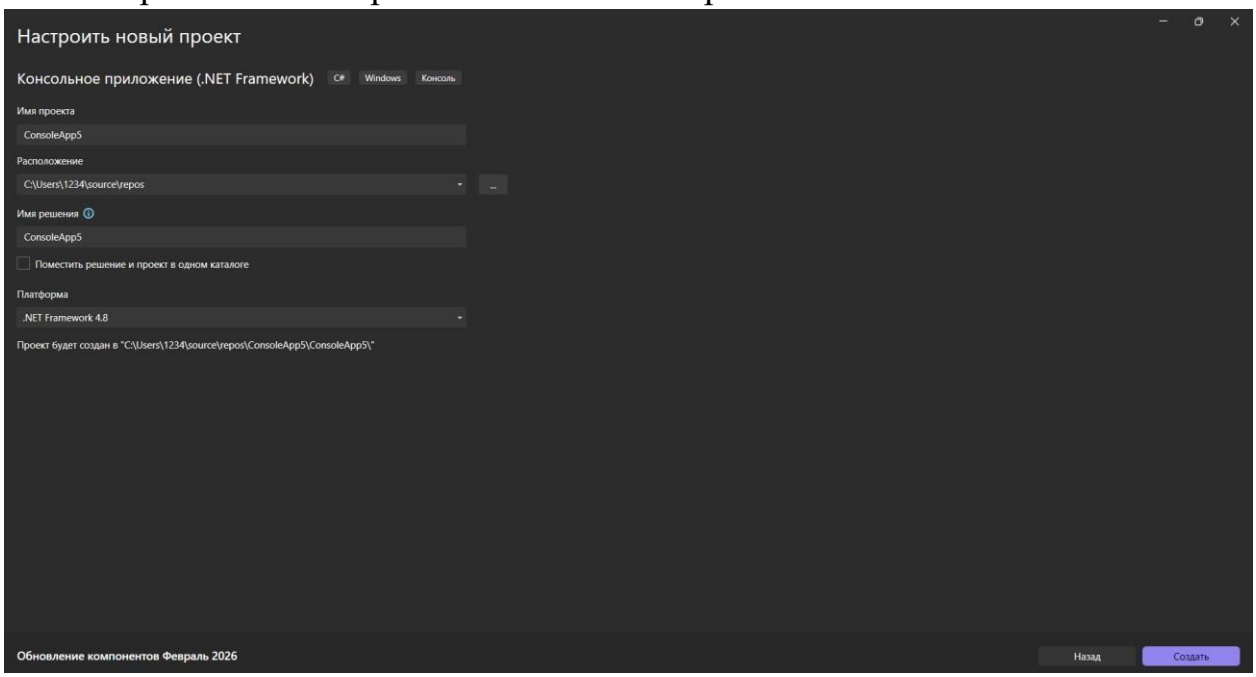


Рисунок 3

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

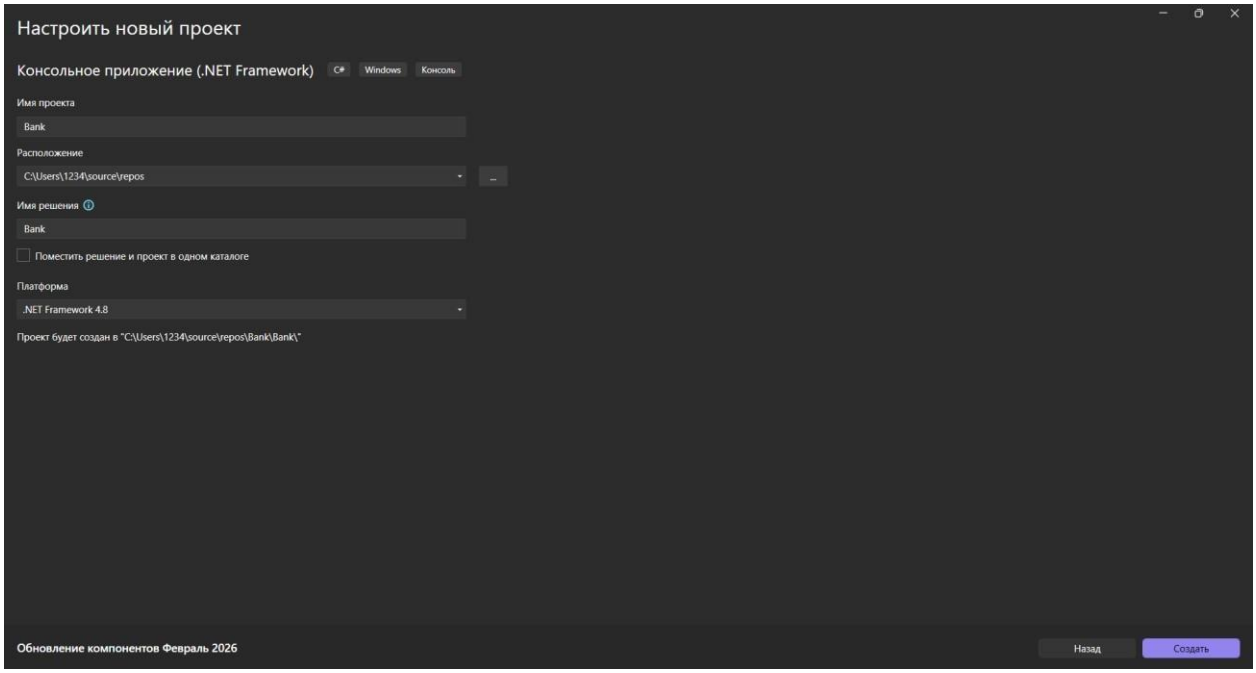


Рисунок 4

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);
        }
    }
}
```

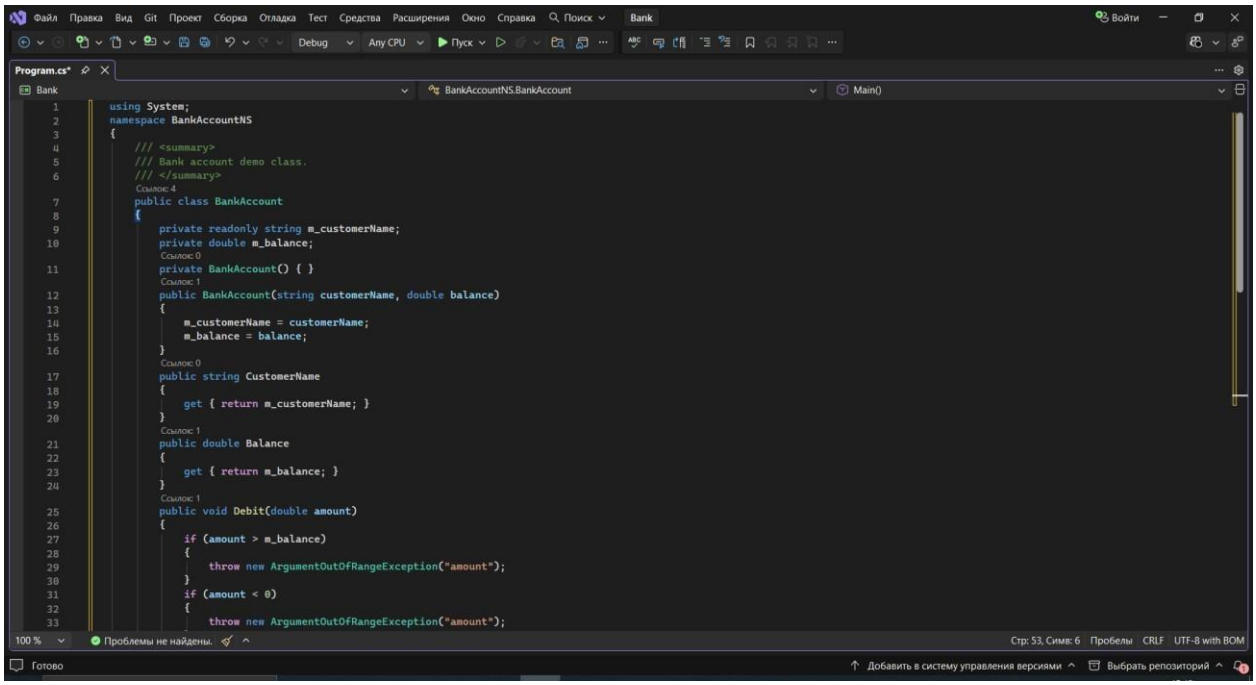


Рисунок 5

6. Я переименовала файл в BankAccount.cs.
Для этого я зашла в Обозреватель решений, нажала на Bank правой кнопки мыши и выбрала Переименовать.

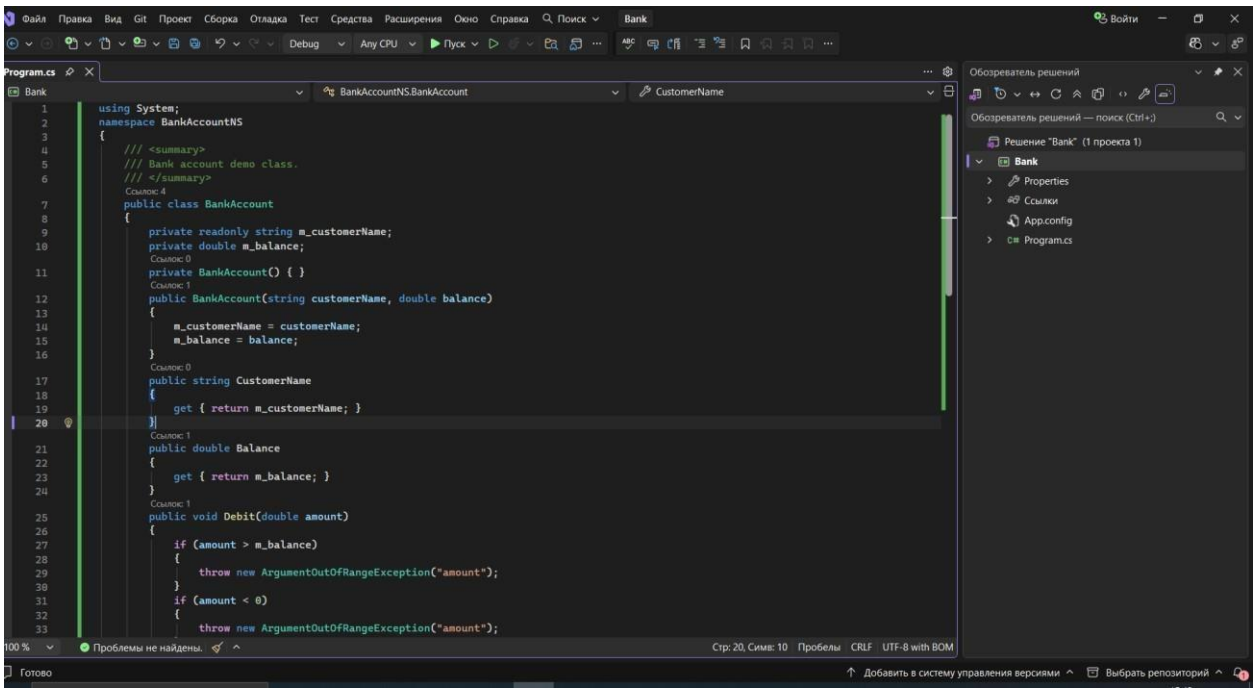


Рисунок 6

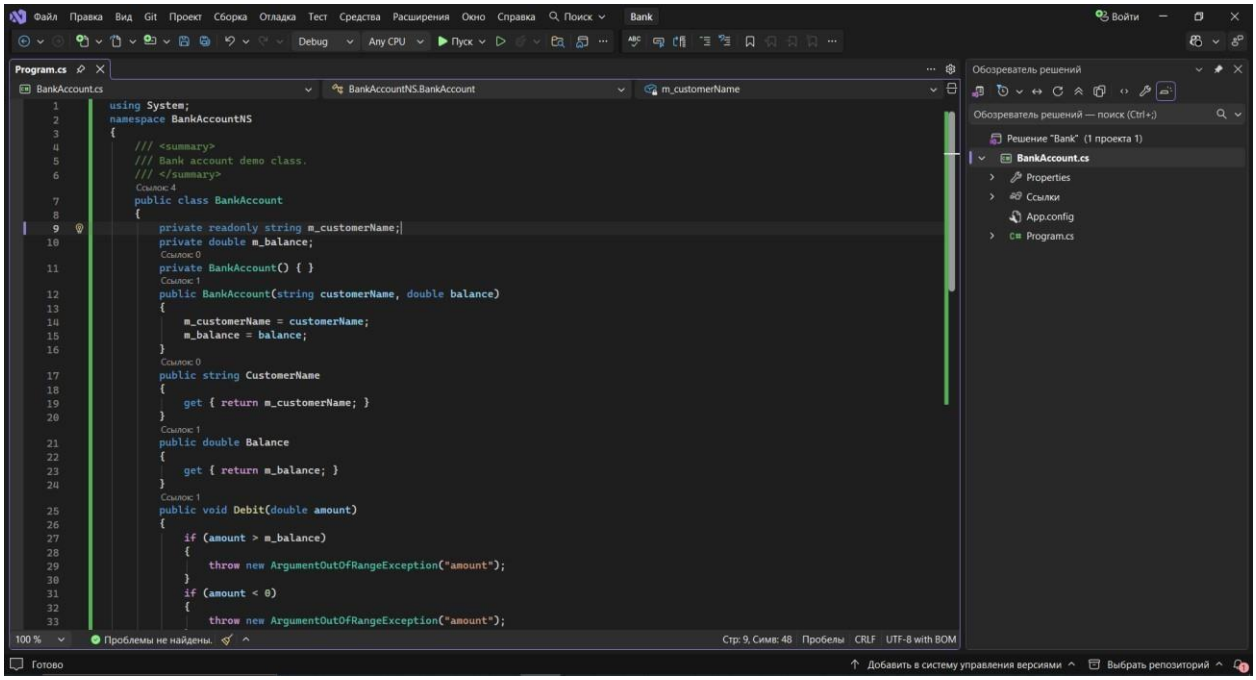


Рисунок 7

7. Я использовала клавиши CTRL +SHIFT + B для того, чтобы произошла сборка (вывод был).

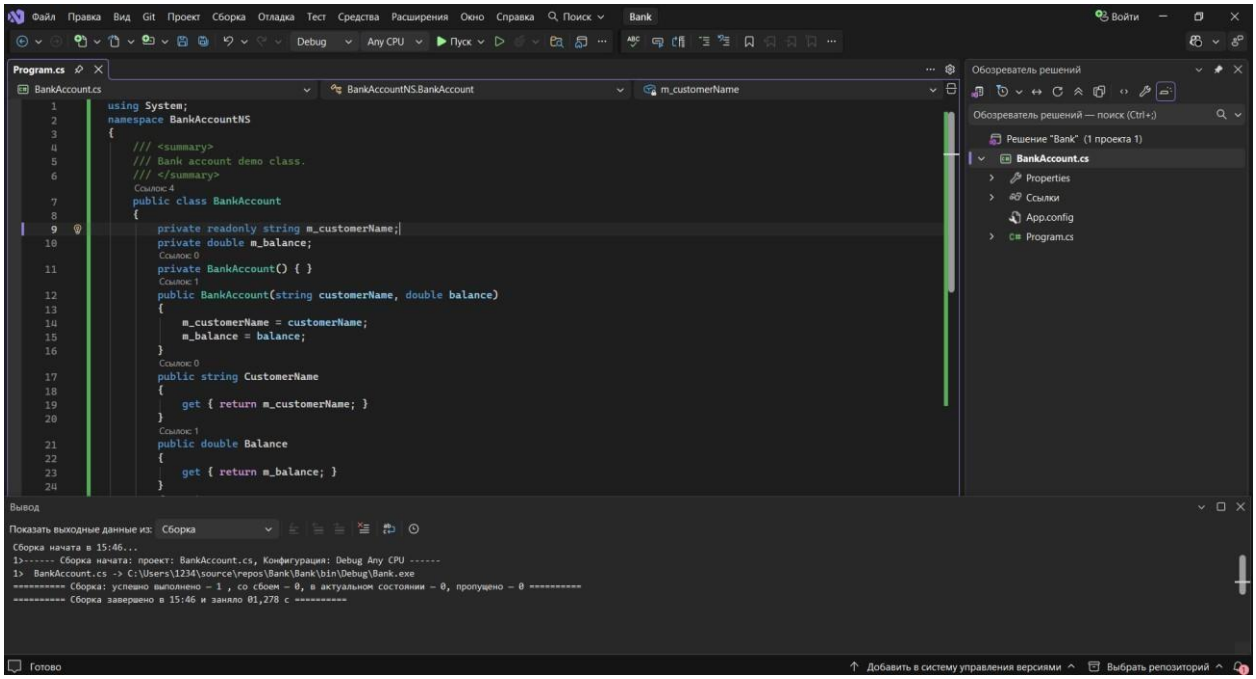


Рисунок 8

Результат моей работы:

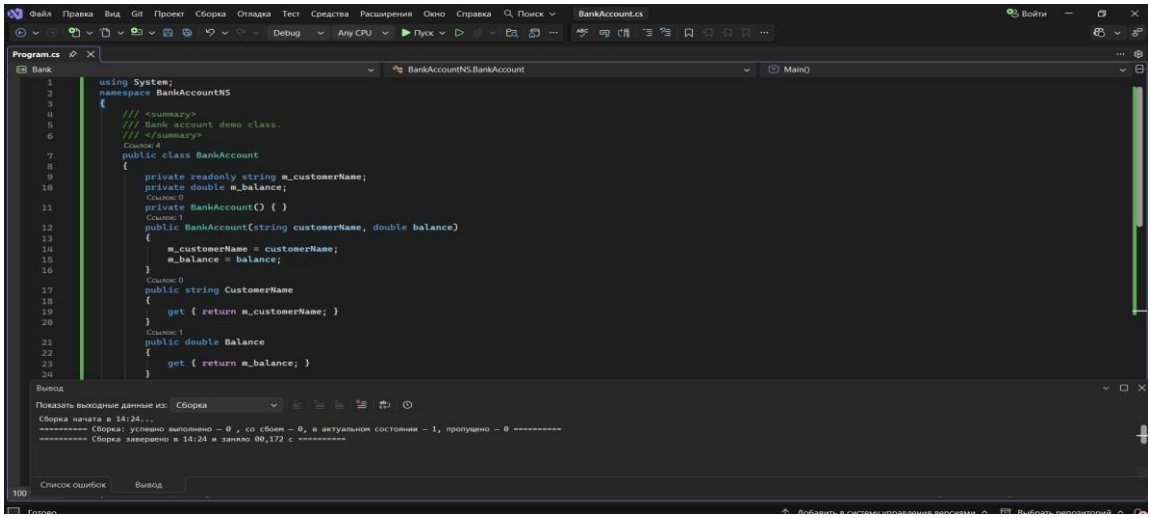


Рисунок 9

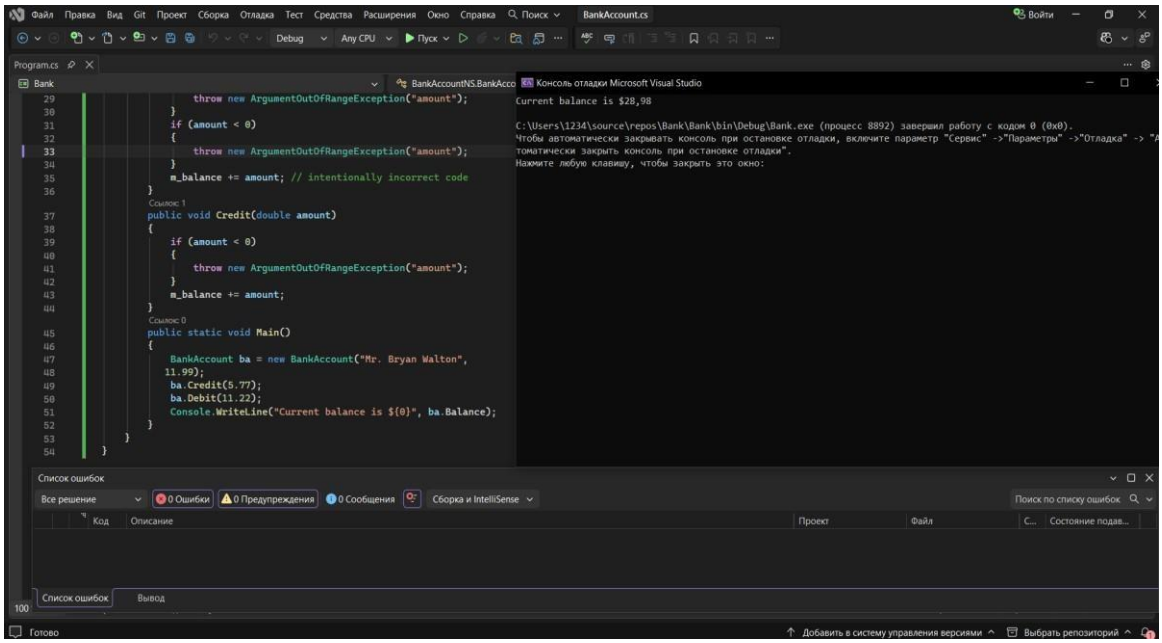


Рисунок 10

## 2. Создание проекта модульного теста.

1. В меню Файл я выбрала Добавить>Создать проект.

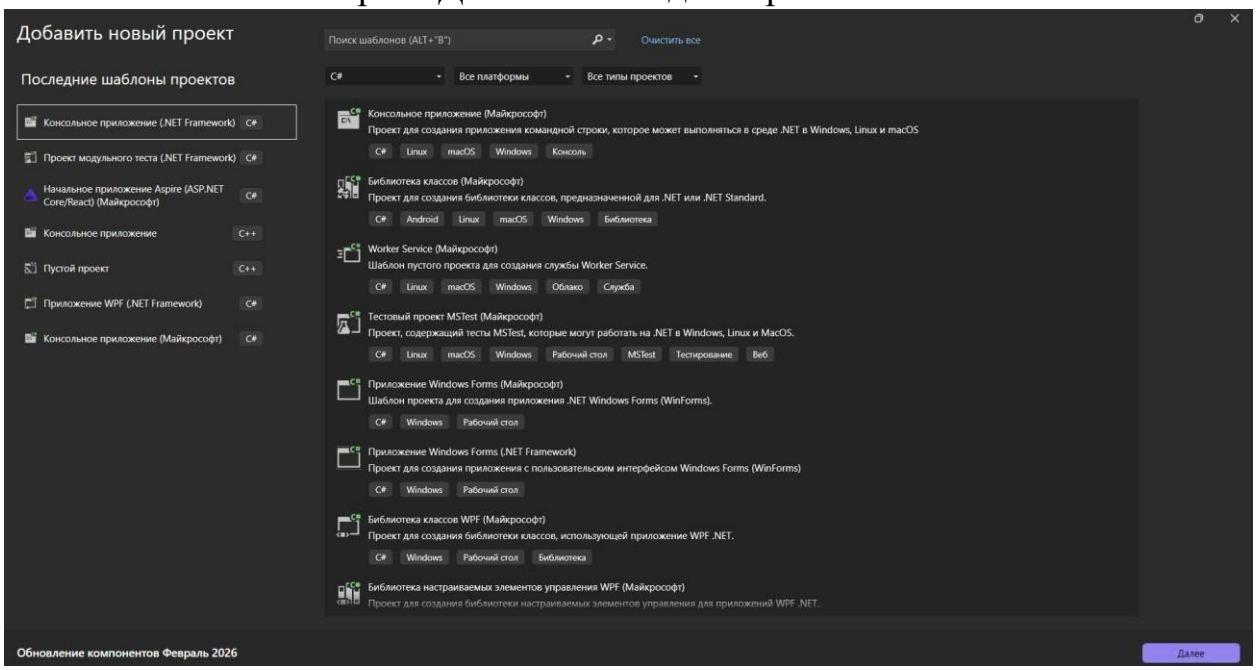


Рисунок 11

2. Я 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.

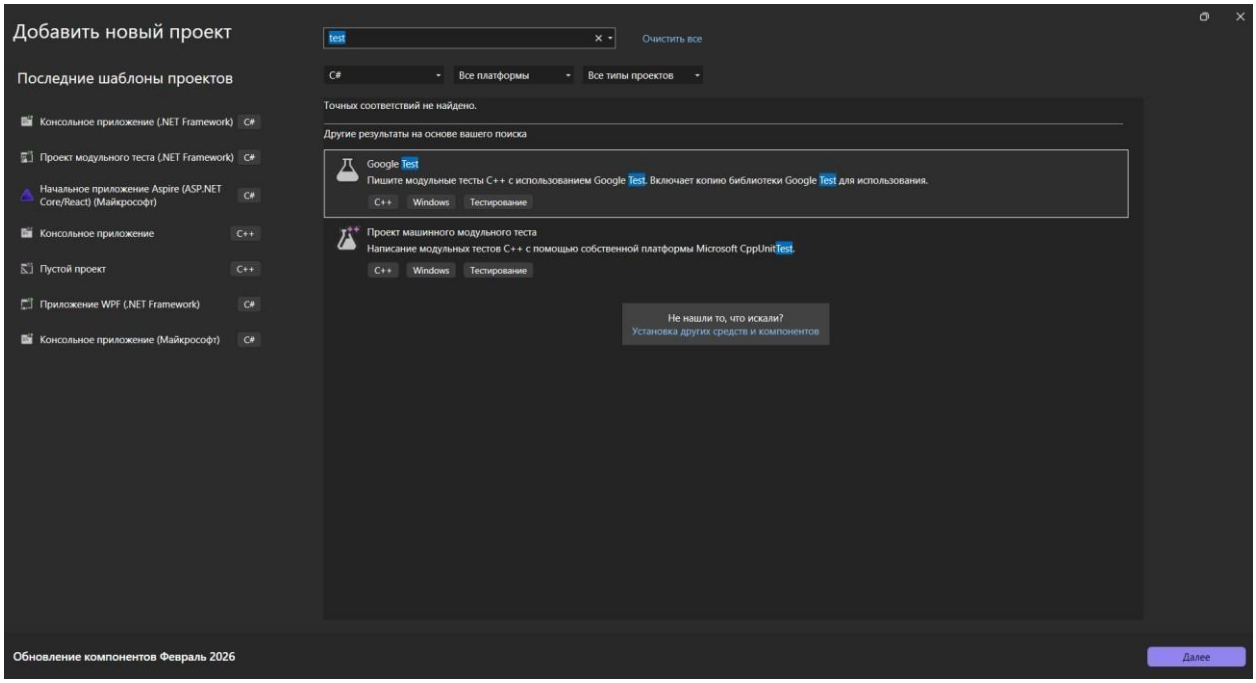


Рисунок 12

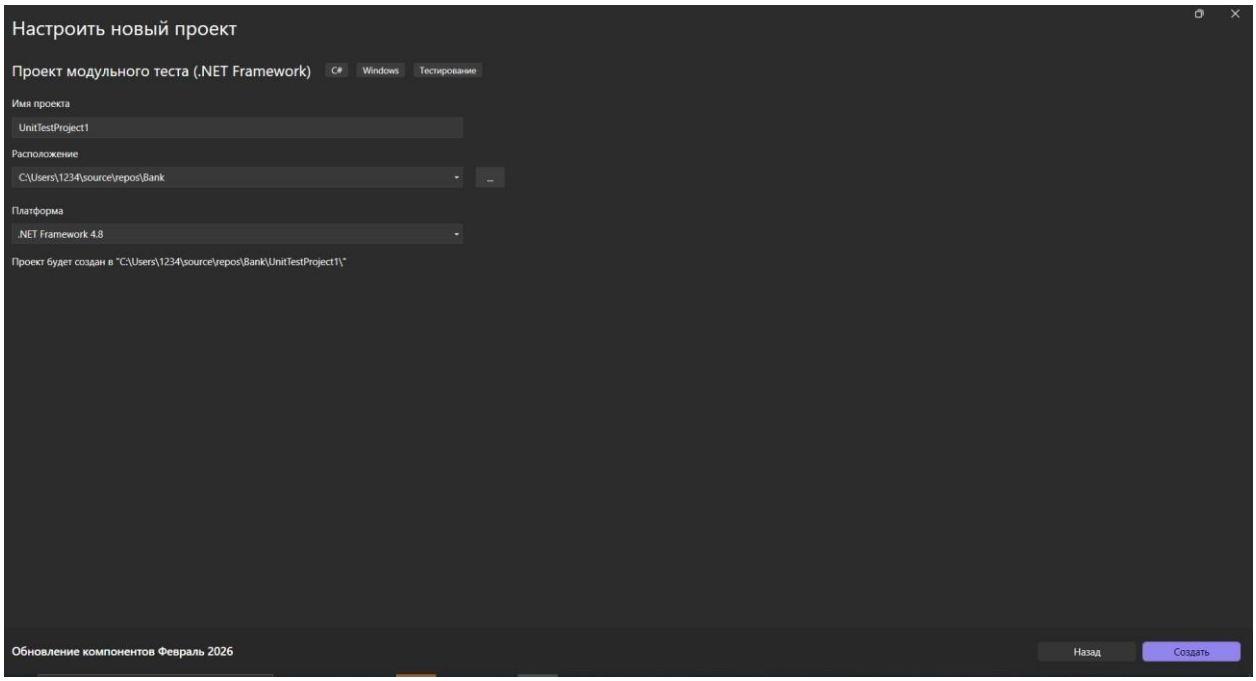


Рисунок 13

3. Я назвала проект BankTests.

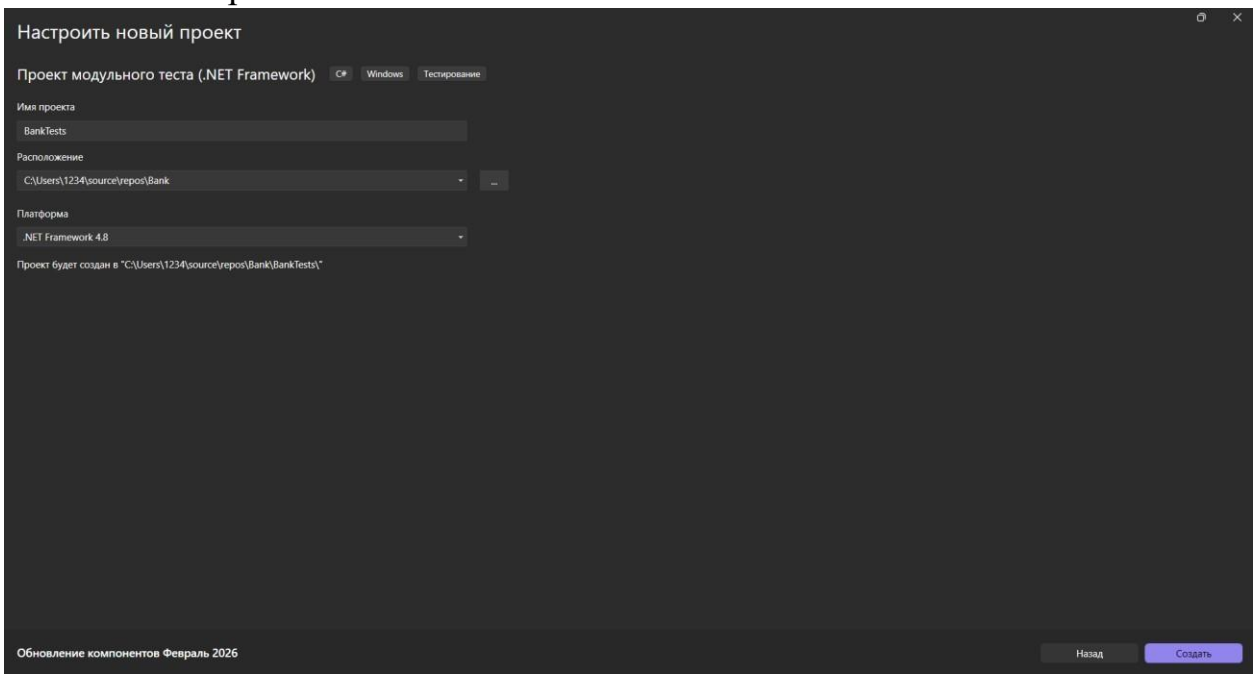


Рисунок 14

4. Я выбрала рекомендуемую версию целевой платформы.

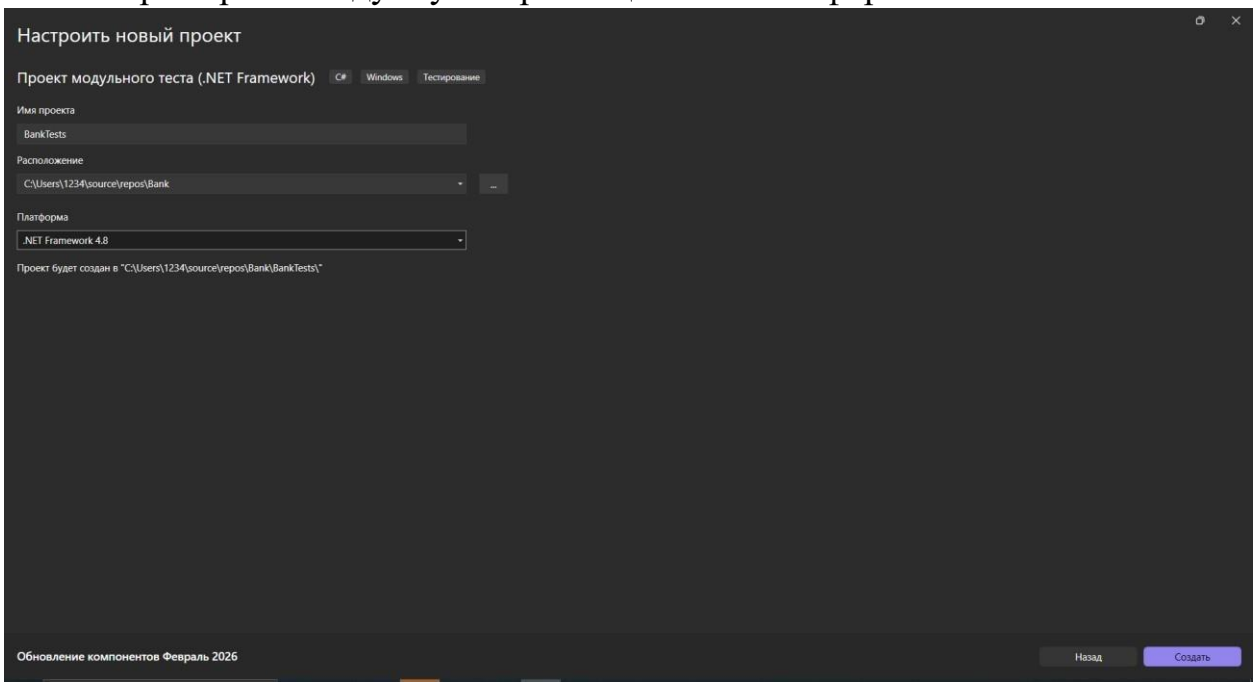


Рисунок 15

5. В проекте BankTests я добавила ссылку на проект Банк.

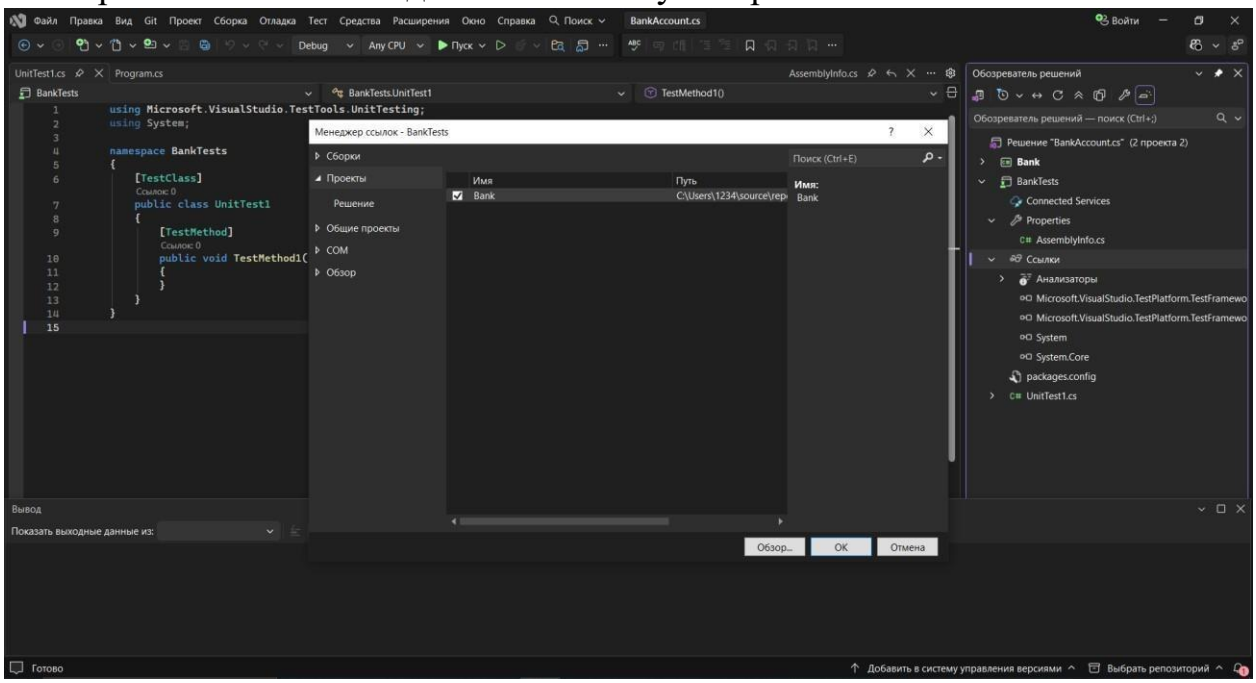


Рисунок 16

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

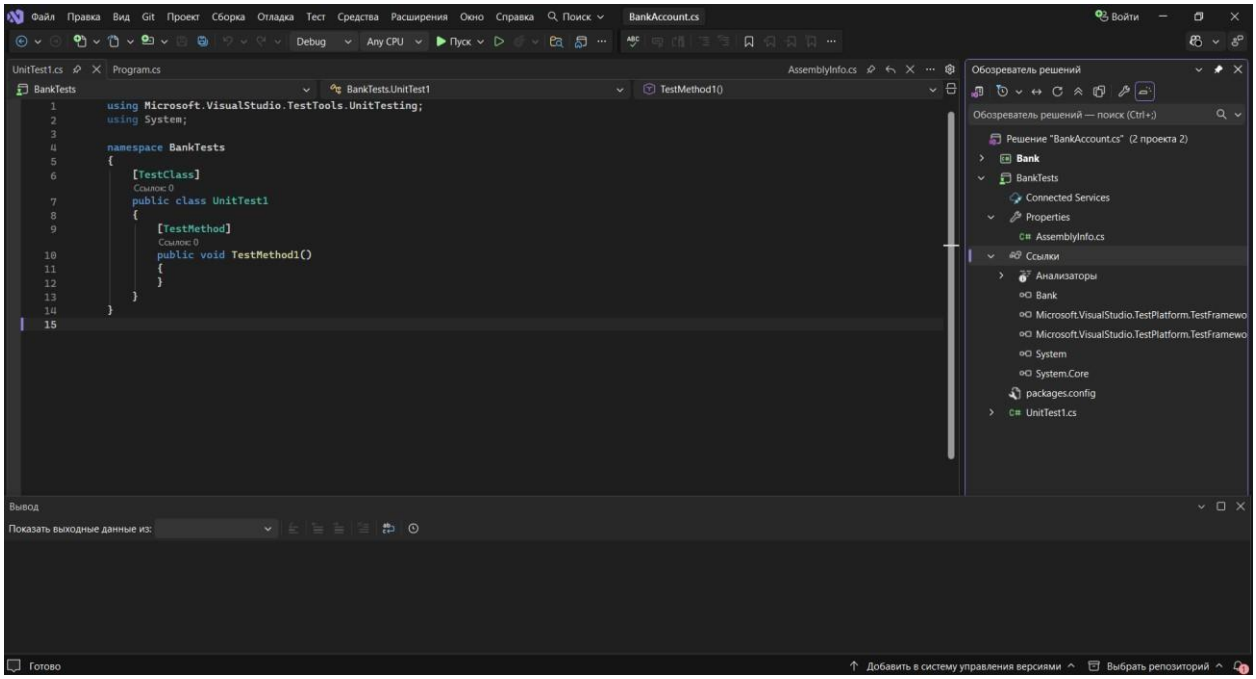


Рисунок 17

## 3. Создание тестового класса.

1. Я переименовала файл в BankAccountTests.cs.

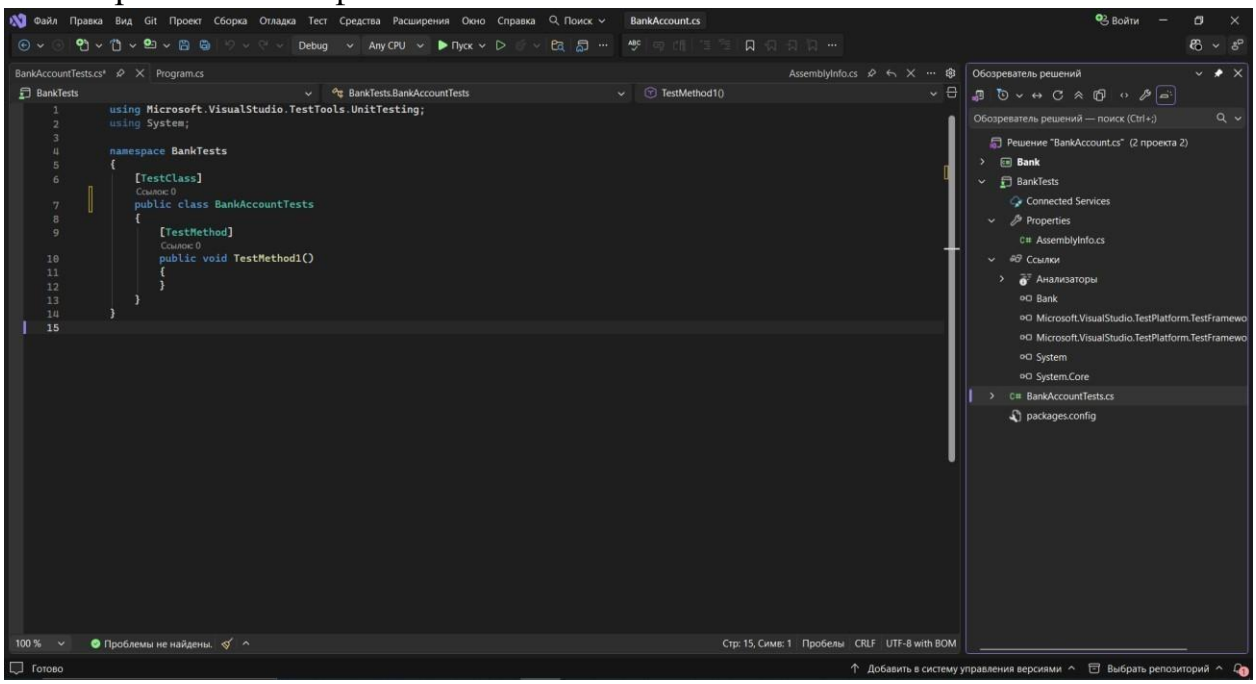


Рисунок 18

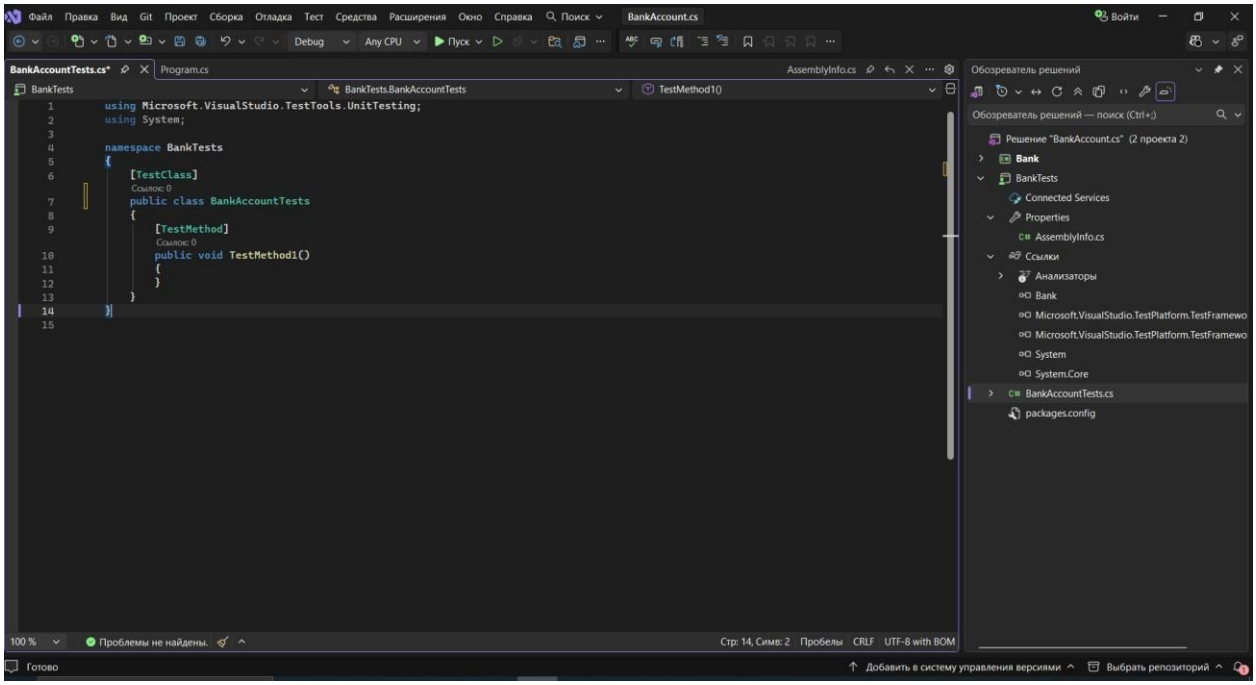


Рисунок 19

2. Я добавила оператор using.

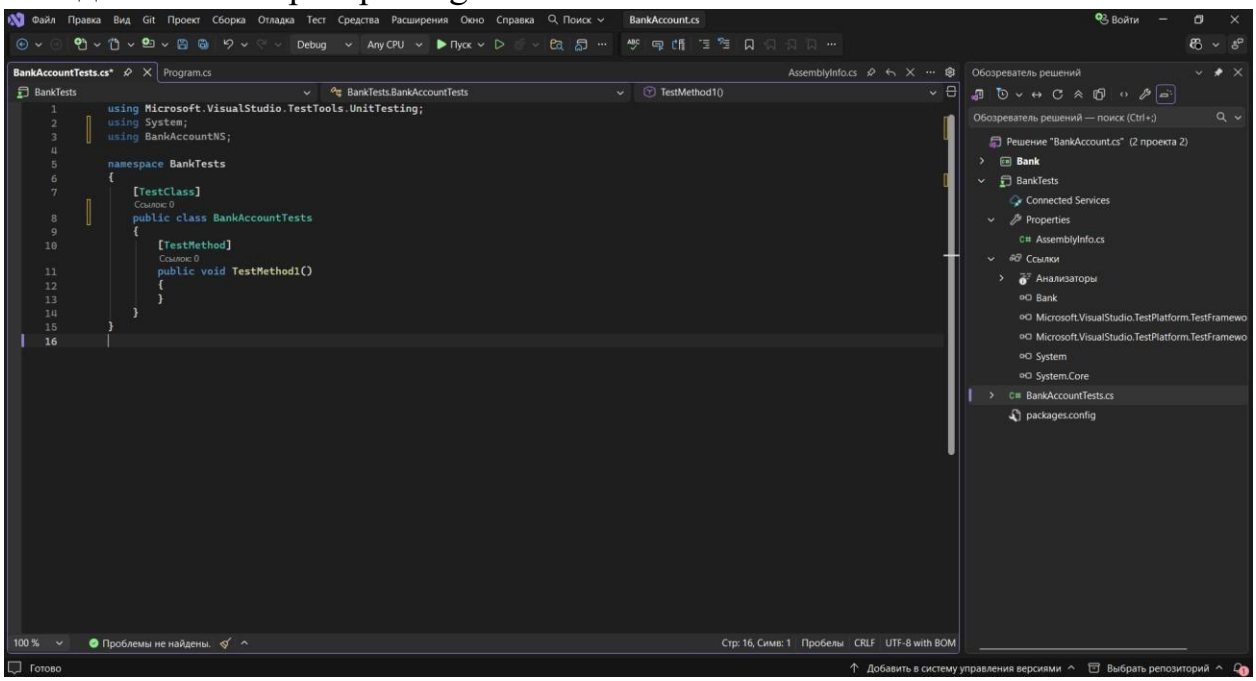


Рисунок 20

**4. Создание метода теста.**

1. Я 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");
       }
   }
}
```

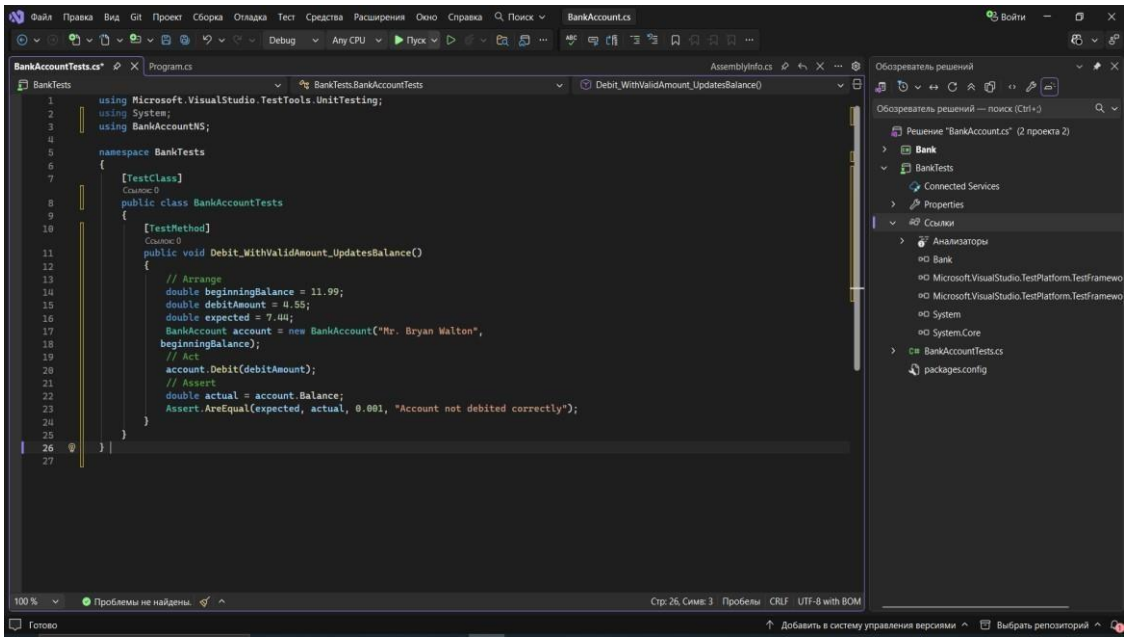
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System;
using BankAccountNS;
namespace BankTests
{
[TestClass]
public class BankAccountTests
{
[TestMethod]
public void Debit_WithValidAmount_UpdatesBalance()
{
// Arrange
double beginningBalance = 11.99;
double debitAmount = 4.55;
double expected = 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");
}
}
}

Рисунок 21

## 5. Сборка и запуск теста.

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

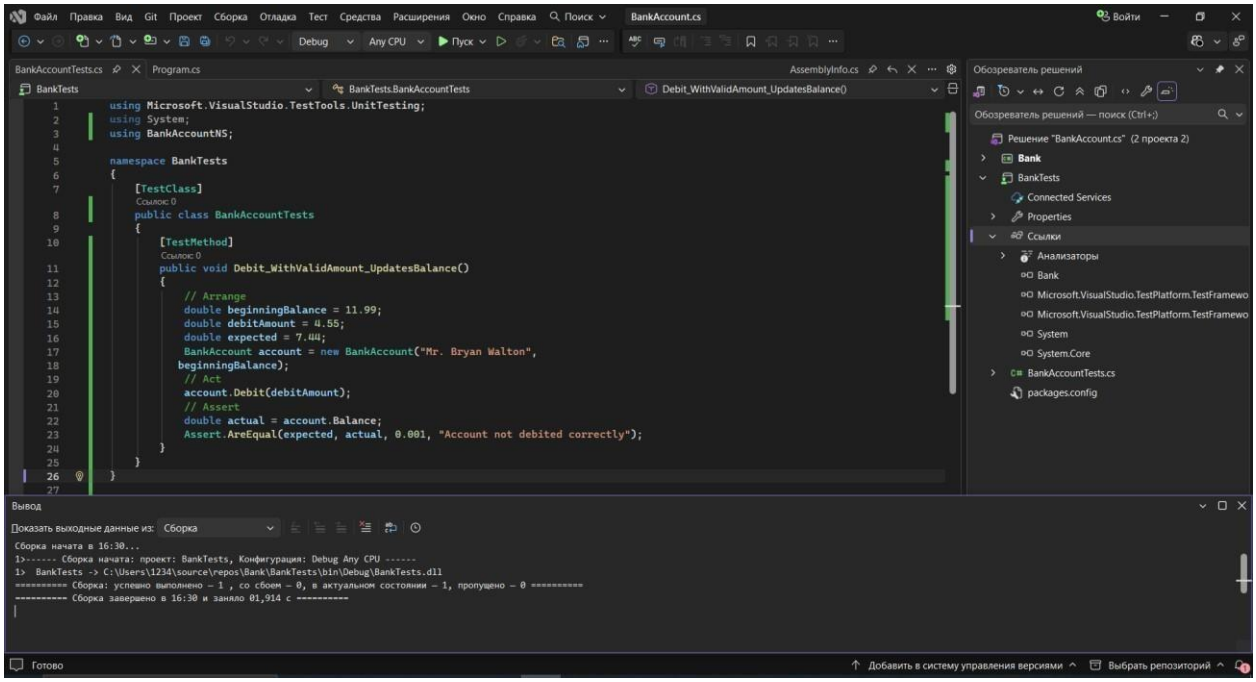


Рисунок 22

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

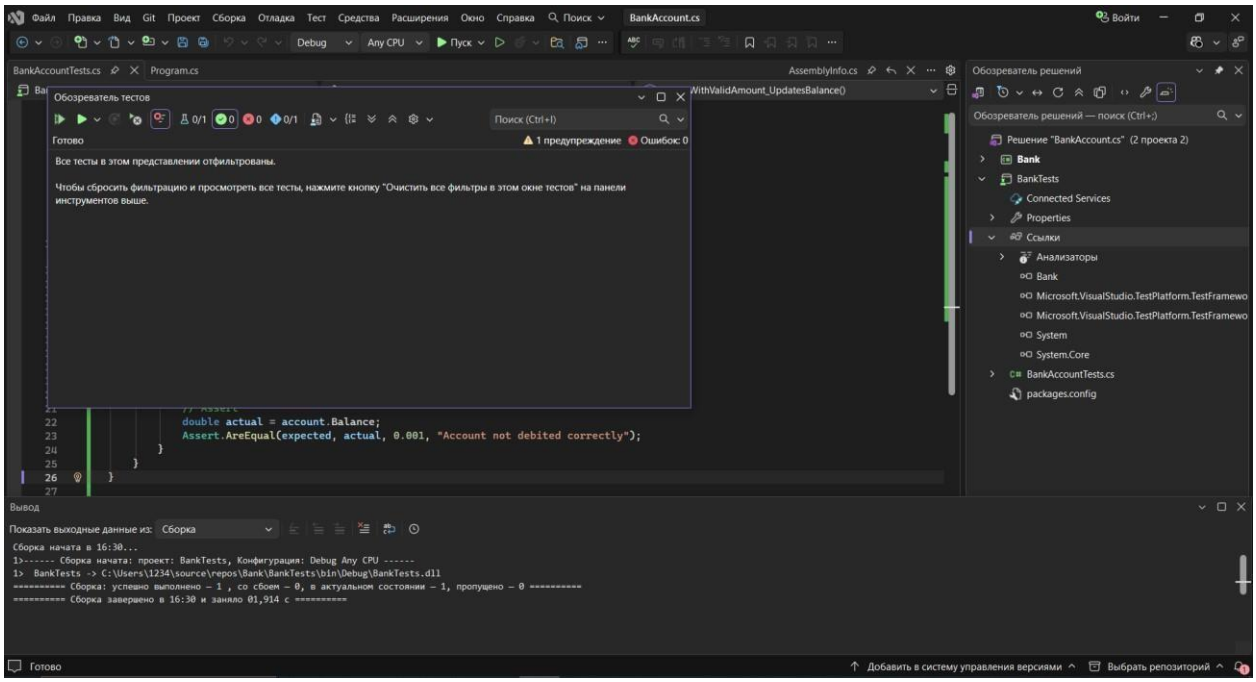


Рисунок 23

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

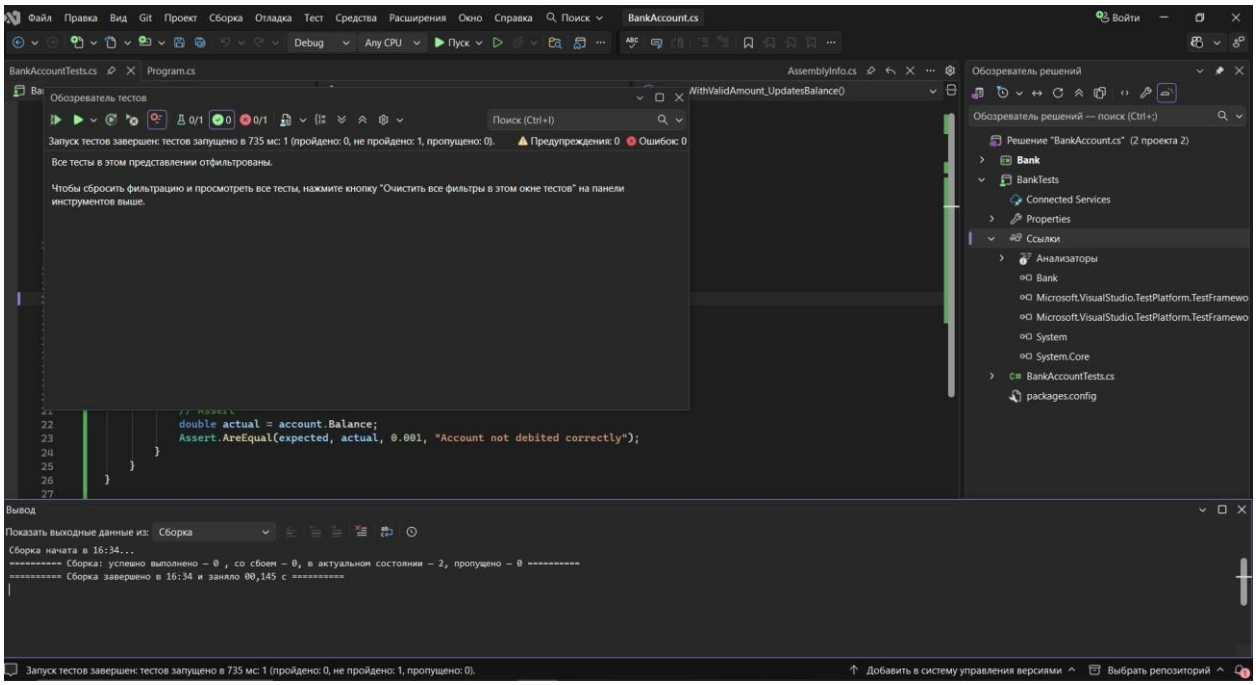


Рисунок 24

4. Я выбрала эту ошибку в обозревателе тестов для просмотра сведений в нижней части окна.

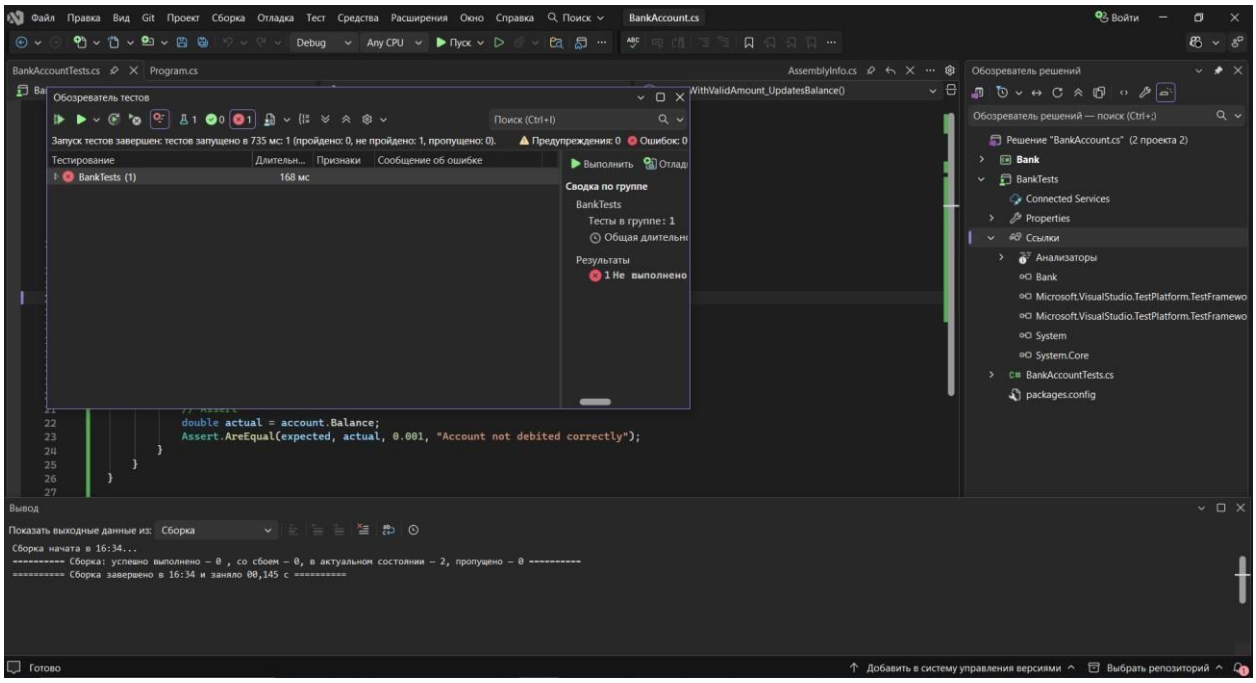


Рисунок 25

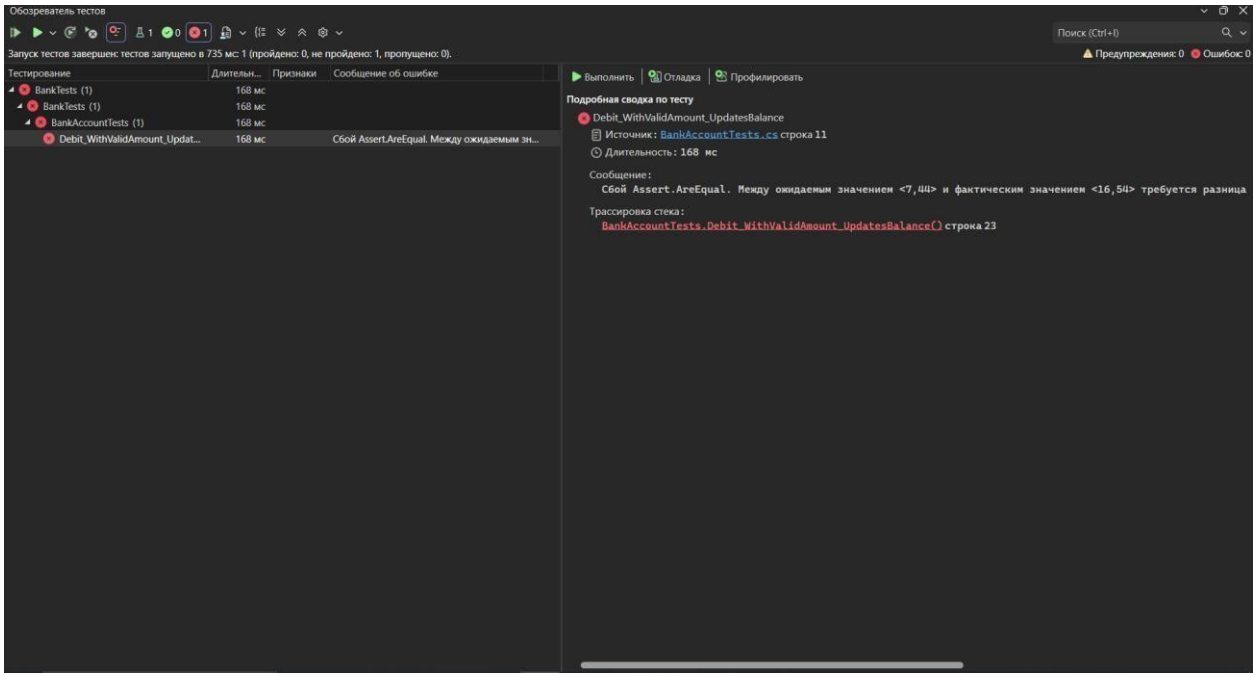
Обозреватель тестов
Поиск (Ctrl+I)
Запуск тестов завершен: тестов запущено в 735 мс: 1 (пройдено: 0, не пройдено: 1, пропущено: 0).
Предупреждения: 0
Ошибок: 0
Тестирование
Длительн...
Признаки
Сообщение об ошибке
BankTests (1) 168 мс
BankTests (1) 168 мс
BankAccountTests (1) 168 мс
Debit_WithValidAmount_Updat... 168 мс
Сбой Assert.AreEqual. Между ожидаемым зн...
Выполнить
Отладка
Профилировать
Подробная сводка по тесту
Debit_WithValidAmount_UpdatesBalance
Источник: BankAccountTests.cs строка 11
Длительность: 168 мс
Сообщение:
Сбой Assert.AreEqual. Между ожидаемым значением <7,44> и фактическим значением <16,54> требуется разница
Трассировка стека:
BankAccountTests.Debit_WithValidAmount_UpdatesBalance() строка 23

Рисунок 26

## 6. Исправление кода и повторный запуск тестов.

Я сделала исправление ошибки: В файле BankAccount.cs
заменила строки с:
m_balance += amount;
на:
m_balance -= amount;

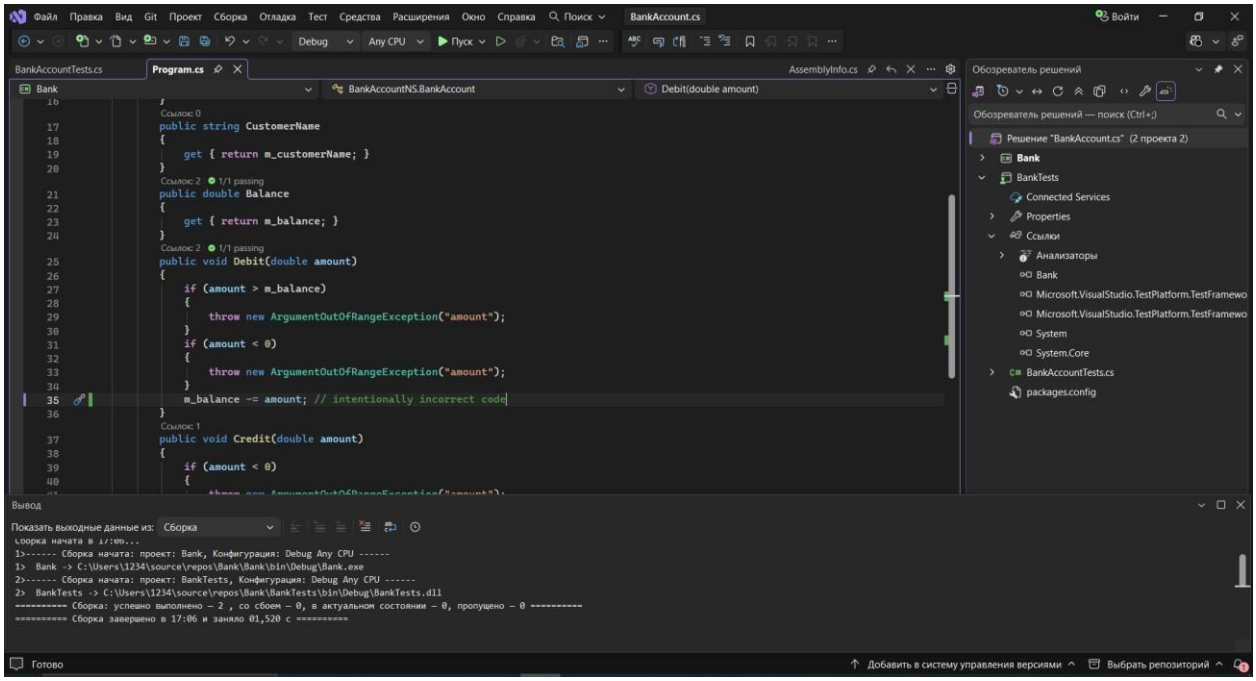


Рисунок 27

Я сделала повторный запуск теста.

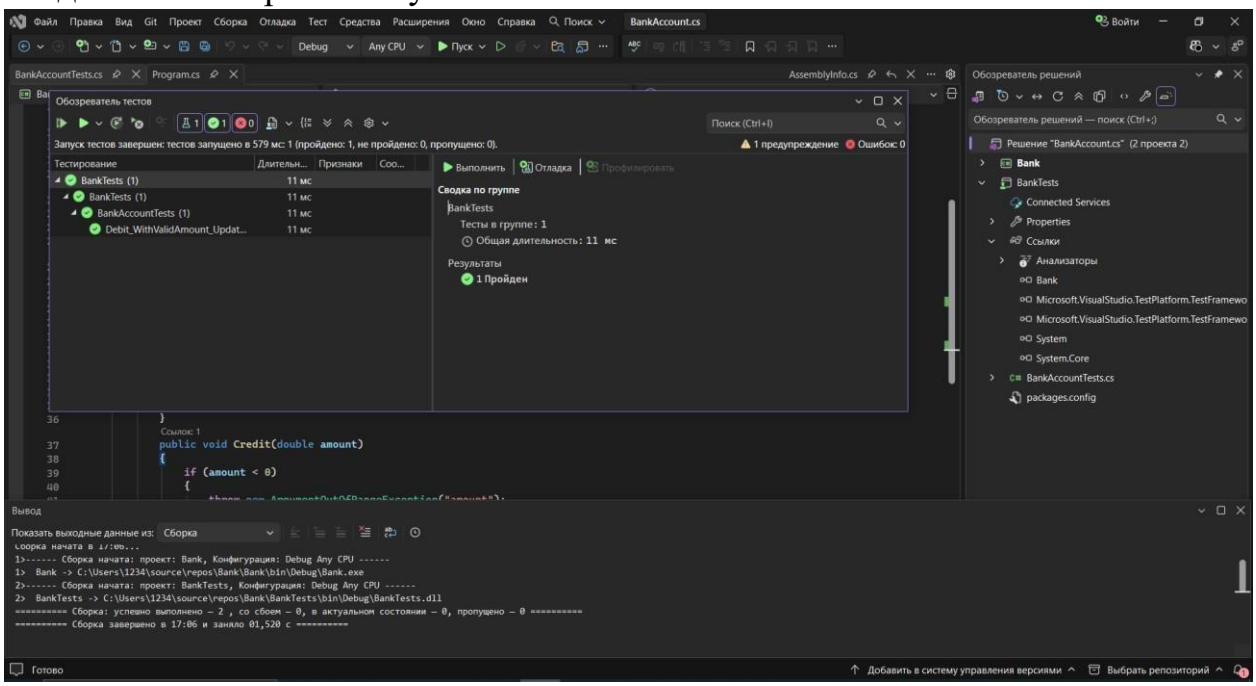


Рисунок 28

**7. Создание и запуск новых методов теста.**

1. Я ввела код:

```
[TestMethod]
public void
Debit_WhenAmountIsLessThanZero_ShouldThrowArgumentOutOfRange()
 {
    // 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));
}
```

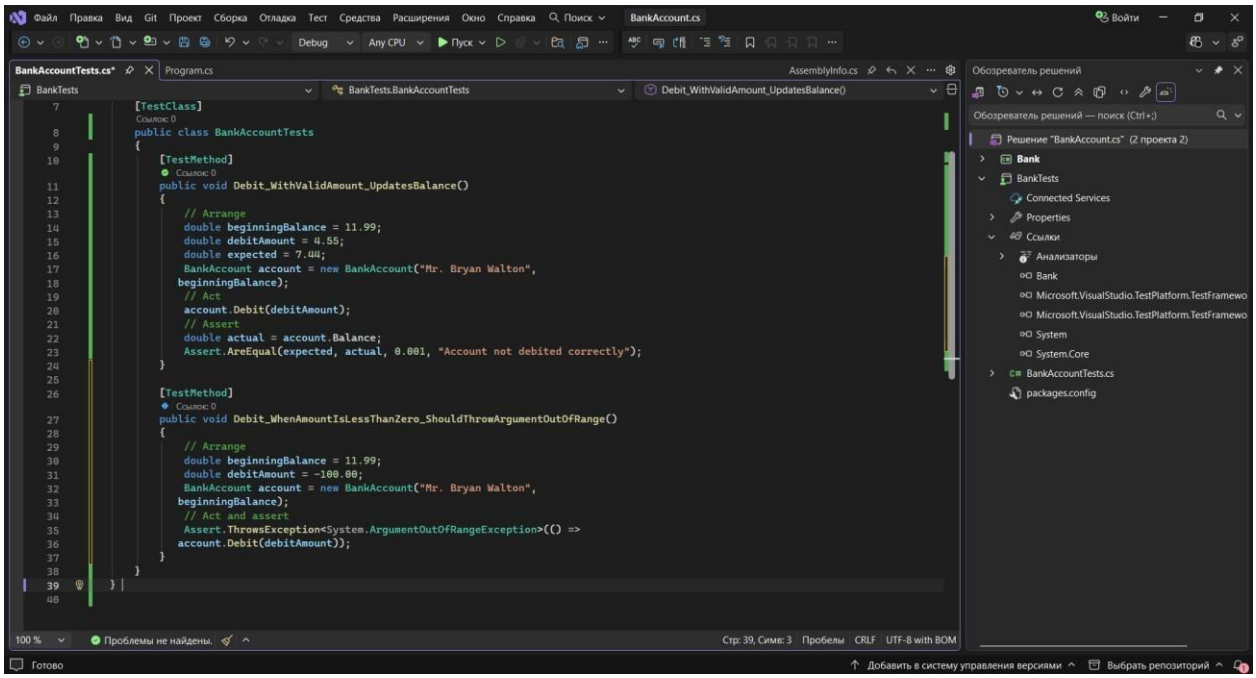


Рисунок 29

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

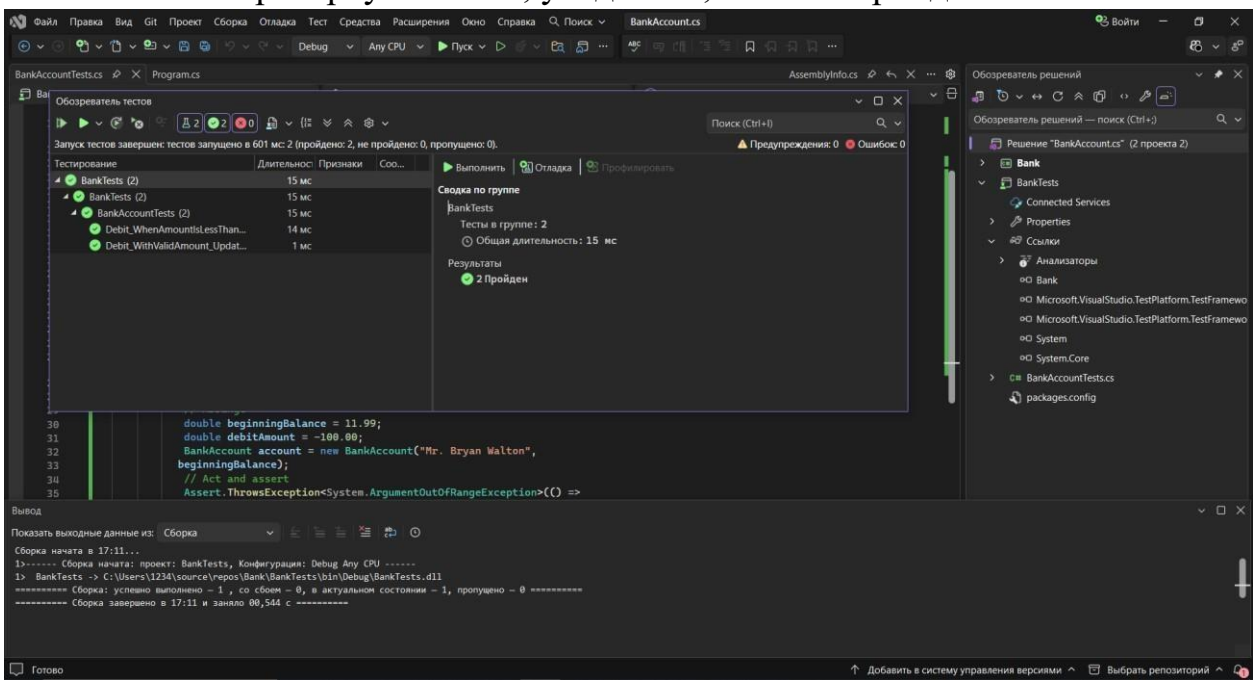


Рисунок 30

## 8. Рефакторинг тестируемого кода.

1. Я добавила это в тестируемый класс ( BankAccount ):
public const string DebitAmountExceedsBalanceMessage = "Debit amount exceeds balance";
public const string DebitAmountLessThanZeroMessage = "Debit amount is less than zero";

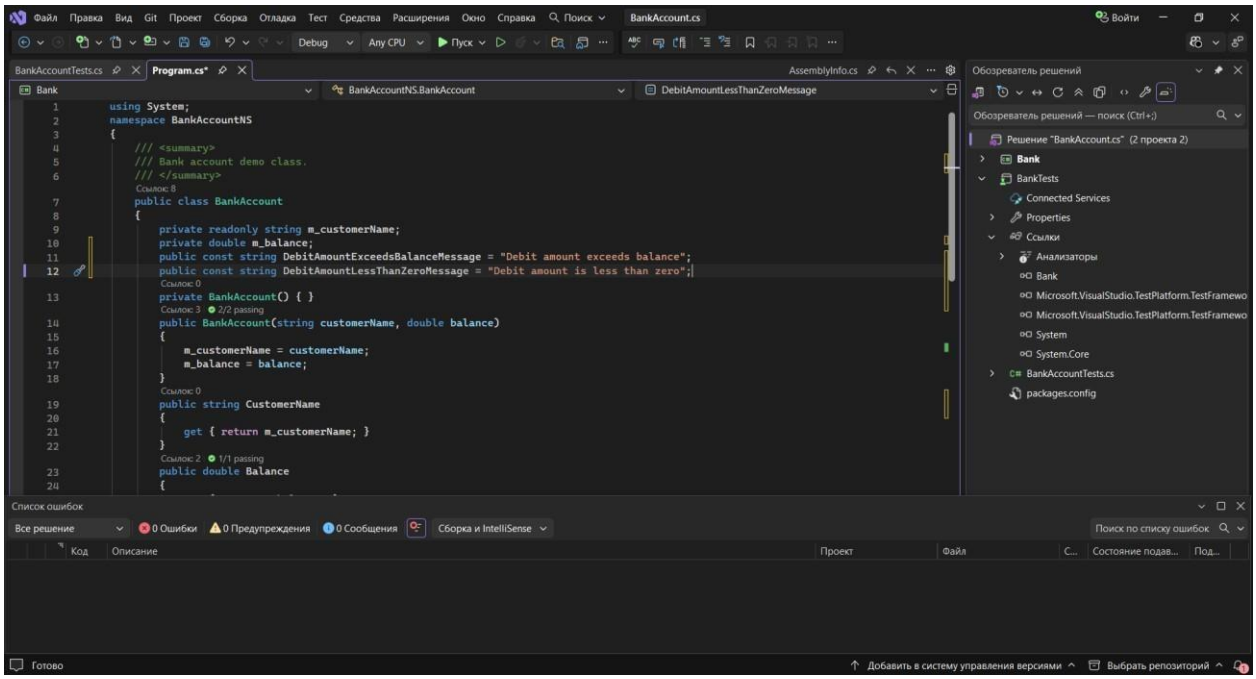


Рисунок 31

2. Я сделала замену двух условных операторатороов в методе Debit:
```
if (amount > m_balance)
 {
    throw new System.ArgumentOutOfRangeException("amount", amount, DebitAmountExceedsBalanceMessage);
 }
 if (amount < 0)
 {
    throw new System.ArgumentOutOfRangeException("amount", amount, DebitAmountLessThanZeroMessage);
 }
```

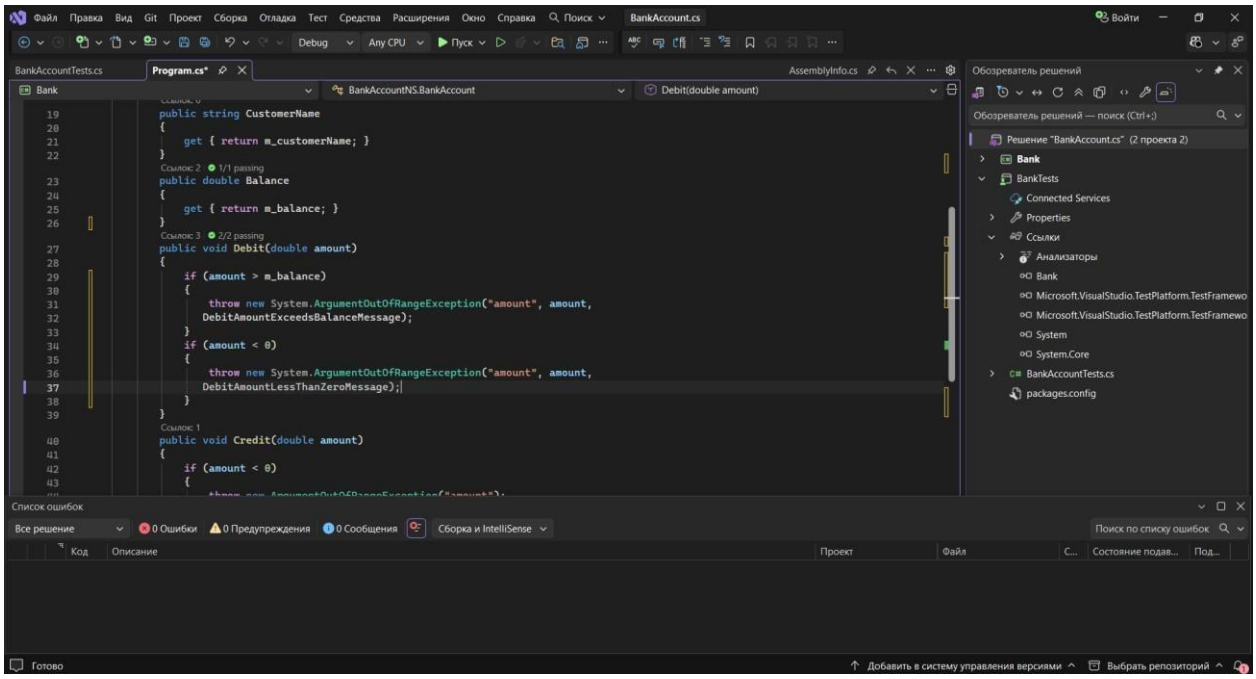
Файл Правка Вид Git Проект Сборка Отладка Тест Средства Расширения Окно Справка Поиск
BankAccount.cs
Войти
Debug
Any CPU
Пуск
BankAccountTests.cs
Program.cs*
AssemblyInfo.cs
Bank
BankAccountNS.BankAccount
Debit(double amount)
public string CustomerName
{
get { return m_customerName; }
}
Ссылок: 2 1/1 passing
public double Balance
{
get { return m_balance; }
}
Ссылок: 3 2/2 passing
public void Debit(double amount)
{
if (amount > m_balance)
{
throw new System.ArgumentOutOfRangeException("amount", amount,
DebitAmountExceedsBalanceMessage);
}
if (amount < 0)
{
throw new System.ArgumentOutOfRangeException("amount", amount,
DebitAmountLessThanZeroMessage);
}
}
Ссылок: 1
public void Credit(double amount)
{
if (amount < 0)
{
Обозреватель решений
Обозреватель решений — поиск (Ctrl+;)
Решение "BankAccount.cs" (2 проекта 2)
Bank
BankTests
Connected Services
Properties
Ссылки
Анализаторы
Bank
Microsoft.VisualStudio.TestPlatform.TestFramewo
Microsoft.VisualStudio.TestPlatform.TestFramewo
System
System.Core
BankAccountTests.cs
packages.config
Список ошибок
Все решение
0 Ошибки
0 Предупреждения
0 Сообщения
Сборка и IntelliSense
Поиск по списку ошибок
Код
Описание
Проект
Файл
Состояние подав...
Под...
Готово
Добавить в систему управления версиями
Выбрать репозиторий

Рисунок 32

**9. Рефакторинг тестовых методов.**

Я ввела код:

```
[TestMethod]
public void
Debit_WhenAmountIsMoreThanBalance_ShouldThrowArgumentOutOfRange()
{
   // 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);
   }
```

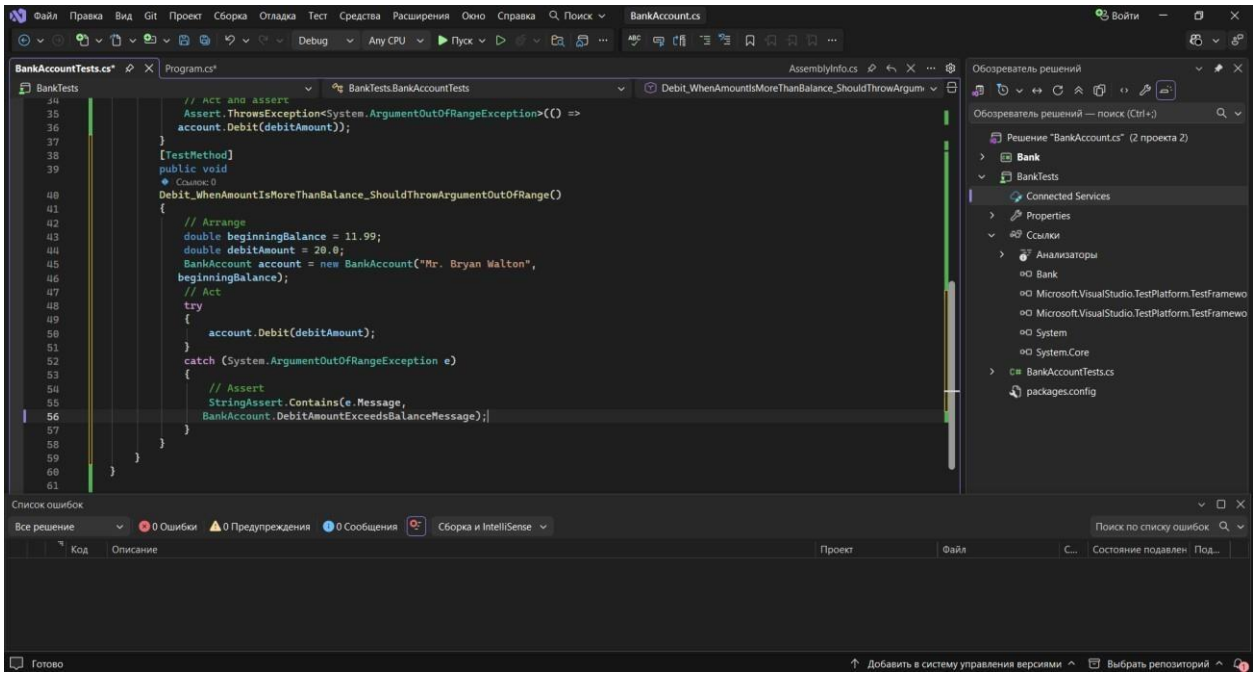


Рисунок 33

## 10. Повторное тестирование, переписывание и анализ.

1. Я редактировала код.

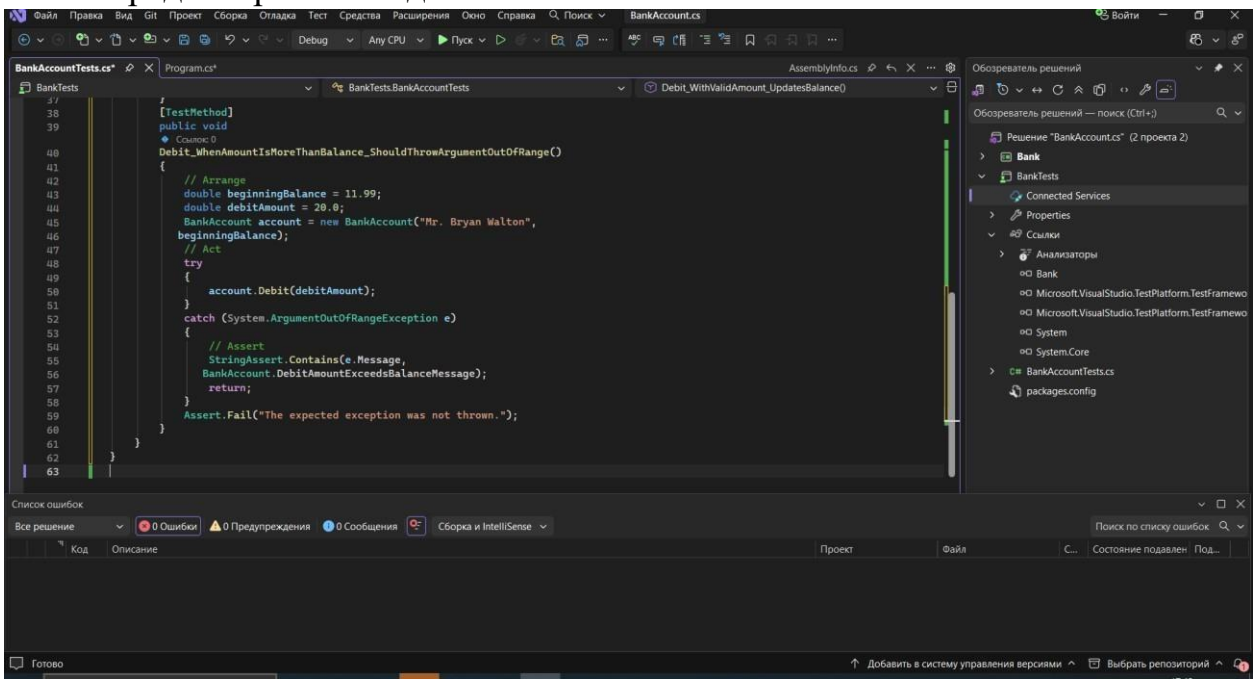


Рисунок 34

2. Я провела проверку тестов. Я обнаружила ошибки в одном из тестов.

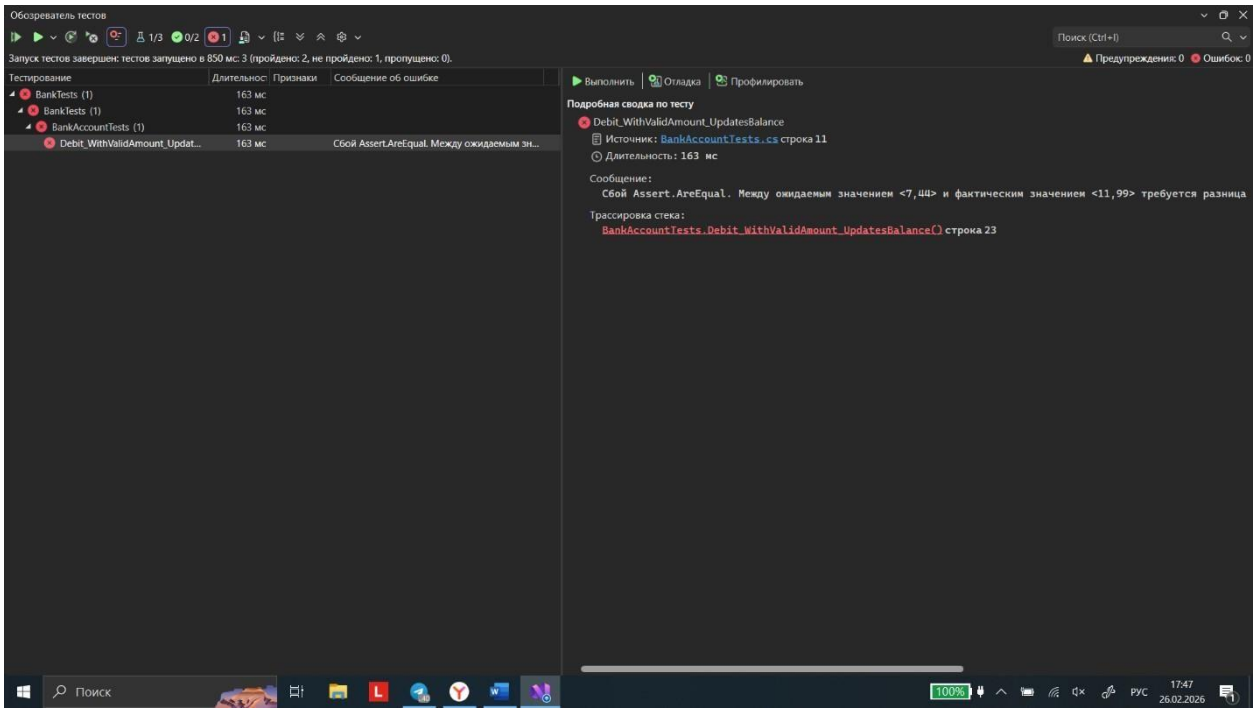


Рисунок 35

3. Я приняла решение анализировать ошибку и искать причины её возникновения, и пыталась решить возникшие ошибки.
Я добавила строку в код для устранения ошибки: m_balance -= amount;

4. Я совершила запуск всех тестов (повторная проверка).

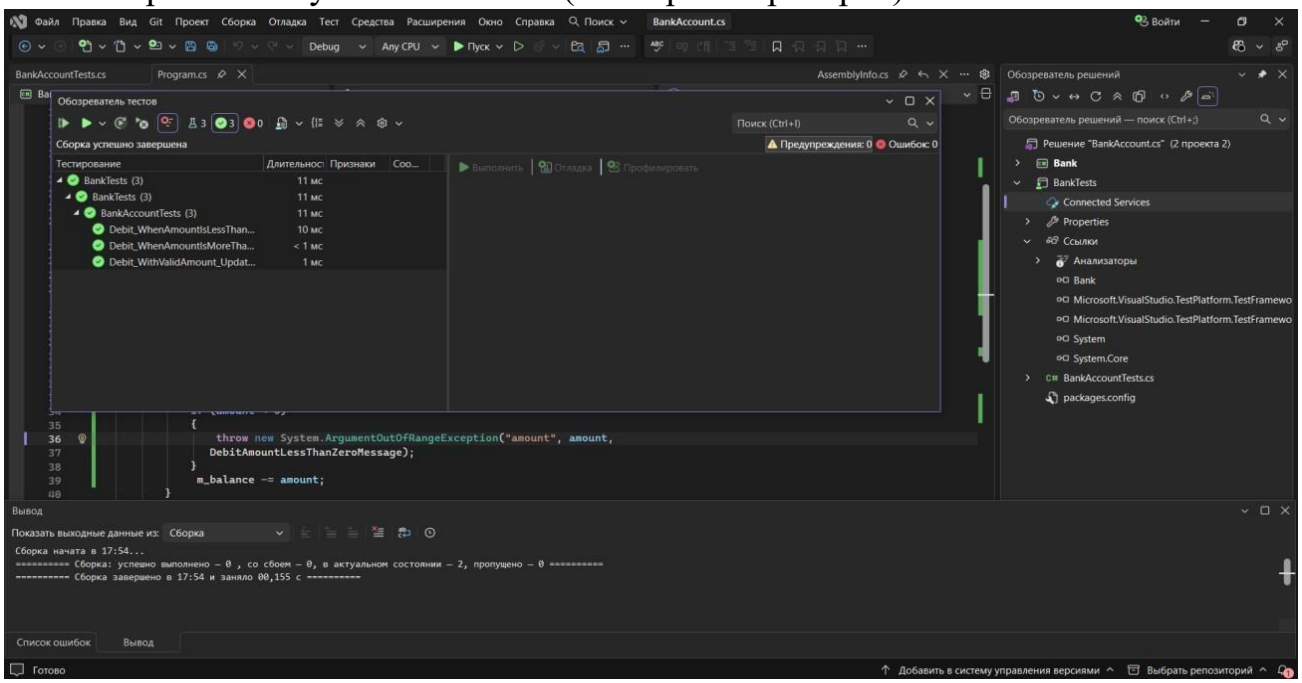


Рисунок 36

5. Я провела анализирование вывода, ошибка устранена. Все тесты выполнены правильно.

6. Проверка работы программы.

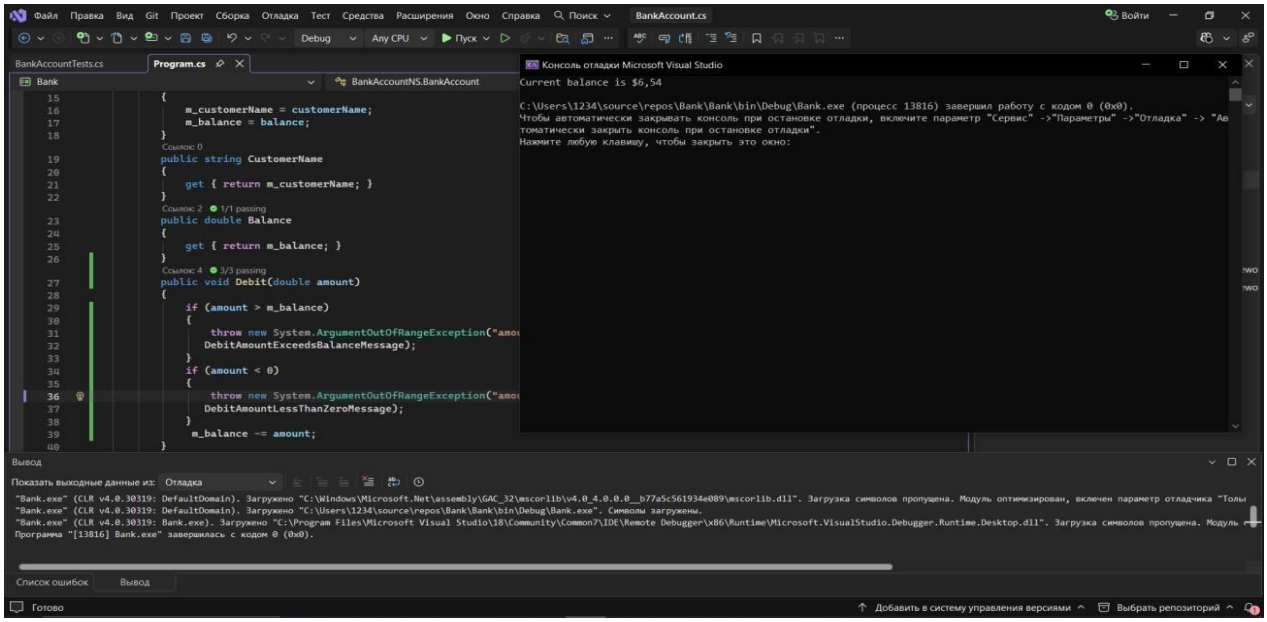


Рисунок 37

## 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. Я научилась составлять официальную документацию. Осознала, что всегда нужно перепроверять свою работу множество раз и что заниматься тестированием очень классно.