

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федерального государственного бюджетного образовательного
учреждения
высшего образования
**«РОССИЙСКИЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ
Г.В.ПЛЕХАНОВА»**
Техникум Пермского института (филиала)

Лабораторная работа
по дисциплине «Поддержка и тестирование программных модулей»
по теме: Создание и запуск модульных тестов для управляемого кода

Руководитель
Д.Б. Берестов /И.О. Фамилия/
«_24_» _____ 02 _____ 2025 г.
Исполнитель
В.Р. Иманаев /И.О. Фамилия/
«_24_» _____ 02 _____ 2025 г.

Пермь, 2026 г.

Оглавление

Создайте проект для тестирования	3
1. Создание файла	3
2. Настройка файла	3
Создание проекта модульного теста.....	5
1. Создание файла	5
2. Настройка файла	6
Тестирование.....	10

Создайте проект для тестирования

1. Создание файла

Сперва мы открываем приложение «Visual Code», в поисковой строке вбиваем консольное приложение и выбираем «консольное приложение(.NET Framework)» (смотрите рис. 1.1)

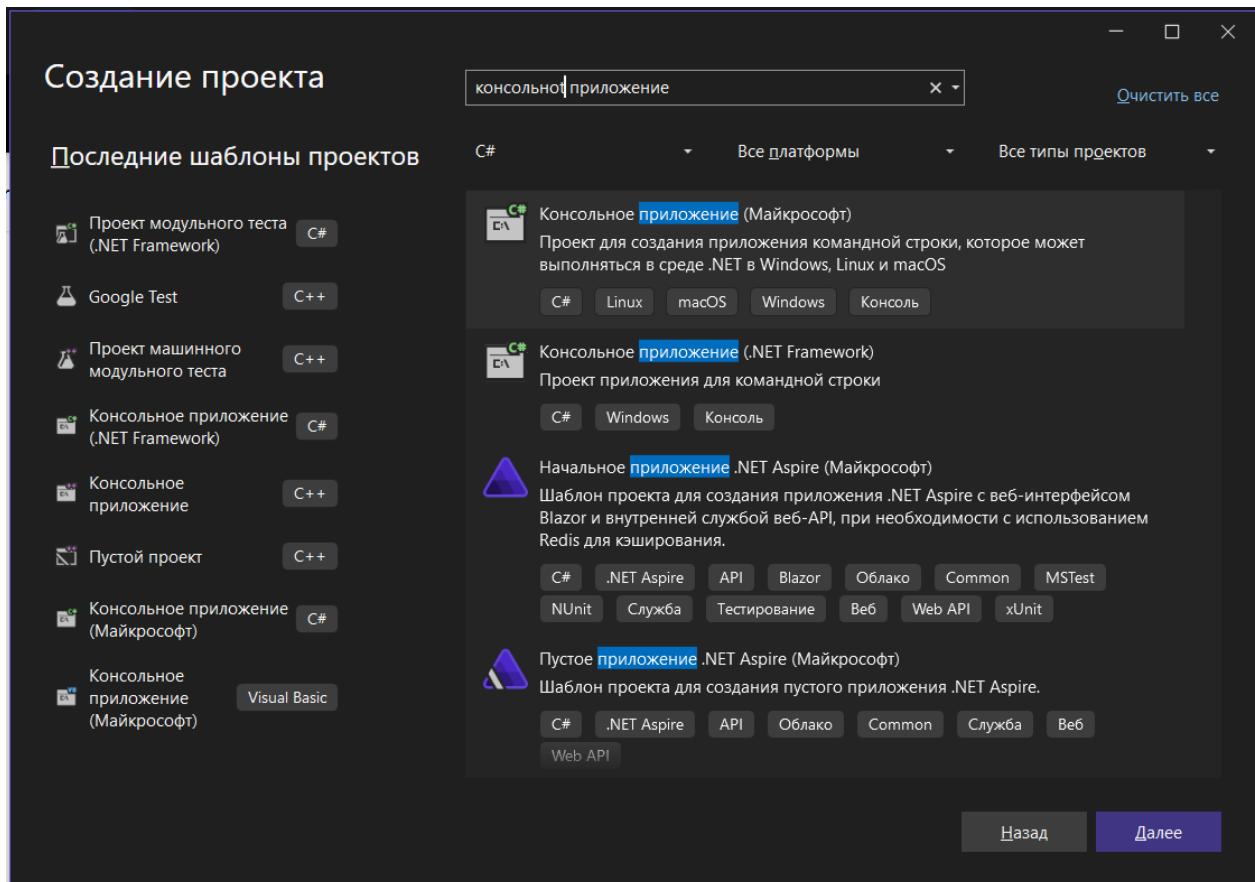


Рис. 1.1

2. Настройка файла

Назначим имя проекта «Bank» и нажмем создать, после этого у нас откроется файл «Program.cs» (его можно открыть через обозреватель решений) и затем вписываем туда след. код C#:

```
using System;
using System.Collections.Generic;
```

```

using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BankAccountNS
{
    /// <summary>
    /// Bank account demo class.
    /// </summary>
    public class BankAccount
    {
        public const string DebitAmountExceedsBalanceMessage = "Debit amount exceeds balance";
        public const string DebitAmountLessThanZeroMessage = "Debit amount is less than zero";

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

```

```
        DebitAmountLessThanZeroMessage);  
    }  
  
    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);  
}  
}
```

После нам нужно собрать решение (**CTRL + SHIFT + B**).

Создание проекта модульного теста

1. Создание файла

В меню «Файл» нужно нажать на кнопку «Добавить» и затем «Создать проект». (смотрите Рис. 2.1)

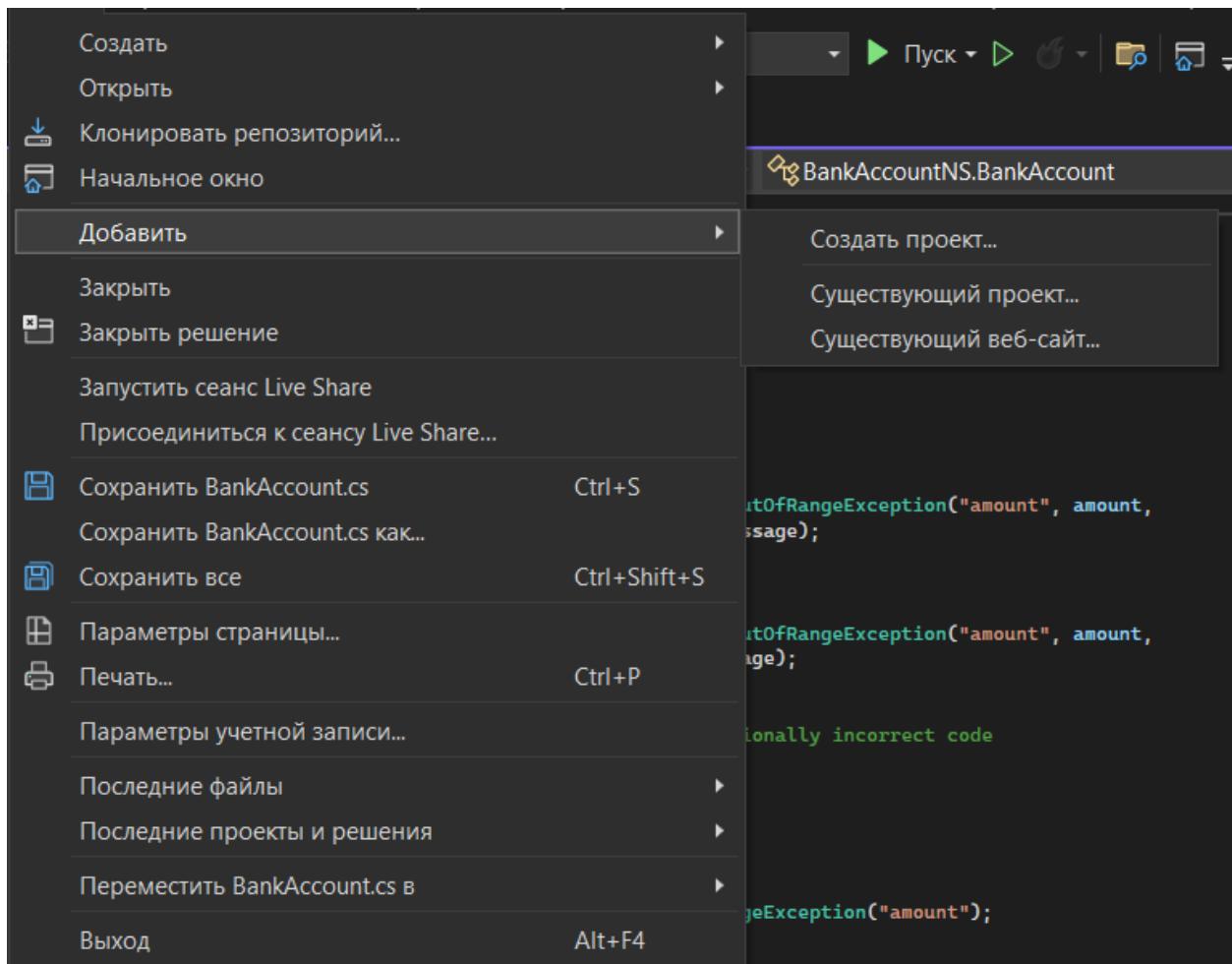


Рис. 2.1

2. Настройка файла

В поисковую строку вводим «Проекта модульного теста», а язык выбираем C# - называем его «BankTests» и создаем. Теперь нам нужно добавить ссылку на наш проект, для этого мы нажимаем на «Проект» - «Добавить ссылку» - «Проекты» и ставим галочку на нашем проекте и нажимаем «OK». (Смотрите Рис.2.2 – 2.3)

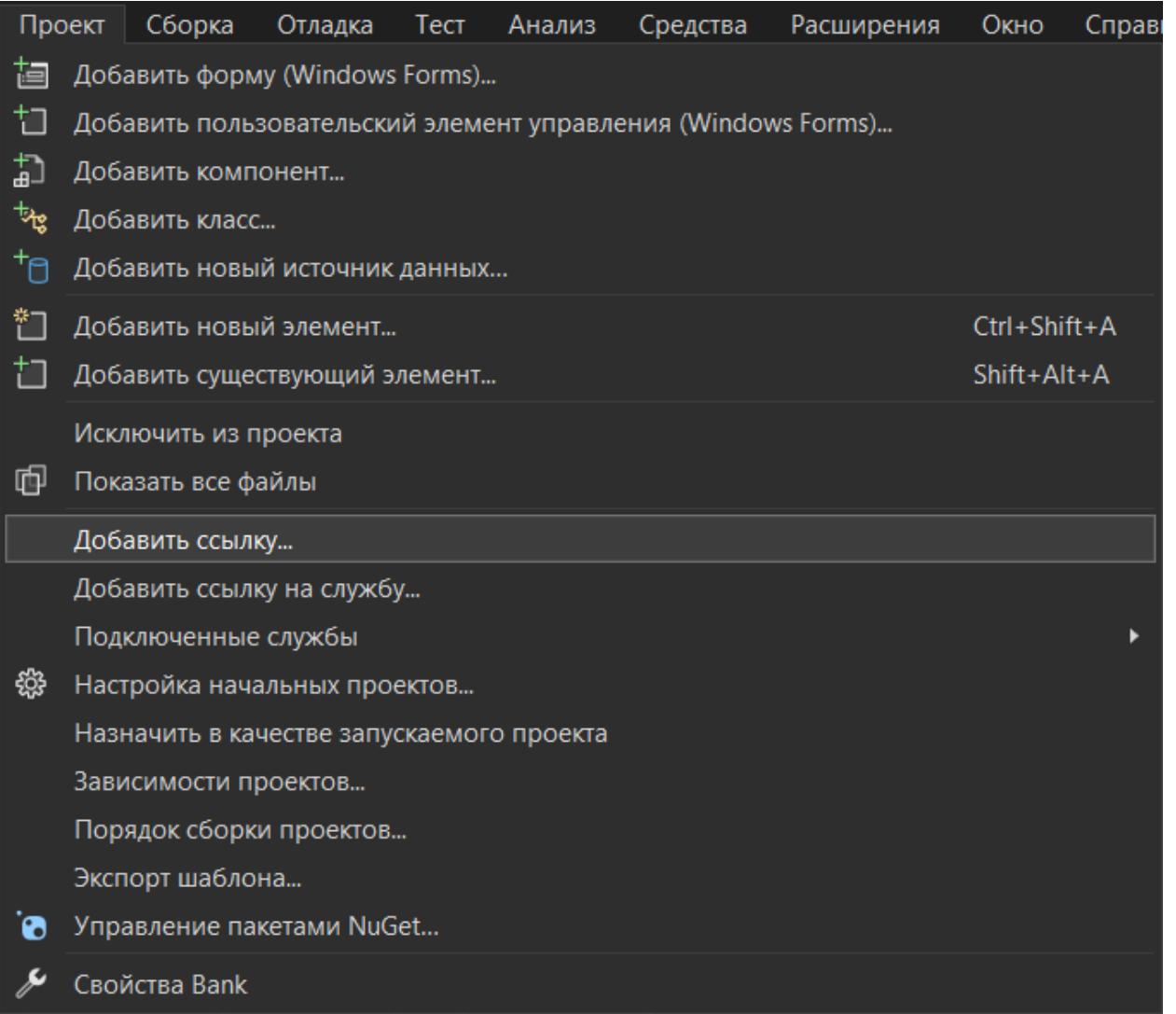


Рис. 2.2

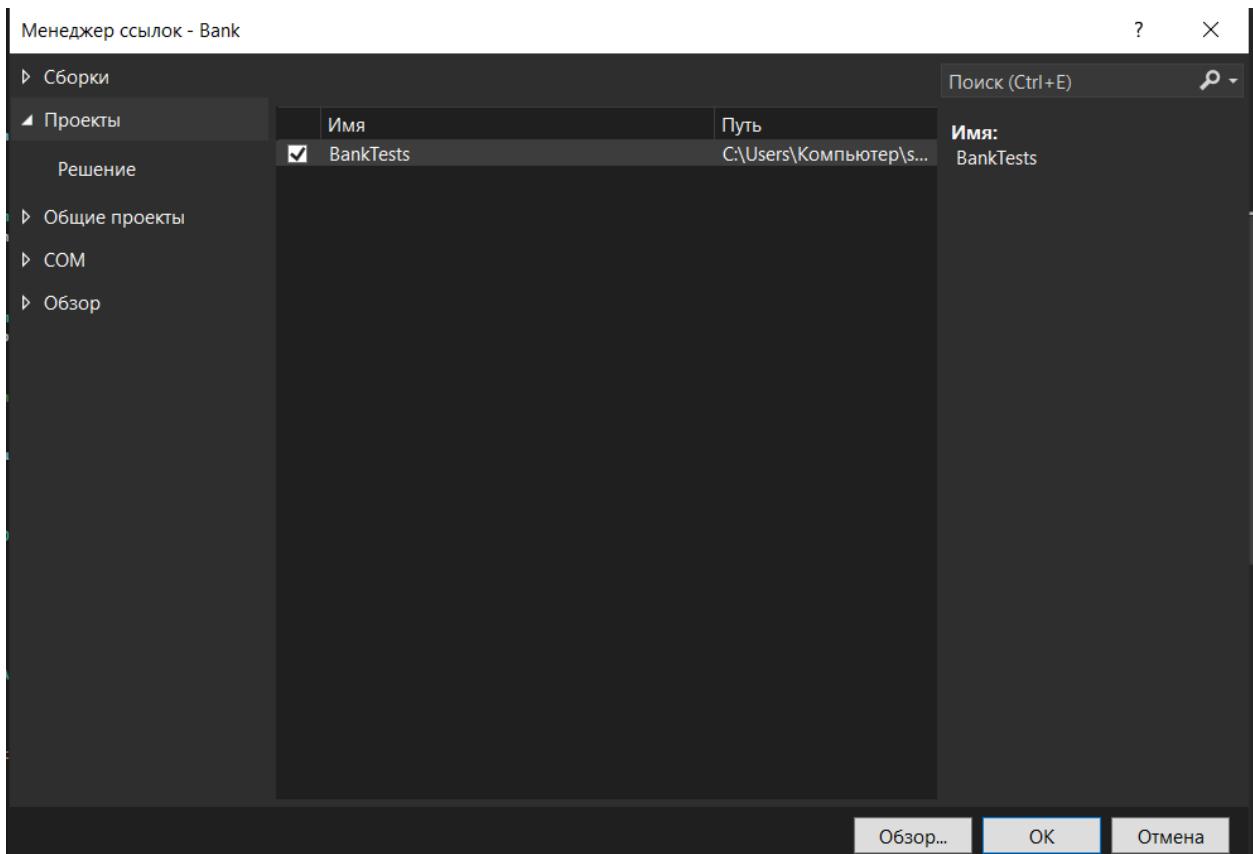


Рис. 2.3

Теперь переименуем файл «UnitTest1.cs» в «BankAccountTests.cs» и вставим туда следующий 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");
}

[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));
}

[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);
        return;
    }
    Assert.Fail("The expected exception was not thrown.");
}
}

```

}

Снова собираем решение (**CTRL + SHIFT + B**).

Тестирование

Чтобы протестировать наш файл нам нужно открыть «Обозреватель тестов» - вкладка «Тест» - «Обозреватель тестов». (Смотрите Рис. 3.1)

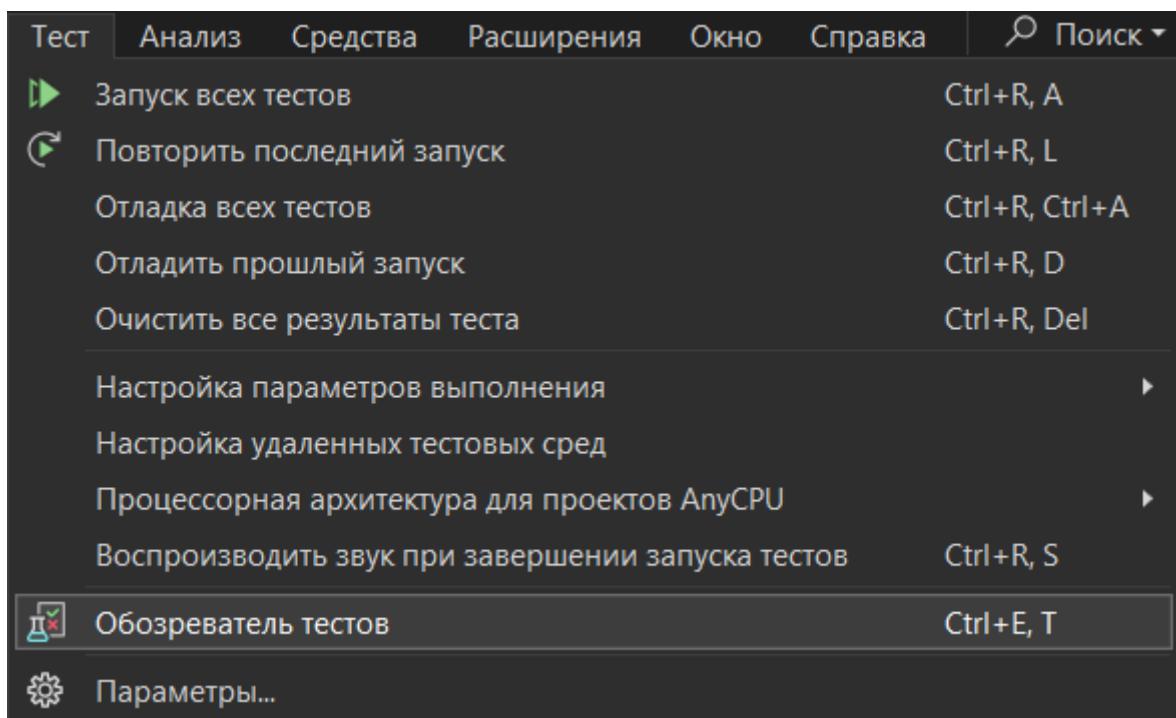


Рис. 3.1

После этих действий у нас откроется окно тестирования с результатом выполнения наших тестов, что бы их проверить нам нужно нажать на зеленую кнопку в левом верхнем углу окна, после чего он протестирует наш файл и покажет их результат. В нашем случае все результаты положительные. (Смотрите Рис. 3.2)

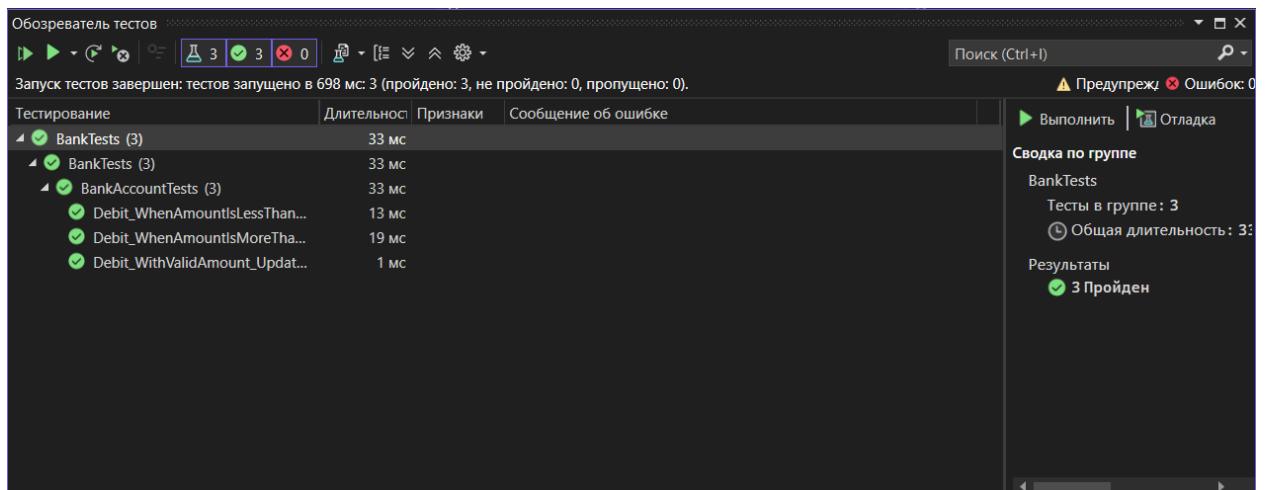


Рис. 3.2