Оглавление

[1. СОЗДАНИЕ И ЗАПУСК МОДУЛЬНЫХ ТЕСТОВ ДЛЯ УПРАВЛЯЕМОГО КОДА 3](#_Toc152526590)

[1.1. Создание проекта для тестирования 3](#_Toc152526591)

[1.2. Создание проекта модульного теста, тестового класса и метода теста 3](#_Toc152526592)

[1.3. Сборка и запуск тестов 3](#_Toc152526593)

[1.4. Анализ результатов модульного тестирования и тестируемого кода 3](#_Toc152526594)

[2. СОЗДАНИЕ МОДУЛЬНОГО ТЕСТА, УПРАВЛЯЕМОГО ДАННЫМИ 4](#_Toc152526595)

[2.1. Создание тестируемого метода и метода теста 4](#_Toc152526596)

[2.2. Анализ покрытия кода тестами 4](#_Toc152526597)

[3. СОЗДАНИЕ, ИЗМЕНЕНИЕ И ОБСЛУЖИВАНИЕ ЗАКОДИРОВАННОГО ТЕСТА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА 5](#_Toc152526598)

[3.1. Создание приложения WPF 5](#_Toc152526599)

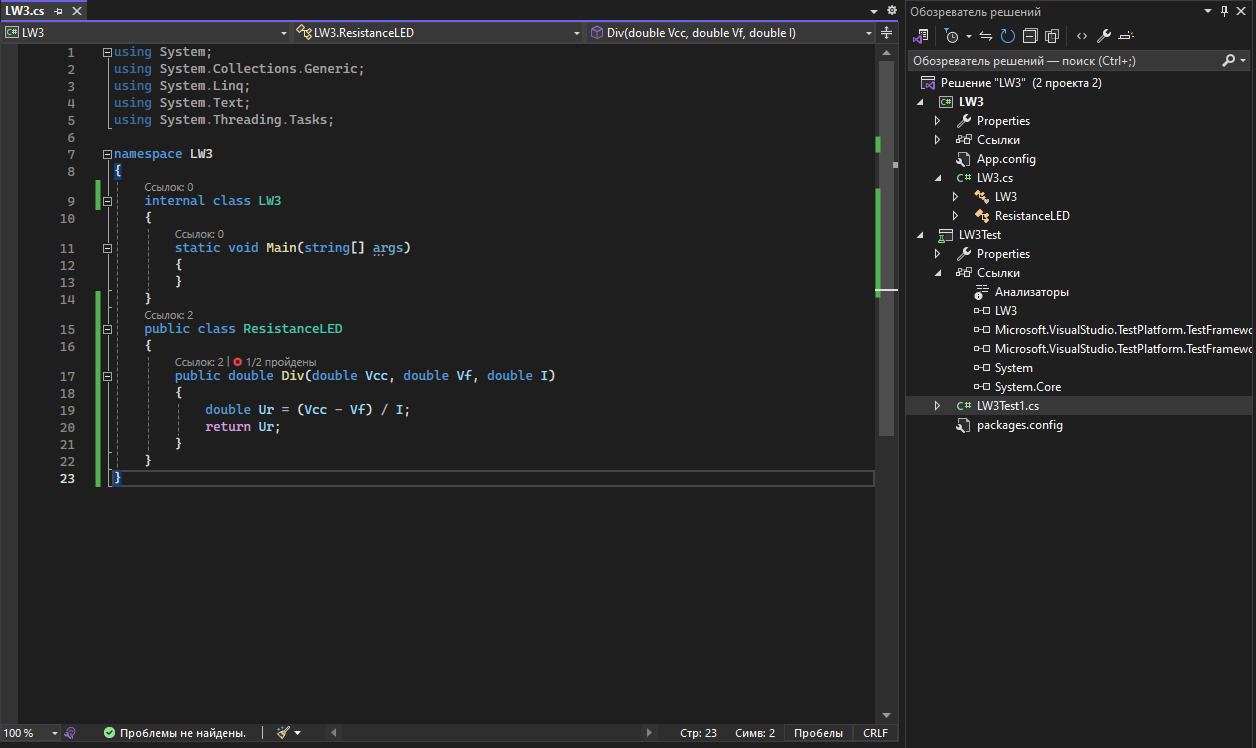
[3.2. Создание ярлыка для приложения WPF и закодирование теста пользовательского интерфейса 5](#_Toc152526600)

# СОЗДАНИЕ И ЗАПУСК МОДУЛЬНЫХ ТЕСТОВ ДЛЯ УПРАВЛЯЕМОГО КОДА

## Создание проекта для тестирования

Для проведения тестирования создаю проект «Laboratory\_work\_3». Проект отобразился в обозревателе решений.

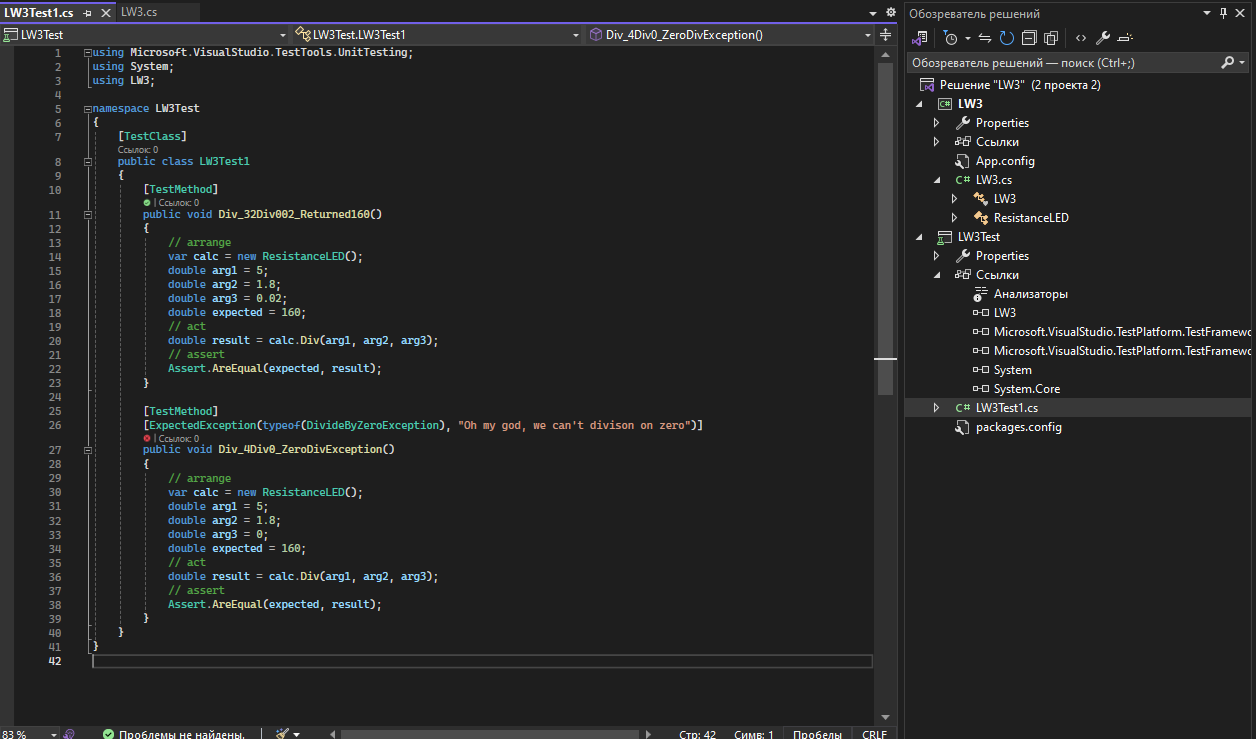
Основной файл проекта Program.cs переименую в LaboratoryWork3.cs и наполню содержимым основной файл проекта (рис. 1).

  
Рисунок 1. Проект для тестирования

Таким образом получился проект с методами для тестирования. Тестирование будет проводиться на методах **ResistanceLED**.

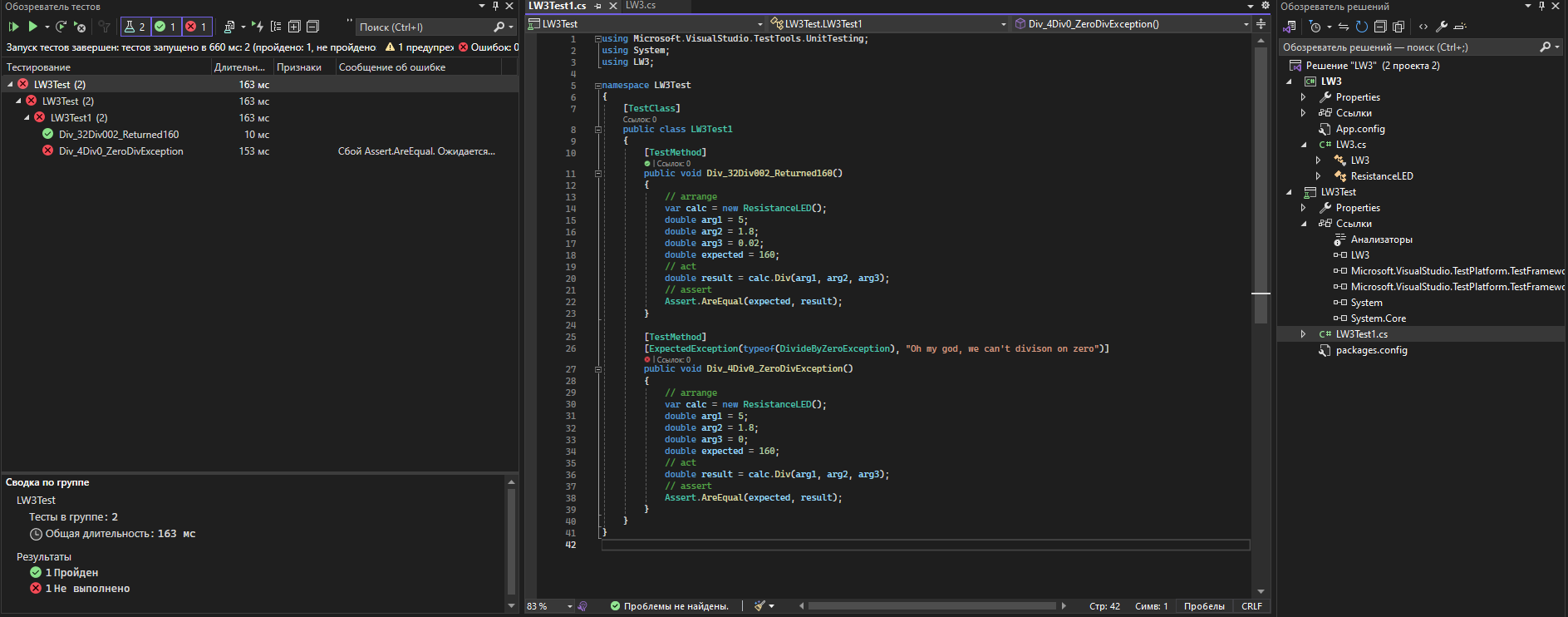
## Создание проекта модульного теста, тестового класса и метода теста

Для реализации тестирования в проект был добавлен модульный тест LW3Test. В модульный тест добавлена ссылка на основной проект LW3.

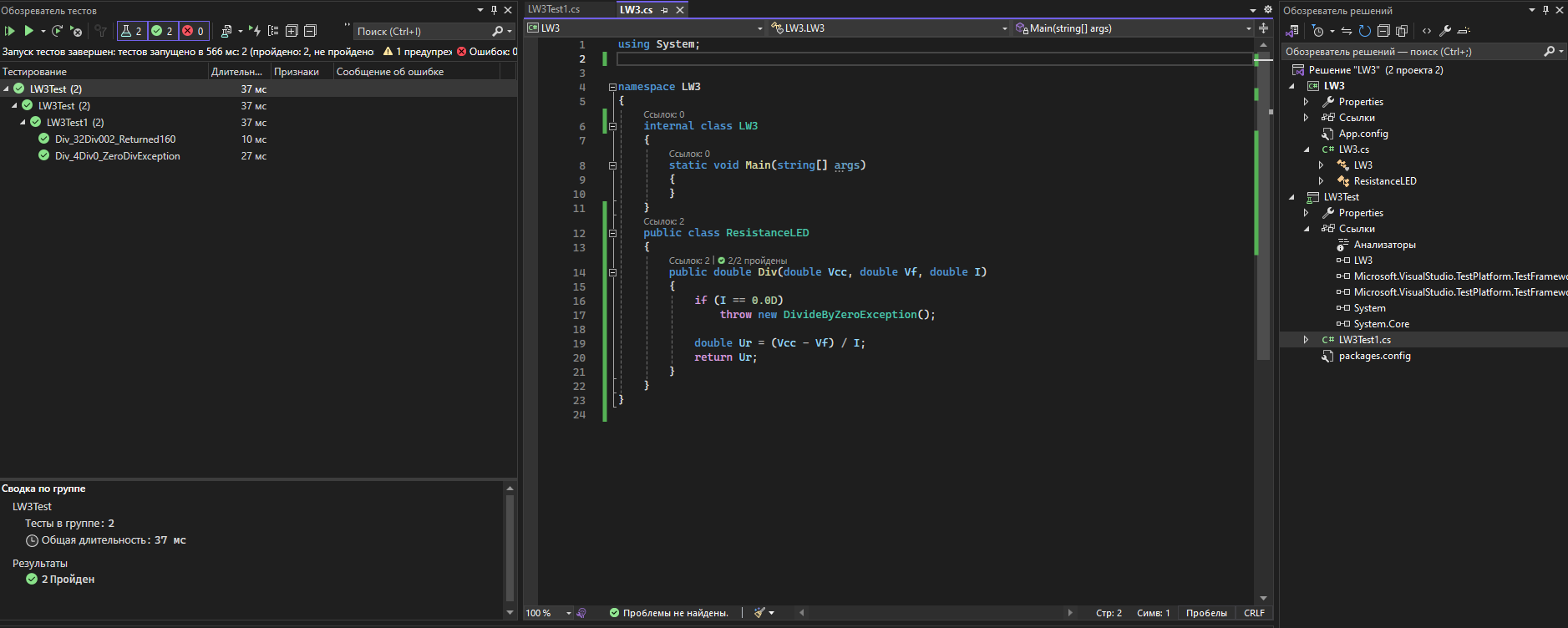
  
Рисунок 2. Методы для тестирования

## Сборка и запуск тестов

Собрав решение запускаю тестирование и получаю результат (рис. 3), на котором видно, что код не приводит к ошибке при делении на нуль.

  
Рисунок 3. Первый запуск тестов

Для решение выявленного недостатка добавлю обработчик исключений, который будет срабатывать при делители равном нулю (рис. 4)

  
Рисунок 4. Результат выполнения тестов после рефакторинга кода

## Анализ результатов модульного тестирования и тестируемого кода

При написание тестируемого метода не была предусмотрена возможность отлова деления на ноль, что привело к неуспешности тестирования в первый раз. Но при добавлении в тестируемый метод обработчика исключений тесты были пройдены с первого раза.

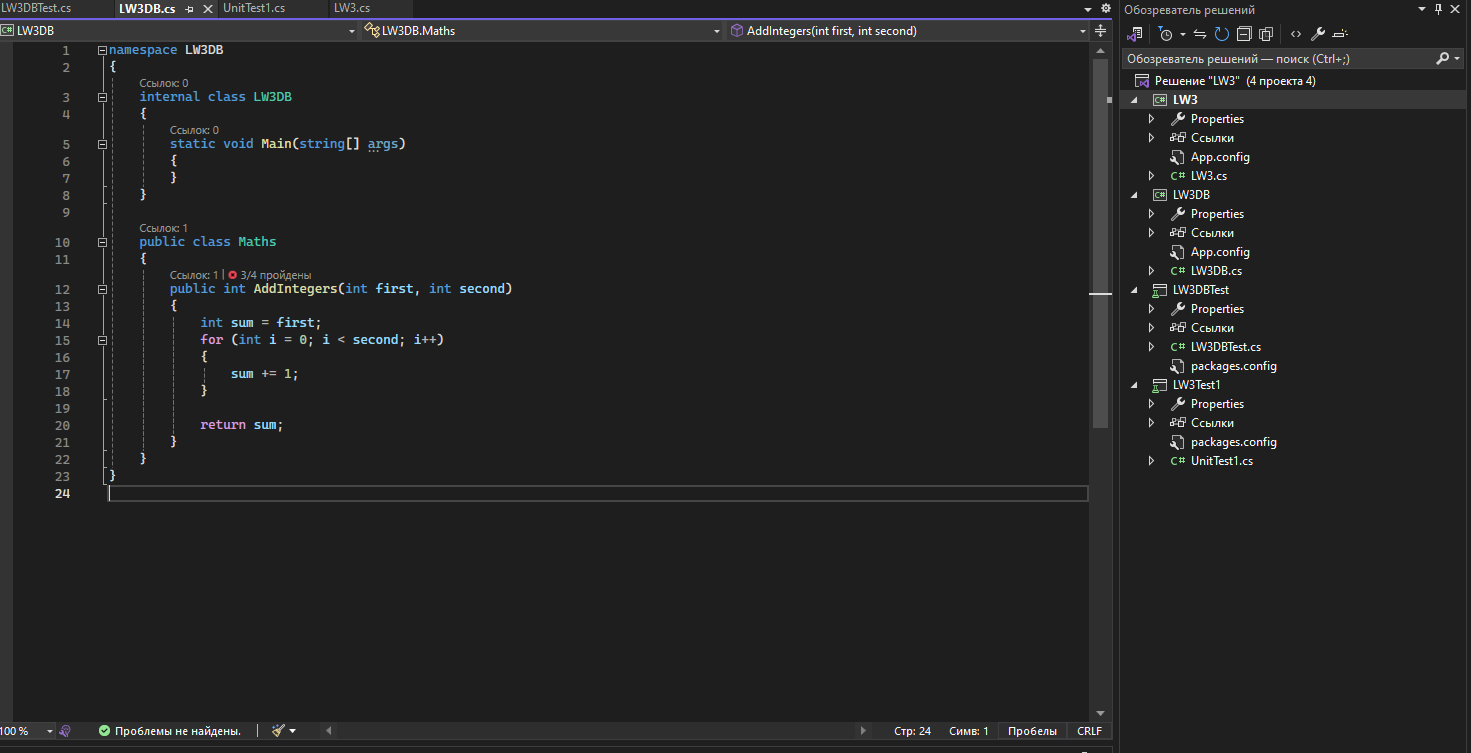
# СОЗДАНИЕ МОДУЛЬНОГО ТЕСТА, УПРАВЛЯЕМОГО ДАННЫМИ

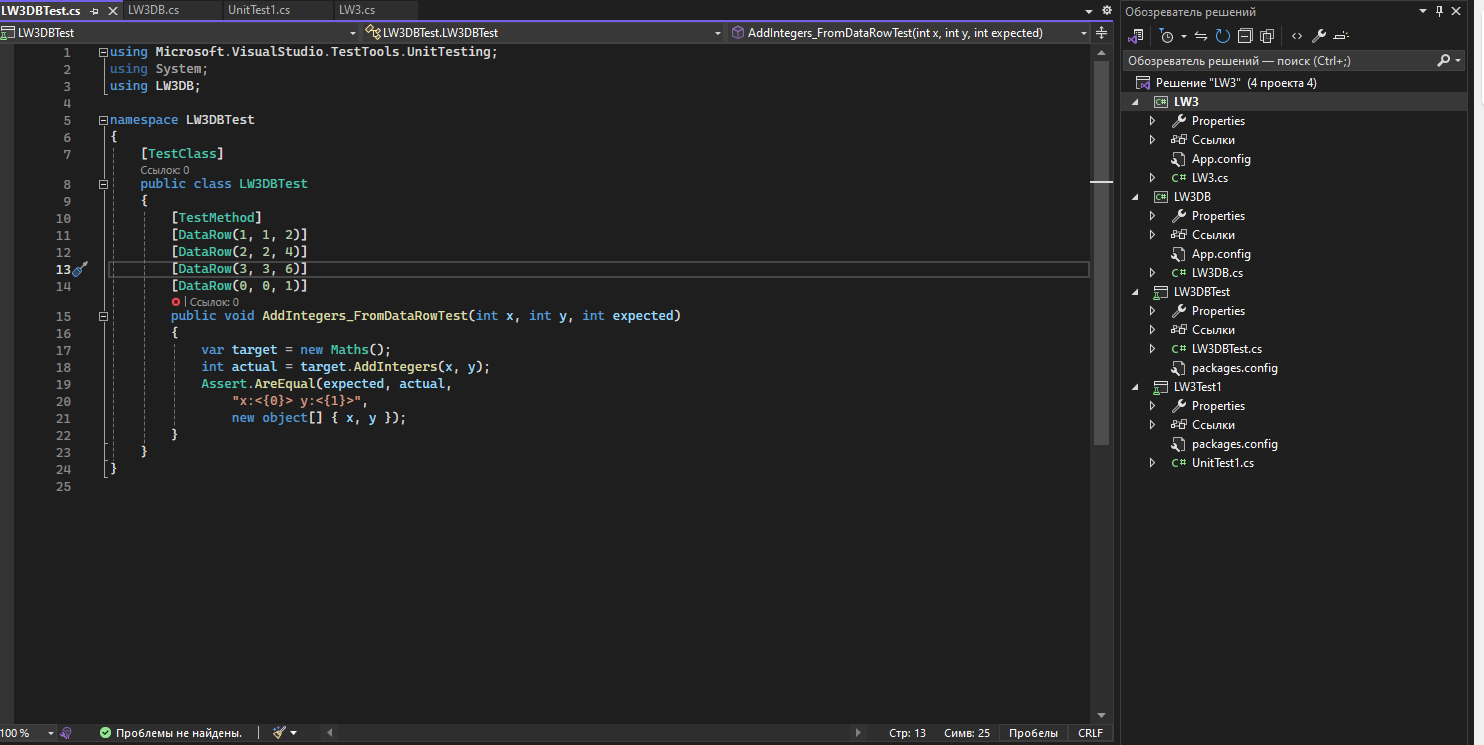
## Создание тестируемого метода и метода теста

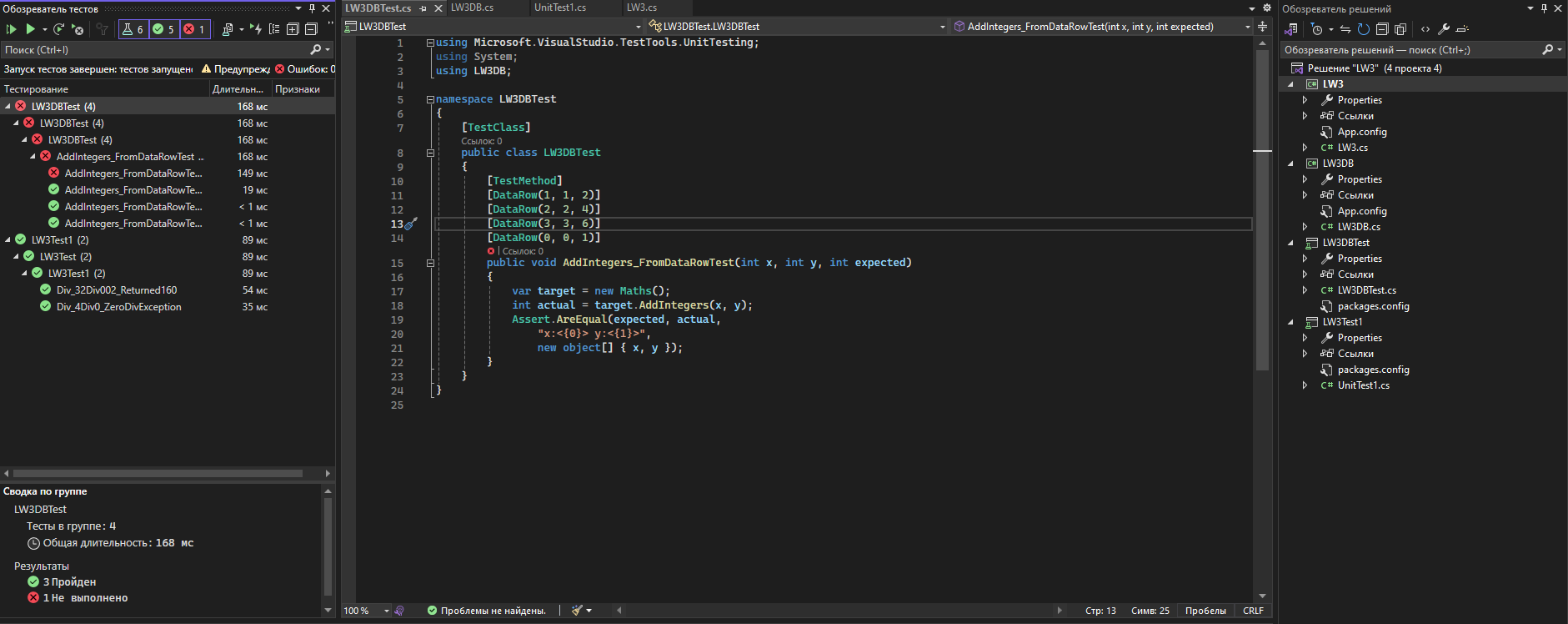
Для создания теста, управляемого данными, необходимо подготовить проект внутри решения, который будет носить название «**LW3DB**». Проект будет состоять из одного метода, который циклично считает сумму двух целых чисел (рис. 5).

Для того чтобы проверить функциональность метода класса был подготовлен модульный тест с именем «LW3DBTest», который содержит метод под названием **DataRowTest** (рис. 6).

После успешной сборки решения было запущено тестирование, из четырёх заявленных тестов успешно завершены три (рис. 7). Это связанно с тем, что последний тест передаёт ошибочные данные.

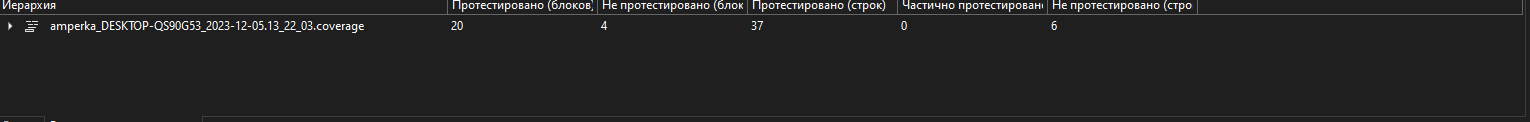
  
Рисунок 5. Метол проекта LW3DB

  
Рисунок 6. Метод DataRowTest модульного теста LW3DBTest

  
Рисунок 7. Результат выполнения тестов

## Анализ покрытия кода тестами

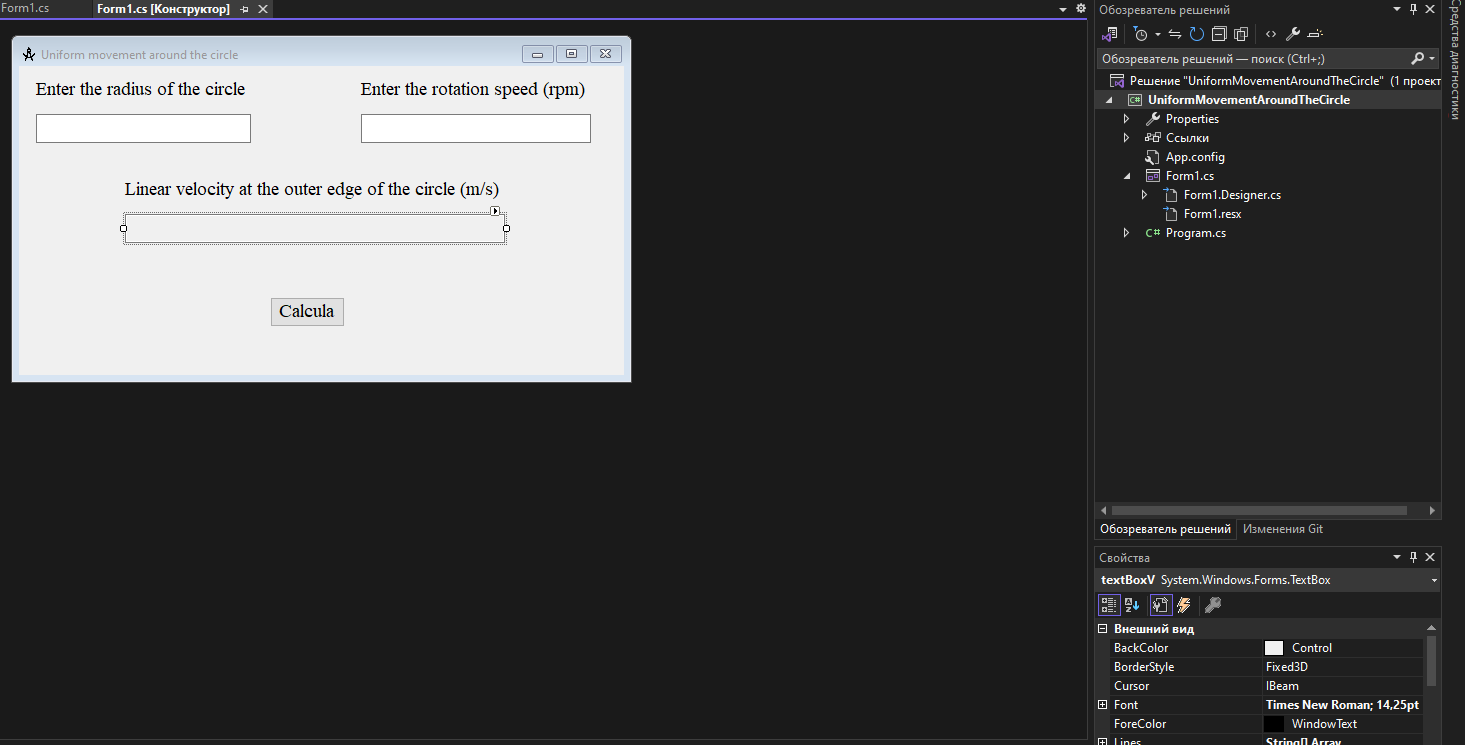
Исходя из анализа покрытия кода тестами – не протестированным остались системные строки кода (инициализация библиотек и блоки кода, содержащий библиотеки), в остальном тесты покрывают весь объём методов проекта.

  
Рисунок 8. Анализ покрытия тестами

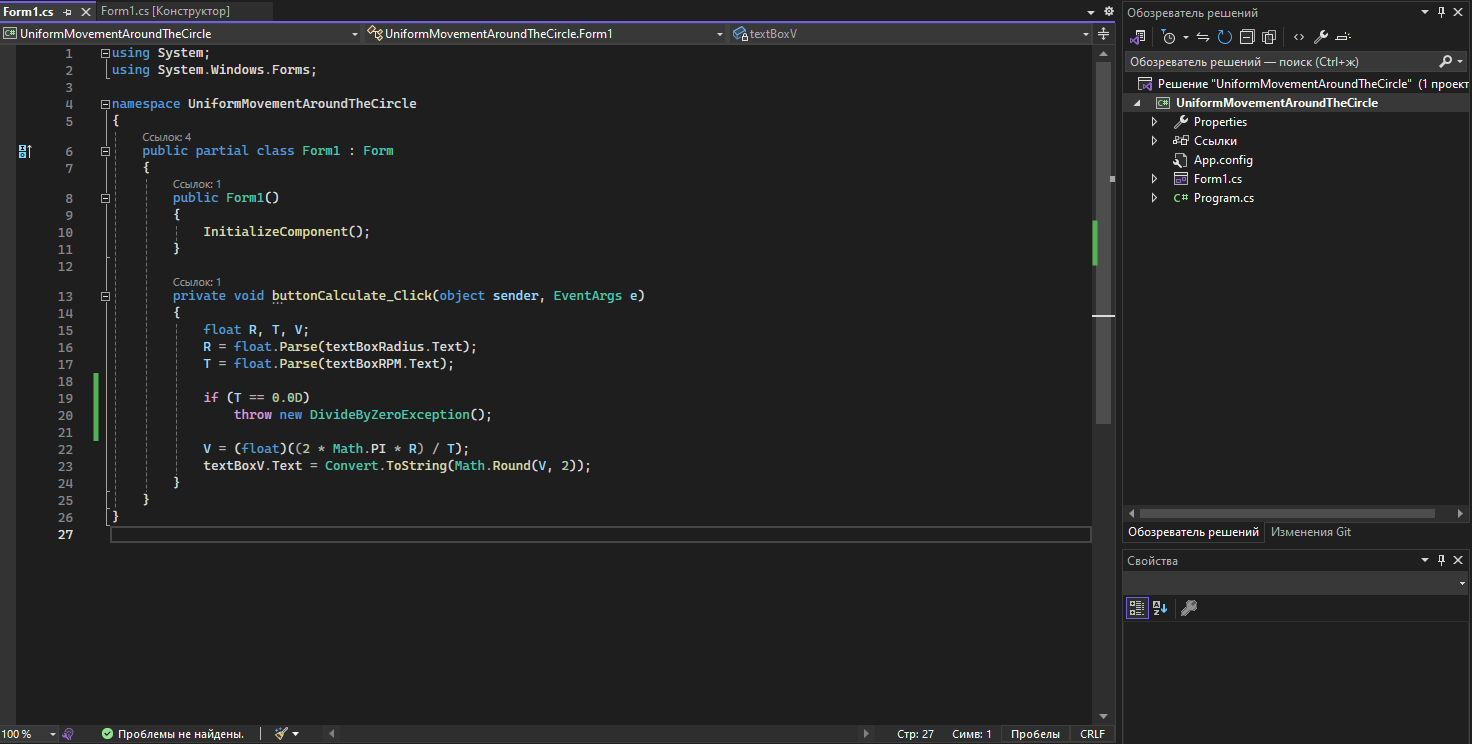
# СОЗДАНИЕ, ИЗМЕНЕНИЕ И ОБСЛУЖИВАНИЕ ЗАКОДИРОВАННОГО ТЕСТА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

## Создание приложения WPF

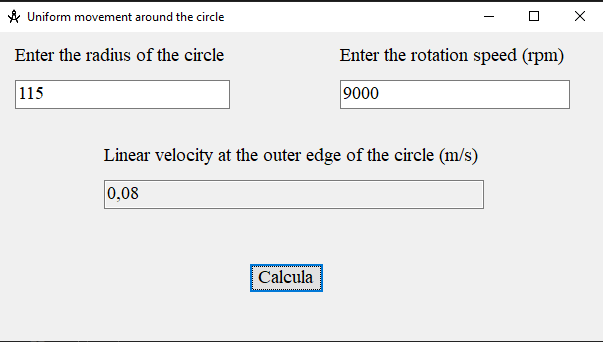
Создам новый проект и назову его «**UniformMovementAroundTheCircle**». По средствам элементов Label, TextBox и Button сформирую пользовательский интерфейс (рис. 9).

  
Рисунок 9. Пользовательский интерфейс

После создания пользовательского интерфейса создам обработчик нажатия на кнопку (рис. 10).

  
Рисунок 10. Обработчик нажатия на копку

После чего проведу тестирование кода (рис. 11).

  
Рисунок 11. Проверка работоспособности кода

## Создание ярлыка для приложения WPF и закодирование пользовательского интерфейса

Для создания ярлыка приложения **UniformMovementAroundTheCircle** на рабочем столе необходимо было использовать команду копирования и вставки ярлыка, итог которого показан на скриншоте ниже (рис. 12).

