МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Казанский национальный исследовательский технический университет

им. А.Н. Туполева – КАИ»

Институт компьютерных технологий и защиты информации

Отделение СПО ИКТЗИ (Колледж информационных технологий)

ЛАБОРАТОРНАЯ РАБОТА № 1

по дисциплине

Поддержка и тестирование программных модулей

Тема: «Unit-тестирование методов сортировки»

Работу выполнил

Студент гр.4335

Желваков А.С.

Принял

Преподаватель Максимов Р.С.

Казань 2025

**1. Цель работы:** изучить возможность создания автоматических тестов, для модульного тестирования.

**2. Задание на лабораторную работу:**

**Задание 1.** Разработать программу, содержащую графический интерфейс, для работы с тремя модулями.

**Задание 2.**  В каждом модуле, должен находиться один метод сортировки числовой последовательности (массива).

**Задание 3.** Разработать unit-тесты для каждого модуля.

**3. Результат выполнения работы:**

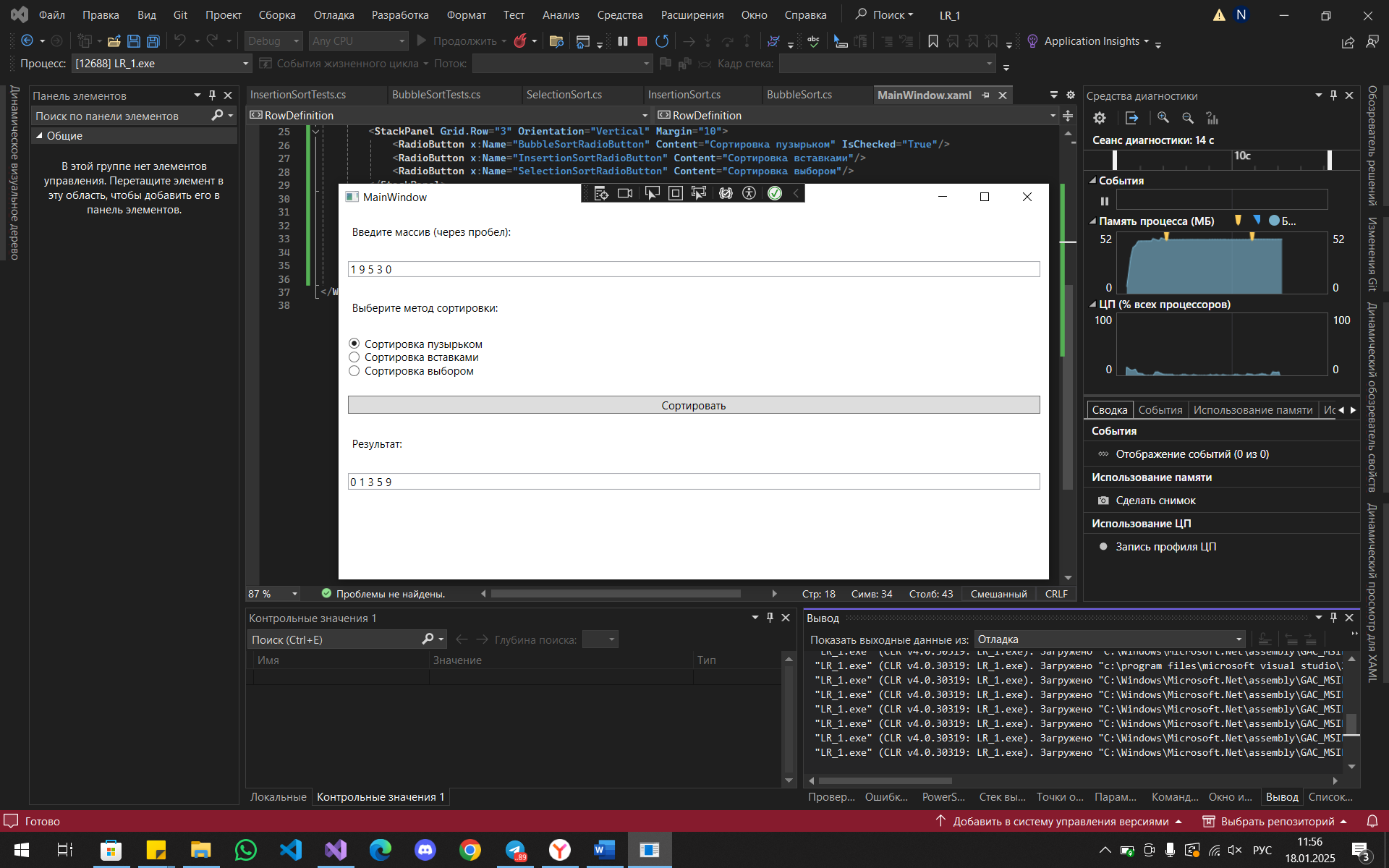


Рисунок 1 – Окно при запуске

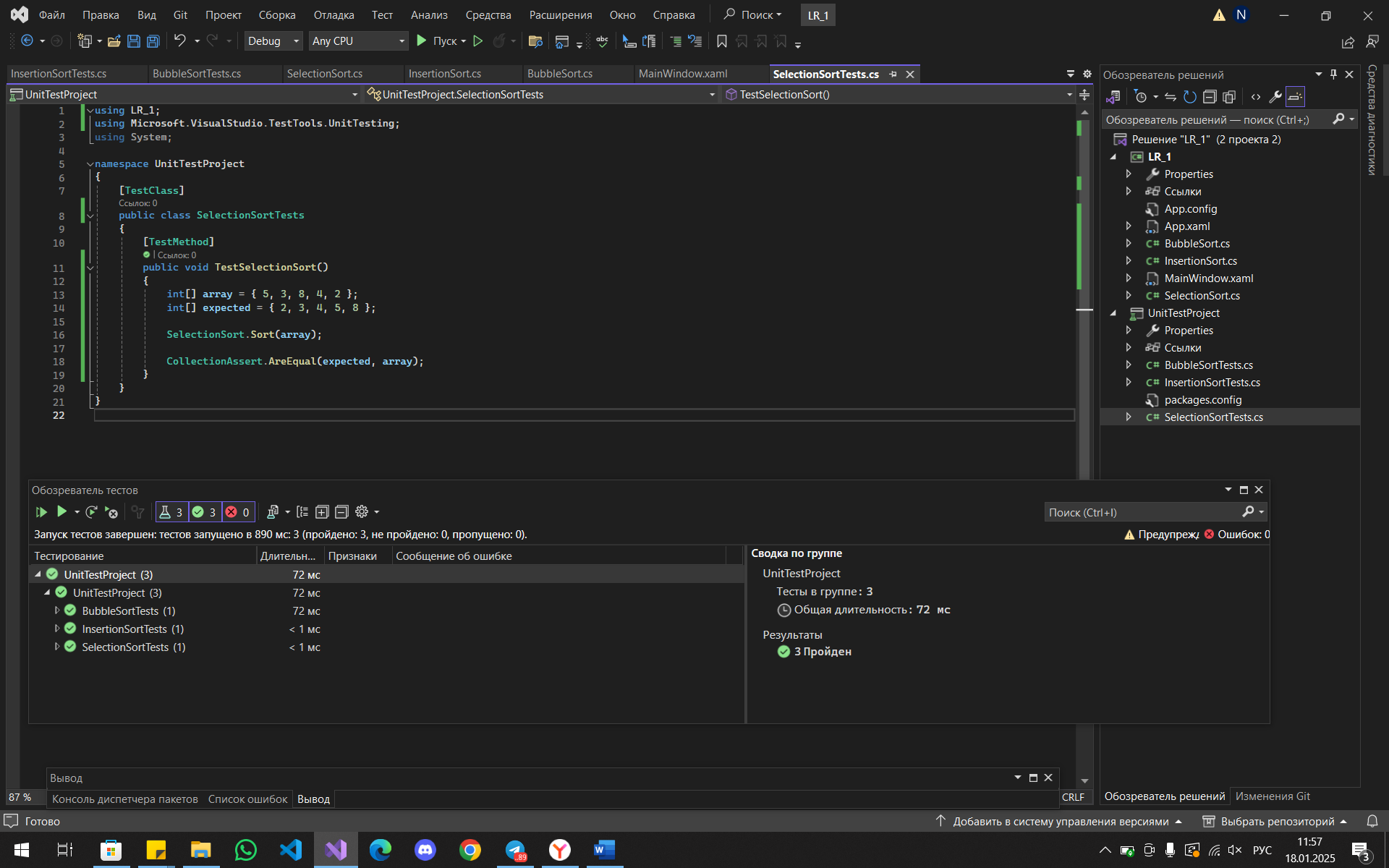


Рисунок 2 – Проверка тестов

**Ответы на контрольные вопросы**

*1) Модульное тестирование – это?*

Модульное тестирование, или юнит-тестирование (англ. unit testing) — процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы.

*2) Что делает метод Assert AreEqual?*

Метод Assert.AreEqual используется в модульных тестах для проверки равенства двух значений. Если значения равны, тест считается пройденным. Если значения не равны, тест завершается с ошибкой

*3) Какие еще методы есть у класса Assert, опишите их?*

Класс Assert содержит множество методов для проверки различных условий. Вот основные из них:

**-Assert.IsTrue(bool condition)**

Проверяет, что условие condition истинно. Если условие ложно, тест проваливается.

-**Assert.IsFalse(bool condition)**

Проверяет, что условие condition ложно. Если условие истинно, тест проваливается.

-**Assert.IsNull(object obj)**

Проверяет, что объект obj равен null. Если объект не null, тест проваливается.

-**Assert.IsNotNull(object obj)**

Проверяет, что объект obj не равен null. Если объект null, тест проваливается.

-**CollectionAssert.AreEqual**

Проверяет, что две коллекции содержат одинаковые элементы в одинаковом порядке.

*4) Сложности, возникшие при разработке модулей?*

При разработке модулей и написании тестов могут возникнуть следующие сложности:

1. **Неправильное понимание требований:** если требования к модулю неясны или неполны, это может привести к ошибкам в реализации и тестах.
2. **Граничные случаи:** трудно предусмотреть все возможные граничные случаи (например, пустые входные данные, отрицательные числа, очень большие значения).
3. **Зависимости между модулями:** если модуль зависит от других модулей или внешних систем (например, базы данных), это может усложнить тестирование.
4. **Ошибки в тестах:** тесты могут быть написаны неправильно, например, проверять не те данные или не учитывать все возможные сценарии.
5. **Рефакторинг кода:** при изменении кода модуля может потребоваться переписывание тестов, что увеличивает время разработки.
6. **Производительность тестов:** если тесты выполняются медленно (например, из-за интеграции с внешними системами), это может замедлить процесс разработки.
7. **Сложность отладки:** если тест проваливается, может быть сложно определить, в чем именно проблема: в коде модуля или в самом тесте.

**Листинг**

*BubbleSortTests.cs*

using LR\_1;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

namespace UnitTestProject

{

[TestClass]

public class BubbleSortTests

{

[TestMethod]

public void TestBubbleSort()

{

int[] array = { 5, 3, 8, 4, 2 };

int[] expected = { 2, 3, 4, 5, 8 };

BubbleSort.Sort(array);

CollectionAssert.AreEqual(expected, array);

}

}

}

*InsertionSortTests.cs*

using LR\_1;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

namespace UnitTestProject

{

[TestClass]

public class InsertionSortTests

{

[TestMethod]

public void TestInsertionSort()

{

int[] array = { 5, 3, 8, 4, 2 };

int[] expected = { 2, 3, 4, 5, 8 };

InsertionSort.Sort(array);

CollectionAssert.AreEqual(expected, array);

}

}

}

*SelectionSortTests.cs*

using LR\_1;

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

namespace UnitTestProject

{

[TestClass]

public class SelectionSortTests

{

[TestMethod]

public void TestSelectionSort()

{

int[] array = { 5, 3, 8, 4, 2 };

int[] expected = { 2, 3, 4, 5, 8 };

SelectionSort.Sort(array);

CollectionAssert.AreEqual(expected, array);

}

}

}