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 в ИКТЗИ (Колледж информационных технологий)

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине

Поддержка и тестирование программных модулей

# Тема: «Интеграционное тестирование»

Работу выполнил

Студент гр.4338

Бусов В.Р.

Принял

Преподаватель

Максимов Р. С.

Казань 2025

**Цель работы**

Изучить возможность создания автоматических тестов, для модульного тестирования.

**Задание на лабораторную работу**

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

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

3. Выполнить интеграционное тестирование.

**Ход выполнения лабораторной работы**

Была разработана программа, сортирующая массив разными видами сортировки. На рисунке 1 демонстрация работы приложения, сортирующего методами пузырьковой сортировки, быстрой сортировкой и шелла.

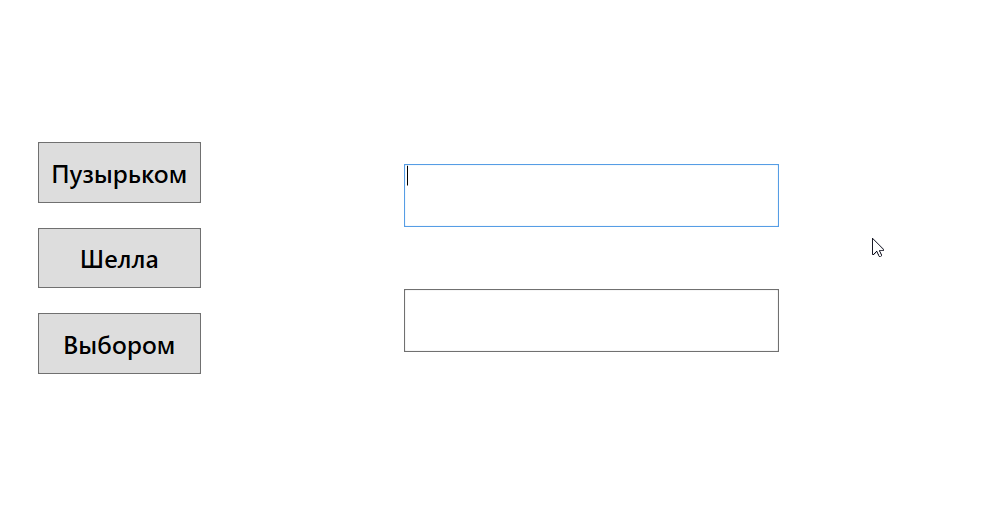


Рисунок 1 – интерфейс разработанной программы

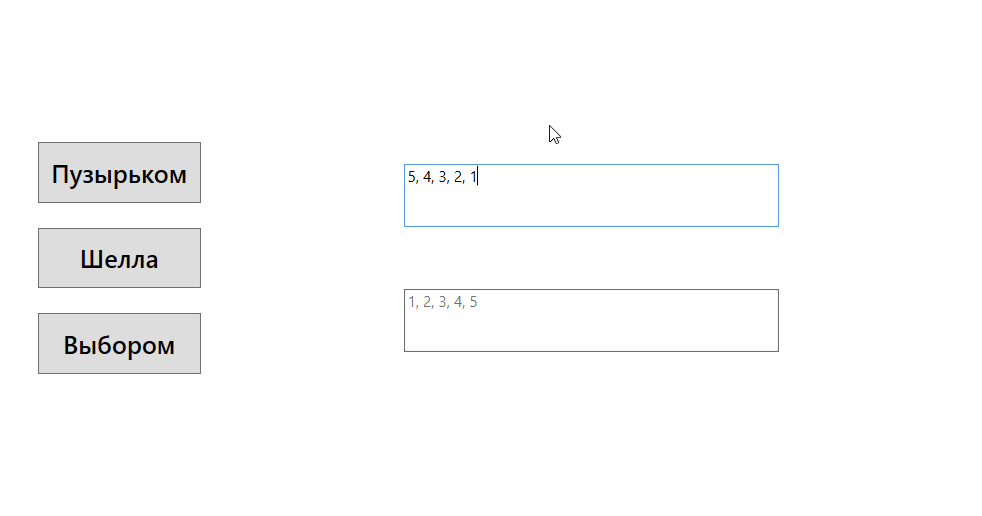


Рисунок 2 – Сортировка методом пузырька



Рисунок 3 – Сортировка методом Шелла

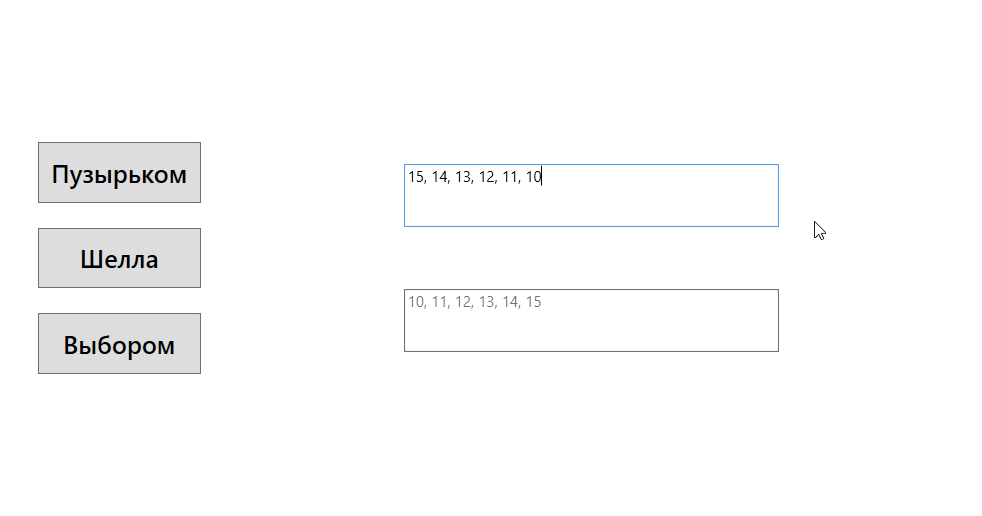


Рисунок 4 – Сортировка методом выбора

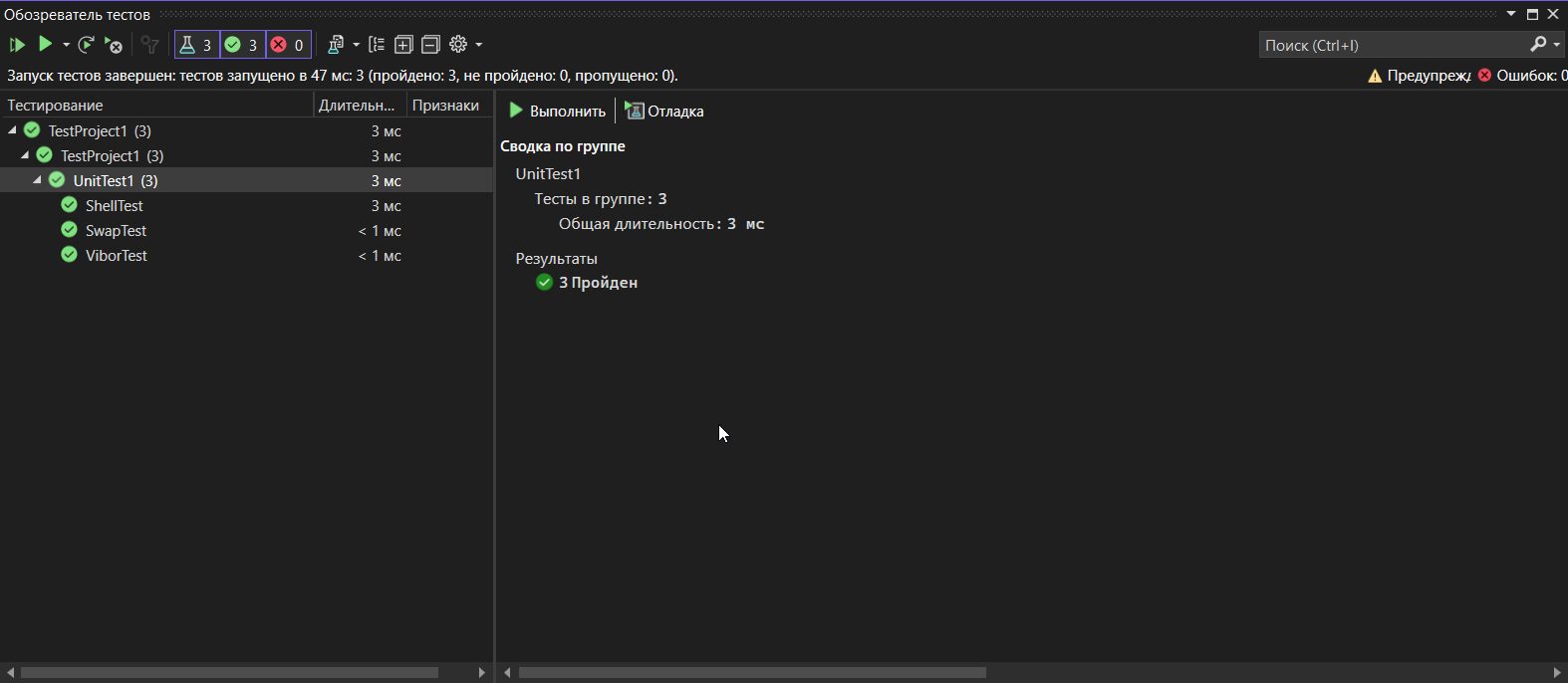


Рисунок 5 – Результат unit-тестирования

После того как программа была разработана, необходимо провести интеграционное тестирование. Были созданы 3 теста проверяющие каждый из методов сортировок. В модули подается неотсортированный массив, состоящий из произвольных значений, проходит через модуль и проверяет, возвращает ли он ожидаемое значение.

**Листинг**

using WpfApp2;

namespace TestProject1

{

[TestClass]

public class UnitTest1

{

[TestMethod]

public void ShellTest()

{

var unsorted\_array = NumGenerator.Generate(20);

unsorted\_array.Sort();

var expected = unsorted\_array;

ShellSort sorter = new ShellSort();

List<int> actual = sorter.sort(unsorted\_array);

CollectionAssert.AreEqual(expected, actual);

}

[TestMethod]

public void SwapTest()

{

var unsorted\_array = NumGenerator.Generate(20);

unsorted\_array.Sort();

var expected = unsorted\_array;

SwapSort sorter = new SwapSort();

var actual = sorter.sort(unsorted\_array);

CollectionAssert.AreEqual(expected, actual);

}

[TestMethod]

public void ViborTest()

{

var unsorted\_array = NumGenerator.Generate(20);

unsorted\_array.Sort();

var expected = NumGenerator.Generate(20);

ViborSort sorter = new ViborSort();

var actual = sorter.sort(unsorted\_array);

CollectionAssert.AreEqual(expected, actual);

}

}

}