

Министерство науки и высшего образования РФ  
Федеральное государственное образовательное учреждение  
высшего образования  
«УФИМСКИЙ УНИВЕРСИТЕТ НАУКИ И ТЕХНОЛОГИЙ»

Кафедра АСУ

## Лабораторная работа №2

по дисциплине

«Объектно-ориентированное моделирование  
и программирование»

«Настройка и использование библиотеки JUnit для модульного  
тестирования программного обеспечения»

Выполнили:  
студенты гр. ПИ-101Бзу  
Уразбахтин Т.А.  
Шаимов А.Р.

Проверил:  
преподаватель  
Казанцев А.В.

г. Уфа 2024

## Лабораторная работа №2

### Настройка и использование библиотеки JUnit для модульного тестирования программного обеспечения

Цель работы: изучение библиотеки JUnit для тестирования программного кода и создания веток, тэгов и выполнения слияния в Git.

#### Ход работы

Таблица – 1 Вариант задания

№	Модератор	Разработчик
19	Создать класс и соответствующий метод: генерирует случайные числа в диапазоне от 80 до 120.	Создать класс и соответствующий метод: реализует создание одномерного массива с 12 элементами, которые генерируются случайно, используя созданный класс разработчика 1.

Таблица – 1 Вариант задания

1. Изучили теоретическую часть.
2. Модератор добавил в проект в основной ствол зависимость JUnit 5. Выполнил свой вариант задания. Разработчик обновил рабочую копию. Результат представлен в соответствии с рисунком 1.



Рисунок 1 – Модератор выполнил свой вариант задания

3. Разработчик создал один юнит-тест для своего варианта задания. Разработчик создал ответвление основного ствола, которое называется «BranchDEV\_Urazbakhtin», внёс изменения в ответвление, согласно, варианту задания. Результат представлен в соответствии с рисунком 2.

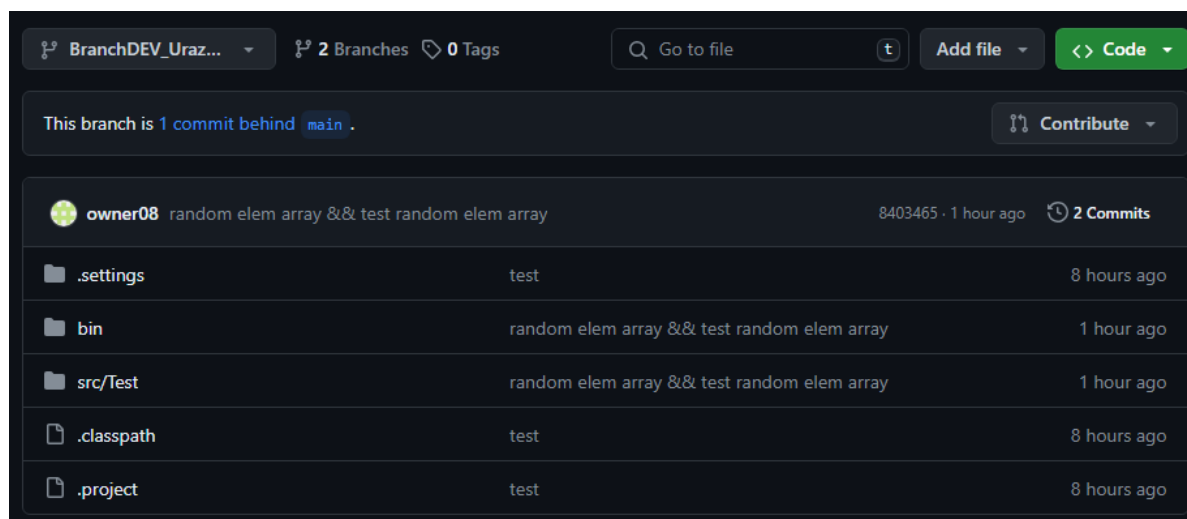


Рисунок 2 – Разработчик выполнил свой вариант задания

4. Провели слияние всех ответвлений в основной ствол. Результат представлен в соответствии с рисунком 3.

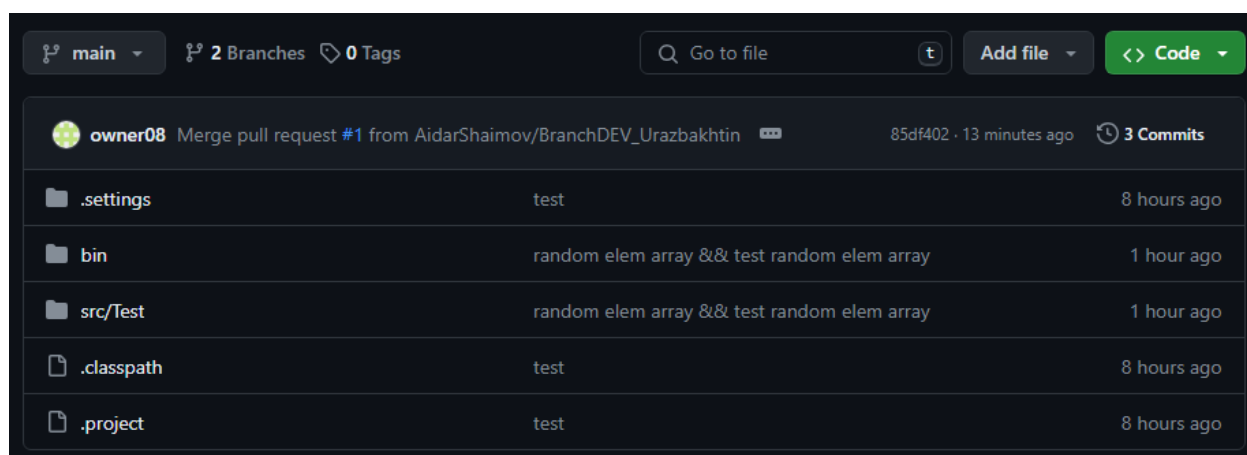
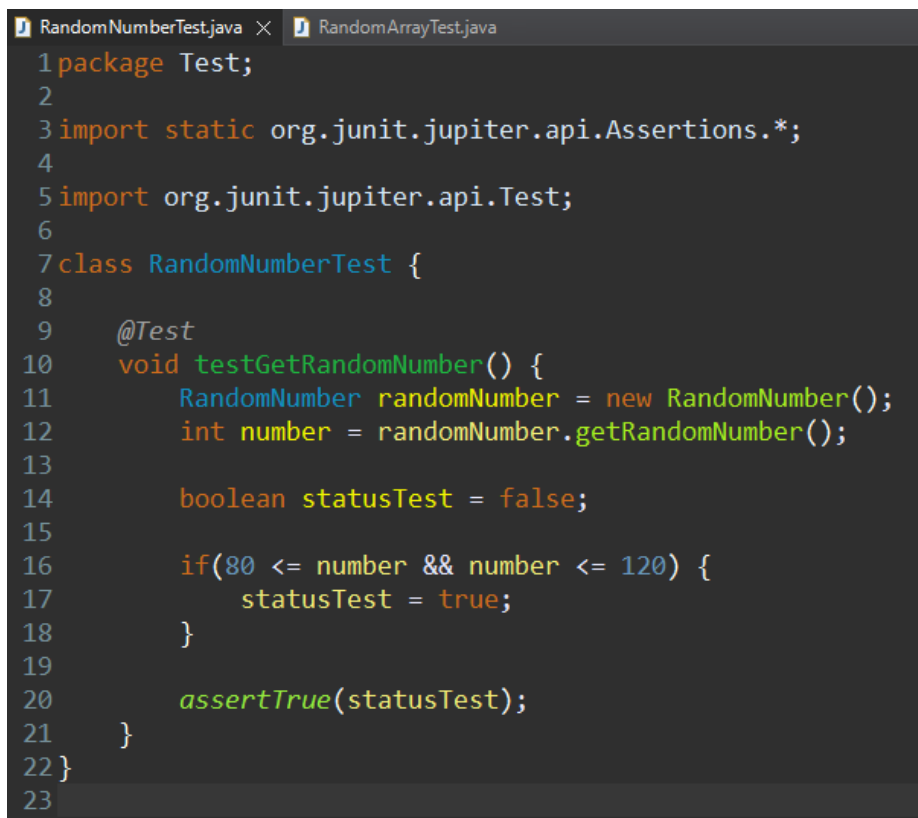


Рисунок 3 – Слияние ответвления в основной ствол

5. Модератор проверил работоспособность созданного программного обеспечения. Результат представлен в соответствии с рисунками 4 и 5.



```
1 package Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 import org.junit.jupiter.api.Test;
6
7 class RandomNumberTest {
8
9     @Test
10    void testGetRandomNumber() {
11        RandomNumber randomNumber = new RandomNumber();
12        int number = randomNumber.getRandomNumber();
13
14        boolean statusTest = false;
15
16        if(80 <= number && number <= 120) {
17            statusTest = true;
18        }
19
20        assertTrue(statusTest);
21    }
22 }
23
```

Рисунок 4 – Код теста 1

```

1 package Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 import org.junit.jupiter.api.BeforeAll;
6 import org.junit.jupiter.api.Test;
7
8 class RandomArrayTest {
9     @BeforeAll
10    public static void randomArray_null() {
11        RandomArray randomArray = new RandomArray();
12        int[] array = randomArray.getRandomArray();
13
14        assertTrue(array != null);
15    }
16
17    @Test
18    void randomArray_length() {
19        RandomArray randomArray = new RandomArray();
20        int[] array = randomArray.getRandomArray();
21
22        assertEquals(12, array.length);
23    }
24
25    @Test
26    void randomArray_nullElem() {
27        RandomArray randomArray = new RandomArray();
28        int[] array = randomArray.getRandomArray();
29
30        boolean statusTest = true;
31
32        for(int i = 0; i < array.length; i++) {
33            if(array[i] == 0) {
34                statusTest = false;
35                break;
36            }
37        }
38
39        assertTrue(statusTest);
40    }
41 }
42

```

Рисунок 5 – Код теста 2

**Заключение:** изучили библиотеки JUnit для тестирования программного кода и создания веток, тэгов и выполнения слияния в Git.

## **Контрольные вопросы.**

### **1. Что такое модульное тестирование?**

Модульное тестирование - процесс тестирования отдельного программного модуля или класса, то есть его отдельных функций или частей, с целью проверки их работоспособности и поиска ошибок.

### **2. Что такое JUnit?**

JUnit - это библиотека для модульного тестирования Java-классов. Она предоставляет инструменты для написания тестовых сценариев, управления тестовыми случаями и обработки результатов тестирования. JUnit помогает разработчикам писать более качественные и надежные программы, обнаруживая ошибки на ранних этапах разработки.

### **3. Как располагаются тесты в JUnit?**

Тесты в JUnit располагаются в специальных классах, которые наследуются от класса `org.junit.Test`. В этих классах определяются методы тестирования, которые обычно имеют названия, начинающиеся с `test`, например `testMethod()`. Чтобы выполнить все тесты в классе, необходимо создать экземпляр этого класса и вызвать метод `JUnitCore.runClasses()`.

### **4. Что такое `assert` и `assume`?**

`Assert` и `Assume` используются для проверки условий во время выполнения теста. `Assert` используется для явного указания того, что условие должно быть истинным, и если оно не выполняется, то генерируется исключение. `Assume` используется для того, чтобы пропустить проверку условия, если оно не выполнено, но при этом в отчет о тестировании записывается информация о том, что проверка была пропущена.

### **5. В чем отличие ветки от метки?**

Ветку можно сравнить с направлением движения в пространстве. Она указывает, в какую сторону будет развиваться история дальше. Метка же - это просто способ отметить определённое место в истории, чтобы можно было вернуться к нему позже.