Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления» (ИУ)

КАФЕДРА «Информационная безопасность» (ИУ8)

Отчёт

по лабораторной работе № 6
по дисциплине «Технологии и методы программирования»

Тема: «Модульное тестирование»

Вариант 4

Выполнил:
А. В. Куликова,
студент группы ИУ8-21М
Проверил: А. Ю. Быков

Москва, 2024

1. Постановка задачи

Разработать класс калькулятор для выполнения бинарных операций: умножение, деление, сложение, вычитание, возведение в степень, взятие логарифма (параметры: аргумент и основание). Пример упрощенного подобного класса (не со всеми операциями) на языке Java представлен ниже.

```
// Тестируемый класс
public class MyCalc {
	public double getSum(double x, double y)
	{
		return x+y;
	}
	public double getMul(double x, double y)
	{
		return x*y;
	}
	public double getDiv(double x, double y)
	{
		return x/y;
	}
	public double getSub(double x, double y)
	{
		return x-y;
	}
}
```

Провести модульное тестирование всех методов этого класса, для каждого метода выполнить не менее двух тестов: проверка истинного значения на равенство с результатом и проверка ложного значения на неравенство и результатом. Пример ограниченного теста на равенство с истинным значением при использовании JUnit приведен ниже.

```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;
// Тестовый класс
class MyTest {
	@Test
	void test() {
		MyCalc calc=new MyCalc();
		assertEquals(calc.getSum(2, 3), 5., "Тест суммы");
		assertEquals(calc.getMul(2, 3), 6., "Тест произведения");
		assertEquals(calc.getDiv(6, 3), 2., "Тест деления");
		assertEquals(calc.getSub(3, 2), 1., "Тест разности");
		// fail("Not yet implemented");
	}
```



```
}
```

Представить результаты тестов при отсутствии ошибок и при ошибке, для этого сознательно внести ошибку в код, например, вместо одной операции использовать другую операцию.

2. Ход работы

Листинг 1 – Код программы main.java

```
class MyCalculator {

    public static void main(String[] args) {
        MyCalculator calculator = new MyCalculator();

        System.out.println("Sum: " + calculator.add(5, 3));
        System.out.println("Difference: " + calculator.subtract(5, 3));
        System.out.println("Product: " + calculator.multiply(5, 3));

        try {
            System.out.println("Quotient: " + calculator.divide(5, 3));
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Power: " + calculator.power(2, 3));
        System.out.println("Logarithm: " + calculator.logarithm(100, 10));
    }

    public double add(double x, double y) {
        return x + y;
    }

    public double subtract(double x, double y) {
        return x - y;
    }

    public double multiply(double x, double y) {
        return x * y;
    }

    public double divide(double x, double y) {
        if (y == 0) {
            throw new IllegalArgumentException("Division by zero is not
allowed");
        }
        return x / y;
    }

    public double power(double base, double exponent) {
        return Math.pow(base, exponent);
    }

    public double logarithm(double argument, double base) {
        if (argument <= 0 || base <= 0 || base == 1) {
```

```
            throw new IllegalArgumentException("Invalid arguments for
logarithm");
        }
        return Math.log(argument) / Math.log(base);
    }
}
```

Результат:

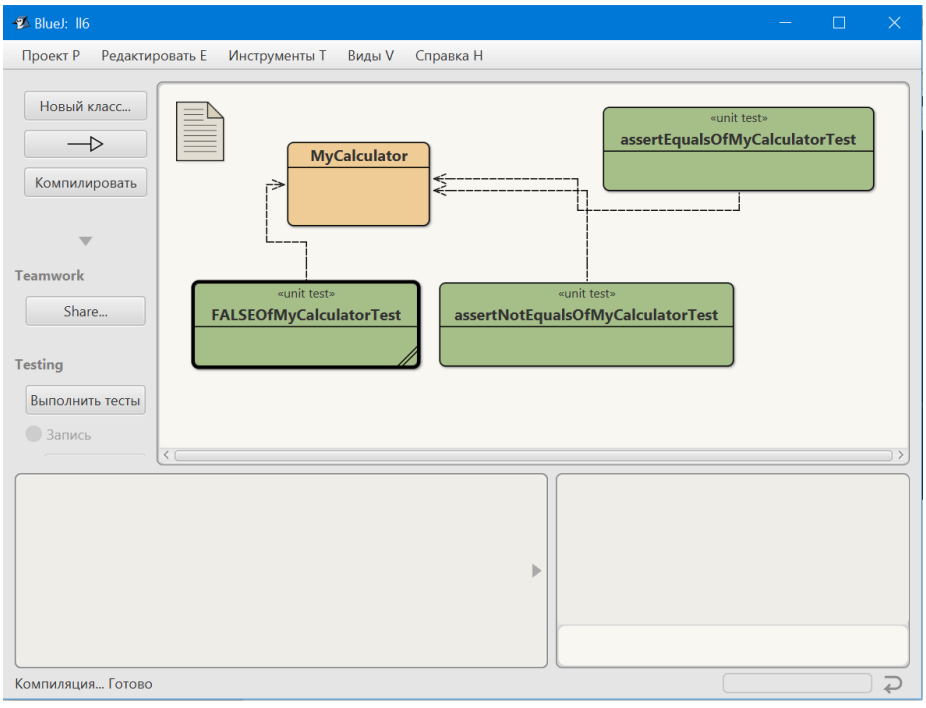
```
rver=n,suspend=y,address=localhost:57565' '@C:\Users\USER\A
ppData\Local\Temp\cp_a6f1e8i044pw5dhln844cpwgb.argfile' 'My
Calculator'
Sum: 8.0
Difference: 2.0
Product: 15.0
Quotient: 1.6666666666666667
Power: 8.0
Logarithm: 2.0
PS C:\javasing\l6\l6>
```

Листинг 2 – Код программы mainTest.java

```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;

class MyCalculatorTest {

    MyCalculator calculator = new MyCalculator();

    @Test
    public void testSubtract() {
        assertEquals(calculator.subtract(5, 3), 2.0, 0.0001, "Test subtraction
with true value");
        assertNotEquals(calculator.subtract(5, 3), 3.0, "Test subtraction with
false value");
    }

    @Test
    public void testMultiply() {
        assertEquals(calculator.multiply(2, 3), 6.0, 0.0001, "Test multiplication
with true value");
```

```
        assertNotEquals(calculator.multiply(2, 3), 5.0, "Test multiplication with
false value");
    }

    @Test
    public void testDivide() {
        assertEquals(calculator.divide(6, 3), 2.0, 0.0001, "Test division with
true value");
        assertNotEquals(calculator.divide(6, 3), 3.0, "Test division with false
value");
    }

    @Test
    public void testPower() {
        assertEquals(calculator.power(2, 3), 8.0, 0.0001, "Test power with true
value");
        assertNotEquals(calculator.power(2, 3), 9.0, "Test power with false
value");
    }

    @Test
    public void testLogarithm() {
        assertEquals(calculator.logarithm(100, 10), 2.0, 0.0001, "Test logarithm
withtrue value");
        assertNotEquals(calculator.logarithm(100, 10), 3.0, "Test logarithm
withfalse value");
    }
}
```

Результат будет проверен в другом **IDE BlueJ**



Где код теста разбит на следующие категории:

- assertEquals – ввод ***верного*** ожидания (результат - правильно)
- assertNotEquals – ввод ***верного неверного*** ожидания (результат - правильно)
- assertEquals – ввод ***неверного*** ожидания (результат - ошибка)

модуль ***FALSEOfMyCalculatorTest***:

```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class FALSEOfMyCalculatorTest
{
   MyCalculator calculator = new MyCalculator();

   @Test
   public void testSubtract() {
      assertEquals(calculator.subtract(5, 3), 3.0, "Test subtraction with false value");
   }

   @Test
   public void testMultiply() {
      assertEquals(calculator.multiply(2, 3), 5.0, "Test multiplication with false value");
   }

   @Test
   public void testDivide() {
      assertEquals(calculator.divide(6, 3), 3.0, "Test division with false value");
   }

   @Test
   public void testPower() {
      assertEquals(calculator.power(2, 3), 9.0, "Test power with false value");
   }

   @Test
   public void testLogarithm() {
      assertEquals(calculator.logarithm(100, 10), 3.0, "Test logarithm withfalse value");
   }
}
```

модуль ***assertNotEqualsOfMyCalculatorTest***:

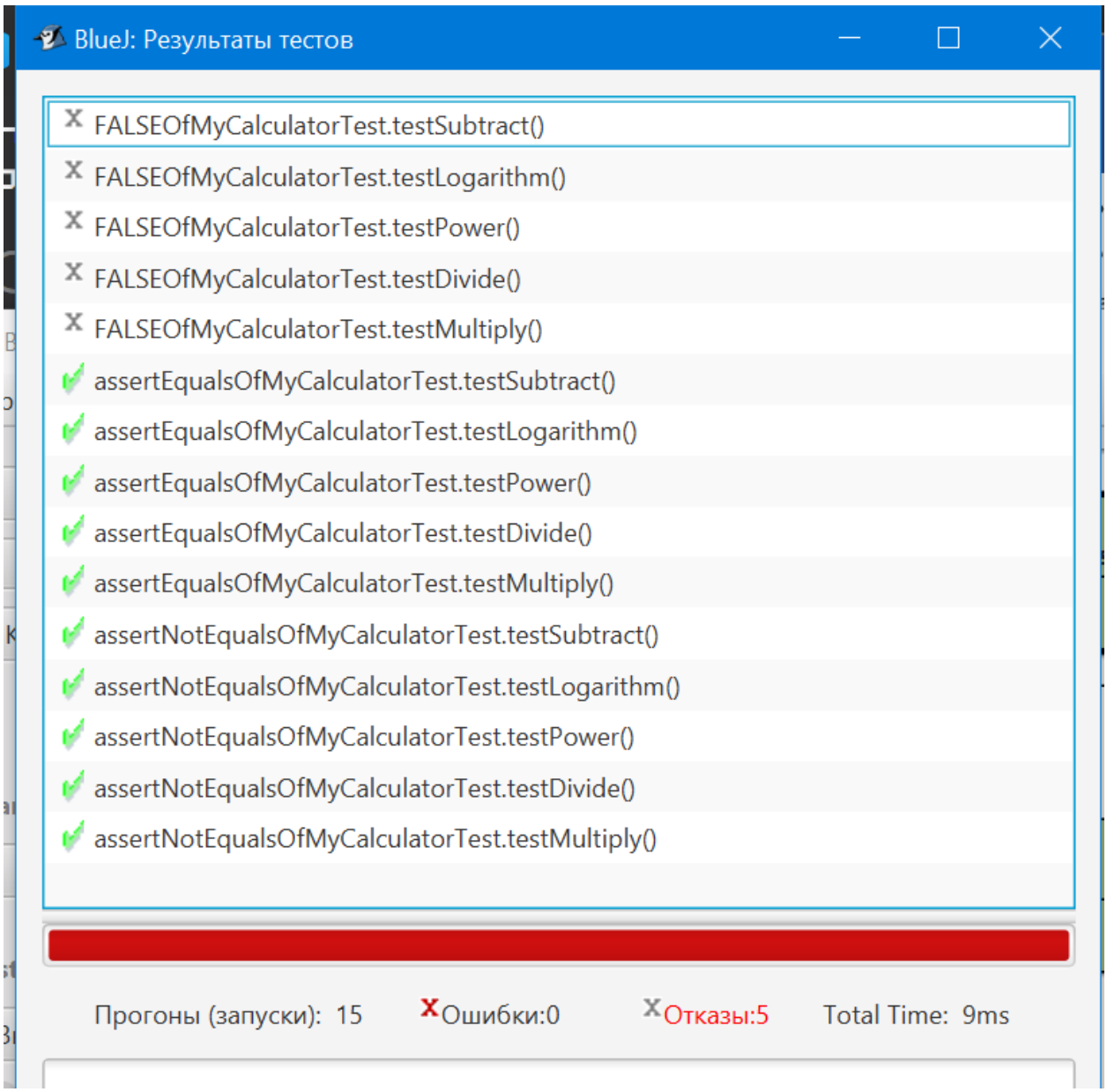
```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class assertNotEqualsOfMyCalculatorTest
{
   MyCalculator calculator = new MyCalculator();

   @Test
```

```
    public void testSubtract() {
        assertNotEquals(calculator.subtract(5, 3), 3.0, "Test subtraction with false value");
    }

    @Test
    public void testMultiply() {
        assertNotEquals(calculator.multiply(2, 3), 5.0, "Test multiplication with false value");
    }

    @Test
    public void testDivide() {
        assertNotEquals(calculator.divide(6, 3), 3.0, "Test division with false value");
    }

    @Test
    public void testPower() {
        assertNotEquals(calculator.power(2, 3), 9.0, "Test power with false value");
    }

    @Test
    public void testLogarithm() {
        assertNotEquals(calculator.logarithm(100, 10), 3.0, "Test logarithm withfalse value");
    }
}
```

модуль ***assertEqualsOfMyCalculatorTest***:

```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class assertEqualsOfMyCalculatorTest
{
    MyCalculator calculator = new MyCalculator();

    @Test
    public void testSubtract() {
        assertEquals(calculator.subtract(5, 3), 2.0, 0.0001, "Test subtraction with true value");
    }

    @Test
    public void testMultiply() {
        assertEquals(calculator.multiply(2, 3), 6.0, 0.0001, "Test multiplication with true value");
    }

    @Test
    public void testDivide() {
        assertEquals(calculator.divide(6, 3), 2.0, 0.0001, "Test division with true value");
    }

    @Test
    public void testPower() {
        assertEquals(calculator.power(2, 3), 8.0, 0.0001, "Test power with true value");
    }

    @Test
    public void testLogarithm() {
        assertEquals(calculator.logarithm(100, 10), 2.0, 0.0001, "Test logarithm withtrue value");
    }
}
```

Результат:


BlueJ: Результаты тестов
FALSEOfMyCalculatorTest.testSubtract()
FALSEOfMyCalculatorTest.testLogarithm()
FALSEOfMyCalculatorTest.testPower()
FALSEOfMyCalculatorTest.testDivide()
FALSEOfMyCalculatorTest.testMultiply()
assertEqualsOfMyCalculatorTest.testSubtract()
assertEqualsOfMyCalculatorTest.testLogarithm()
assertEqualsOfMyCalculatorTest.testPower()
assertEqualsOfMyCalculatorTest.testDivide()
assertEqualsOfMyCalculatorTest.testMultiply()
assertNotEqualsOfMyCalculatorTest.testSubtract()
assertNotEqualsOfMyCalculatorTest.testLogarithm()
assertNotEqualsOfMyCalculatorTest.testPower()
assertNotEqualsOfMyCalculatorTest.testDivide()
assertNotEqualsOfMyCalculatorTest.testMultiply()
Прогоны (запуски): 15
Ошибки:0
Отказы:5
Total Time: 9ms