**WEEK-2**

**Advanced JUnit Testing Exercises**

**Superset ID: 6419740**

**Exercise 1: Parameterized Tests**

**Code:**

**EvenChecker.java:**

public class EvenChecker {

public boolean isEven(int number) {

return number % 2 == 0;

}

}

**EvenCheckerTest.java:**

import org.junit.jupiter.params.ParameterizedTest;

import org.junit.jupiter.params.provider.ValueSource;

import static org.junit.jupiter.api.Assertions.\*;

public class EvenCheckerTest {

EvenChecker checker = new EvenChecker();

@ParameterizedTest

@ValueSource(ints = {2, 4, 6, 8, 10})

void testIsEven(int number) {

assertTrue(checker.isEven(number));

}

@ParameterizedTest

@ValueSource(ints = {1, 3, 5, 7, 9})

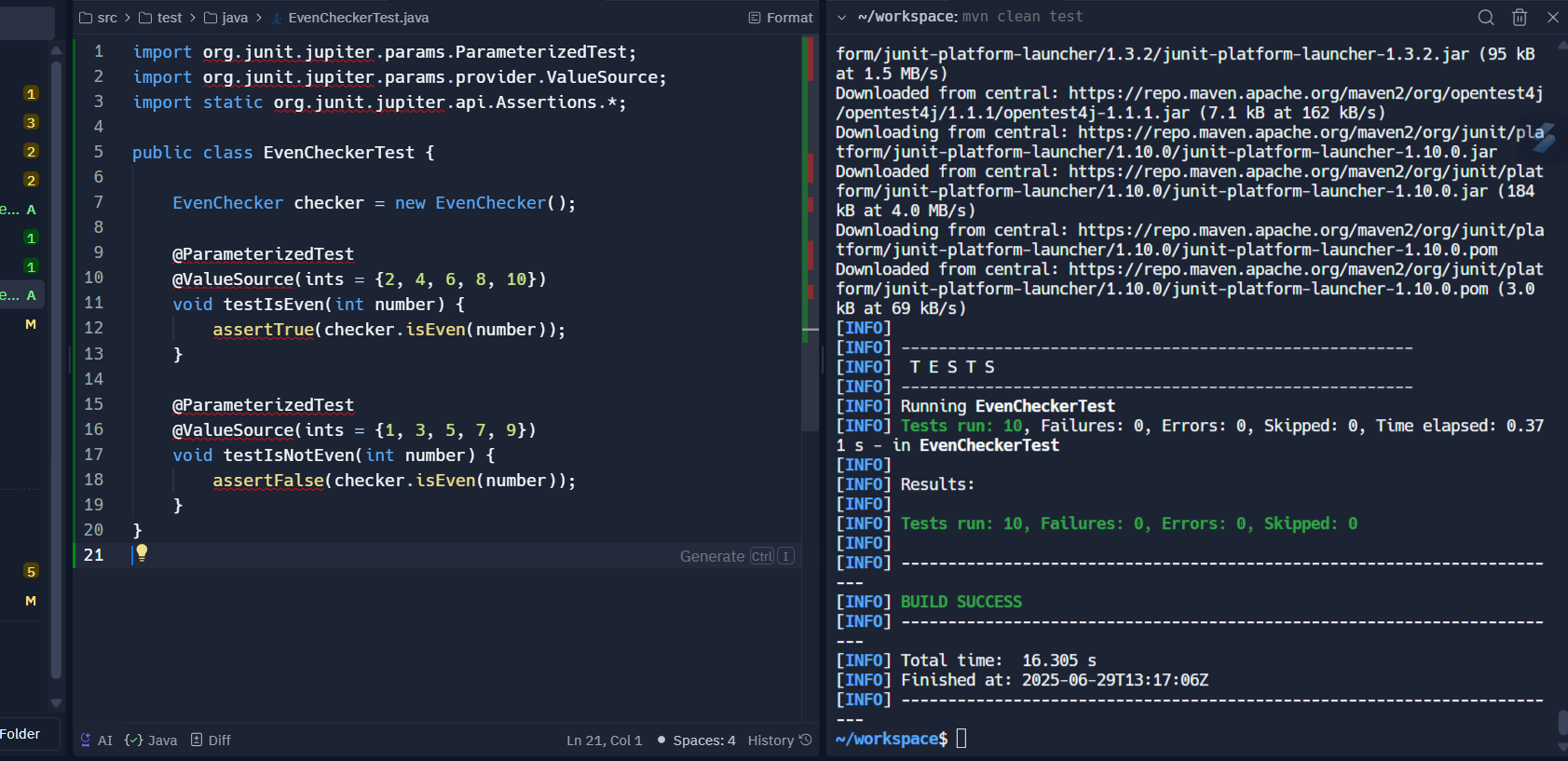
void testIsNotEven(int number) {

assertFalse(checker.isEven(number));

}

}

**Output:**



**Exercise 2: Test Suites and Categories**

**Code:**

**EvenChecker.java:**

public class EvenChecker {

public boolean isEven(int number) {

return number % 2 == 0;

}

}

**Calculator.java:**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**EvenCheckerTest.java:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class EvenCheckerTest {

@Test

void testEven() {

EvenChecker checker = new EvenChecker();

assertTrue(checker.isEven(4));

}

}

**CalculatorTest.java:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class CalculatorTest {

@Test

void testAdd() {

Calculator calc = new Calculator();

assertEquals(8, calc.add(3, 5));

}

}

**AllTest.java:**

import org.junit.platform.suite.api.SelectClasses;

import org.junit.platform.suite.api.Suite;

@Suite

@SelectClasses({

CalculatorTest.class,

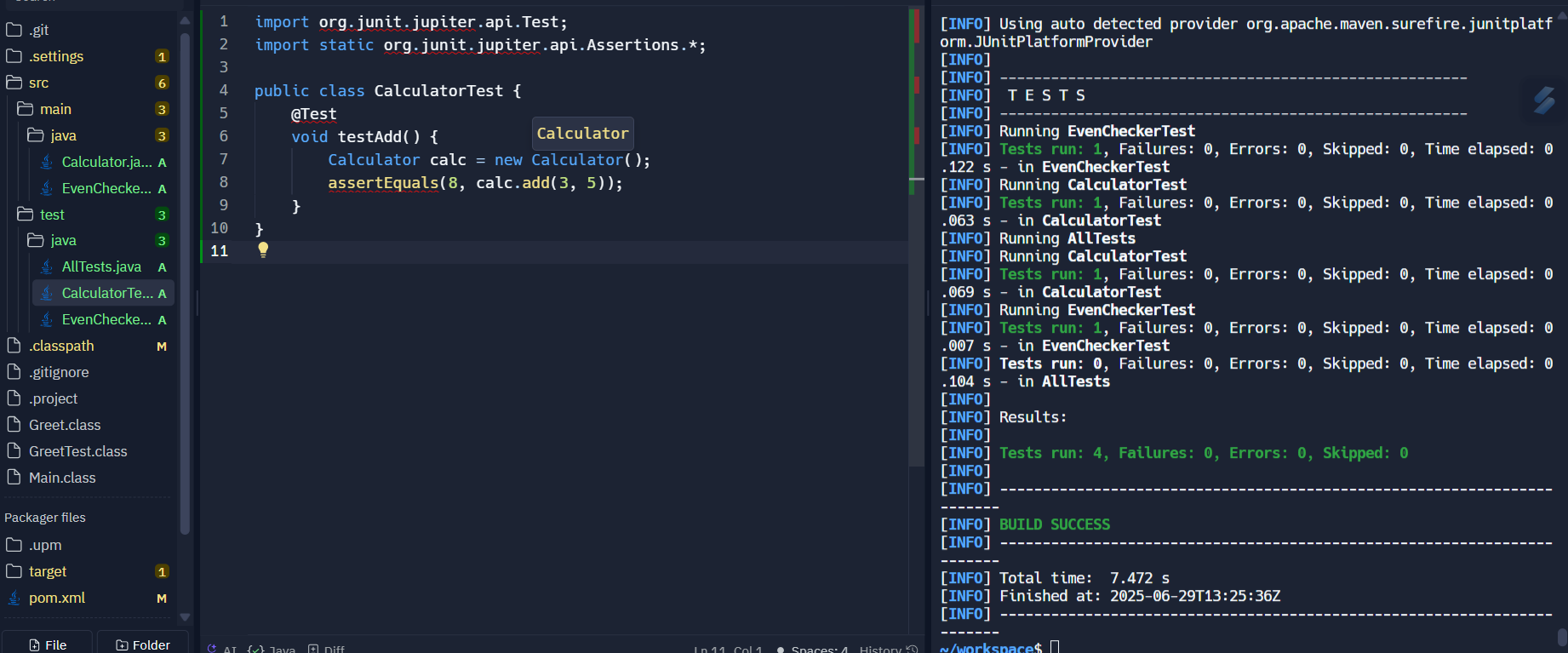
EvenCheckerTest.class

})

public class AllTests {

}

**Output:**

****

**Exercise 3: Test Execution Order**

**Code:**

**OrderedTests.java:**

import org.junit.jupiter.api.\*;

@TestMethodOrder(MethodOrderer.OrderAnnotation.class)

public class OrderedTests {

@Test

@Order(1)

public void firstTest() {

System.out.println("Running First Test");

Assertions.assertTrue(true);

}

@Test

@Order(2)

public void secondTest() {

System.out.println("Running Second Test");

Assertions.assertEquals(2, 1 + 1);

}

@Test

@Order(3)

public void thirdTest() {

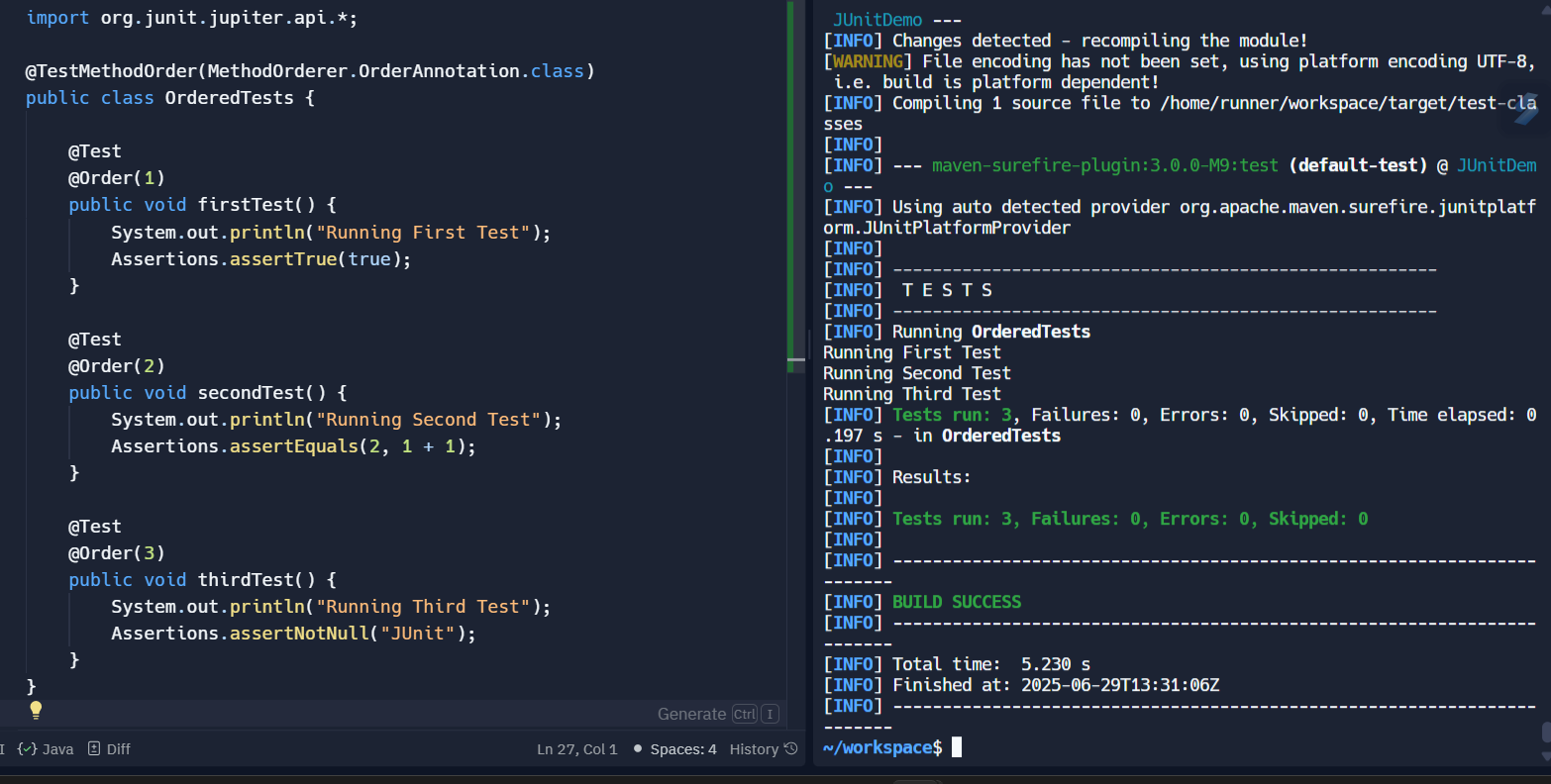
System.out.println("Running Third Test");

Assertions.assertNotNull("JUnit");

}

}

**Output:**

****

**Exercise 4: Exception Testing**

**Code:**

**ExceptionThrower.java:**

public class ExceptionThrower {

public void throwException(boolean shouldThrow) {

if (shouldThrow) {

throw new IllegalArgumentException("Invalid argument passed!");

}

}

}

**ExceptionThrowerTest.java:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class ExceptionThrowerTest {

@Test

public void testThrowsException() {

ExceptionThrower et = new ExceptionThrower();

assertThrows(IllegalArgumentException.class, () -> et.throwException(true));

}

@Test

public void testDoesNotThrowException() {

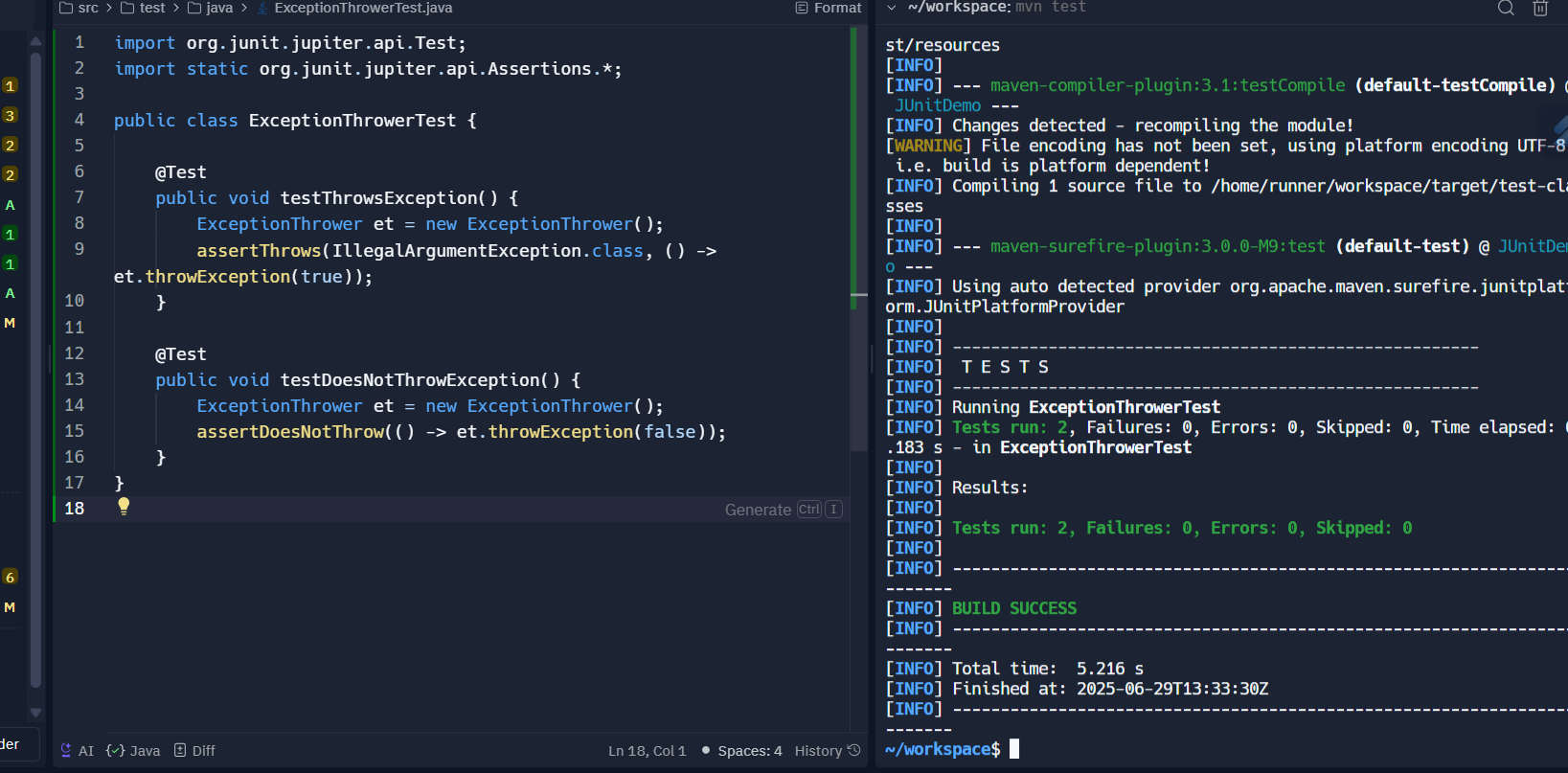
ExceptionThrower et = new ExceptionThrower();

assertDoesNotThrow(() -> et.throwException(false));

}

}

**Output:**

****

**Exercise 5: Timeout and Performance Testing**

**Code:**

**PerformanceTester.java:**

public class PerformanceTester {

public void performTask() {

for (int i = 0; i < 1000000; i++) {

Math.sqrt(i);

}

}

}

**PerformanceTesterTest.java:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

import java.time.Duration;

public class PerformanceTesterTest {

@Test

public void testPerformTaskCompletesQuickly() {

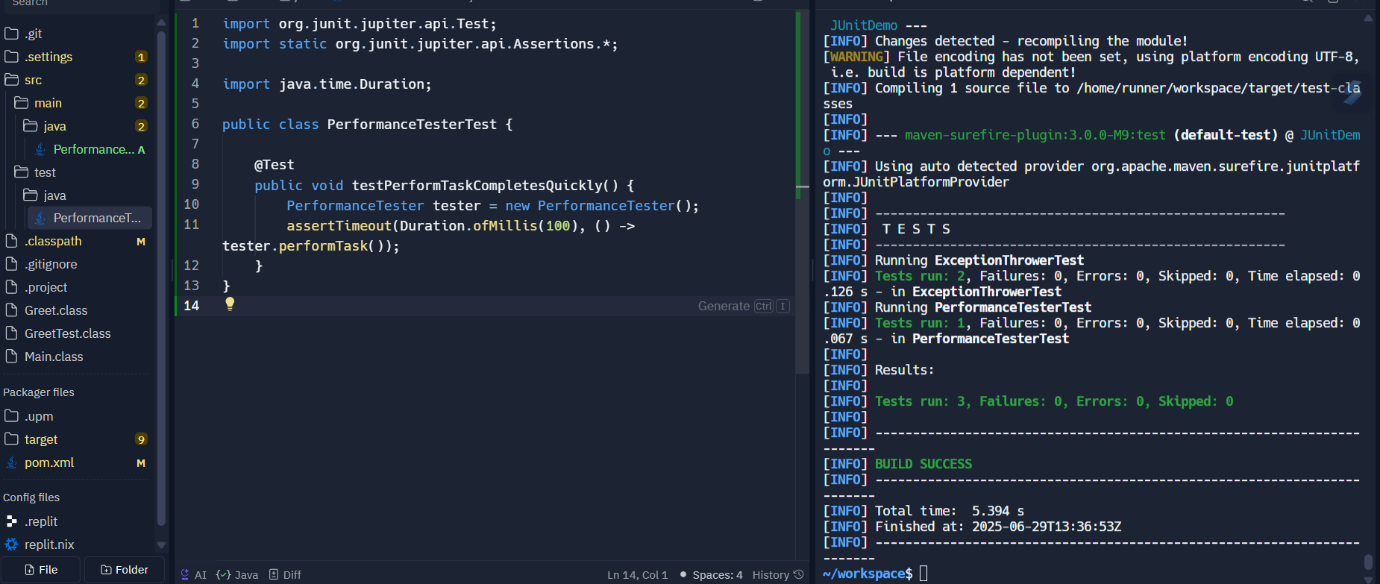
PerformanceTester tester = new PerformanceTester();

assertTimeout(Duration.ofMillis(100), () -> tester.performTask());

}

}

**Output:**

****