**WEEK - 2**

**Junit Advanced JUnit Testing Exercises**

**Exercise 1: Parameterized Tests**

**CODE:**

**EvenChecker.java:**

public class EvenChecker {

public static 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 {

@ParameterizedTest

@ValueSource(ints = {2, 4, 6, 8, 10, 0, -2})

public void testIsEven\_withEvenNumbers(int number) {

assertTrue(EvenChecker.isEven(number), number + " should be even");

}

@ParameterizedTest

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

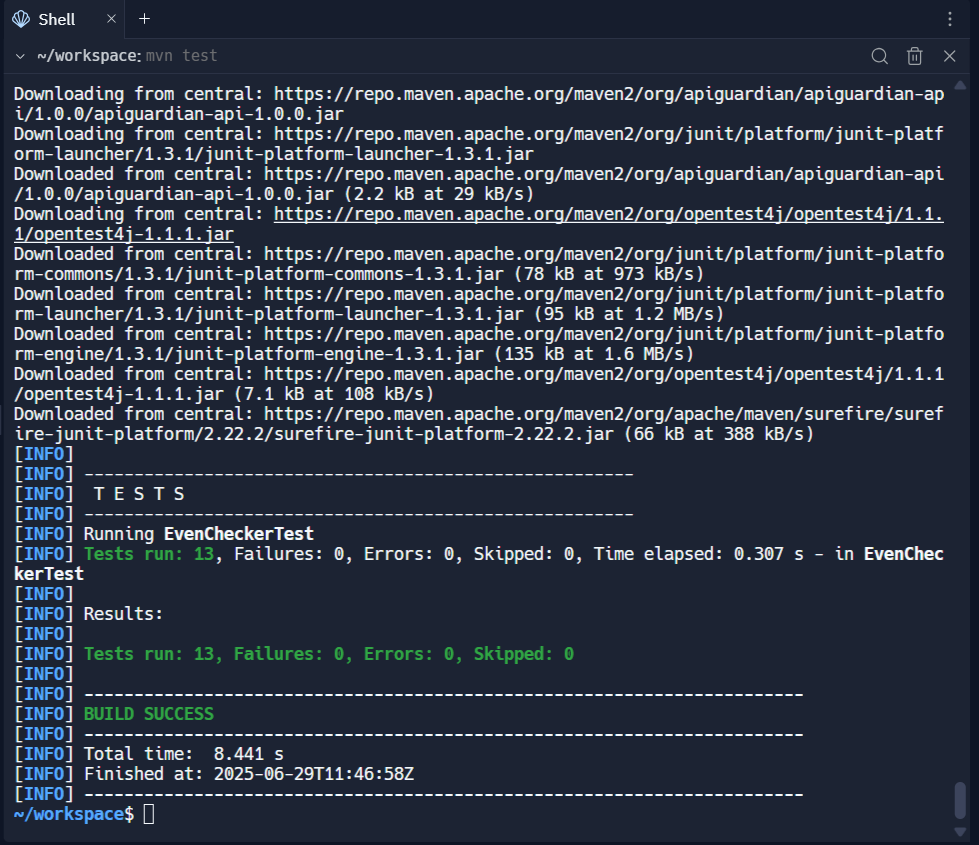
public void testIsEven\_withOddNumbers(int number) {

assertFalse(EvenChecker.isEven(number), number + " should be odd");

}

}

**Output:**

****

**Exercise 2: Test Suites and Categories**

**EventChecker.java:**

public class EvenChecker {

public static boolean isEven(int number) {

return number % 2 == 0;

}

}

**AllTests.java:**

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

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

@Suite

@SelectClasses({

EvenCheckerTest.class,

CalculatorFixtureTest.class

})

public class AllTests {

}

**EvenChecker.java:**

public class EvenChecker {

public static boolean isEven(int number) {

return number % 2 == 0;

}

}

**CalculatorFixtureTest.java:**

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.Test;

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

class Calculator {

public int add(int a, int b) {

return a + b;

}

}

public class CalculatorFixtureTest {

private Calculator calculator;

@BeforeEach

void setUp() {

calculator = new Calculator();

}

@AfterEach

void tearDown() {

calculator = null;

}

@Test

void testAddition() {

assertEquals(7, calculator.add(3, 4));

}

@Test

void testAdditionWithZero() {

assertEquals(5, calculator.add(5, 0));

}

}

**OUTPUT:**

A screenshot of a computer program

AI-generated content may be incorrect.

**Exercise 3: Test Execution Order Scenario:**

**CODE:**

**OrderedTests.java:**

import org.junit.jupiter.api.Order;

import org.junit.jupiter.api.Test;

import org.junit.jupiter.api.TestMethodOrder;

import org.junit.jupiter.api.MethodOrderer.OrderAnnotation;

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

@TestMethodOrder(OrderAnnotation.class)

public class OrderedTests {

@Test

@Order(1)

void testInitialize() {

System.out.println("Running testInitialize");

assertTrue(true);

}

@Test

@Order(2)

void testProcess() {

System.out.println("Running testProcess");

assertEquals(5, 2 + 3);

}

@Test

@Order(3)

void testCleanup() {

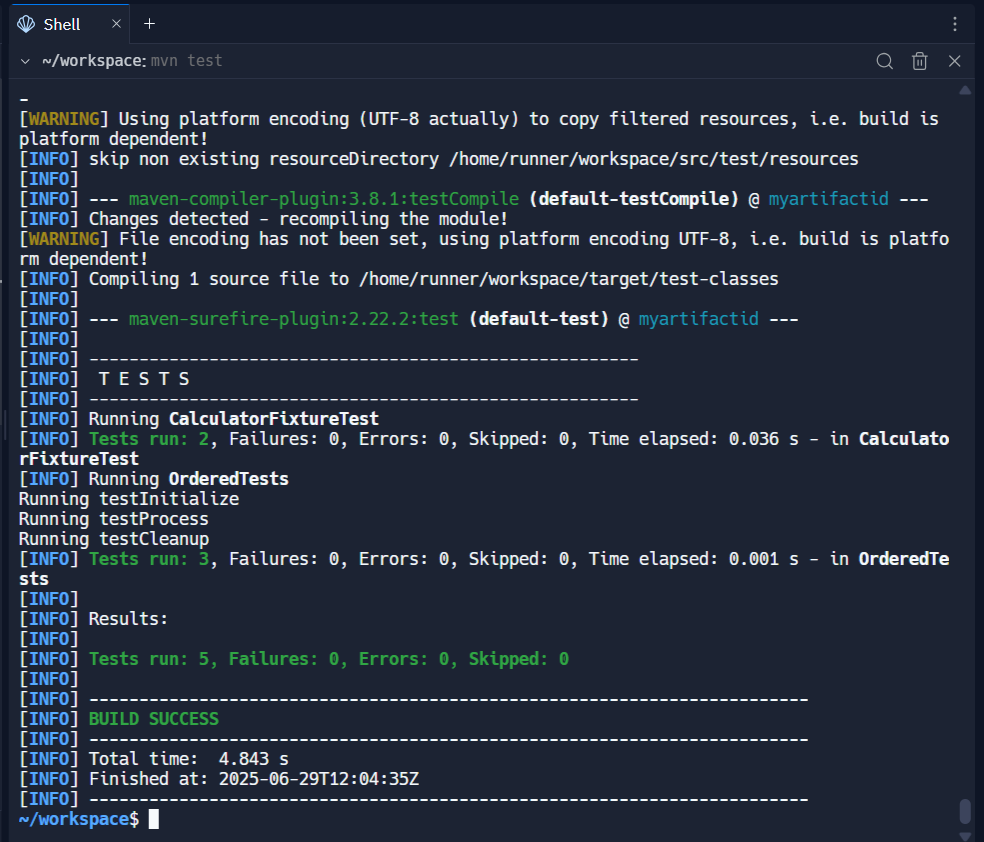
System.out.println("Running testCleanup");

assertNotNull("Done");

}

}

**OUTPUT:**



**Exercise 4: Exception Testing**

**CODE:**

**ExceptionThrower.java:**

public class ExceptionThrower {

public void throwException() {

throw new IllegalArgumentException("Invalid argument passed");

}

}

**ExceptionThrowerTest.java:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertThrows;

public class ExceptionThrowerTest {

@Test

void testExceptionIsThrown() {

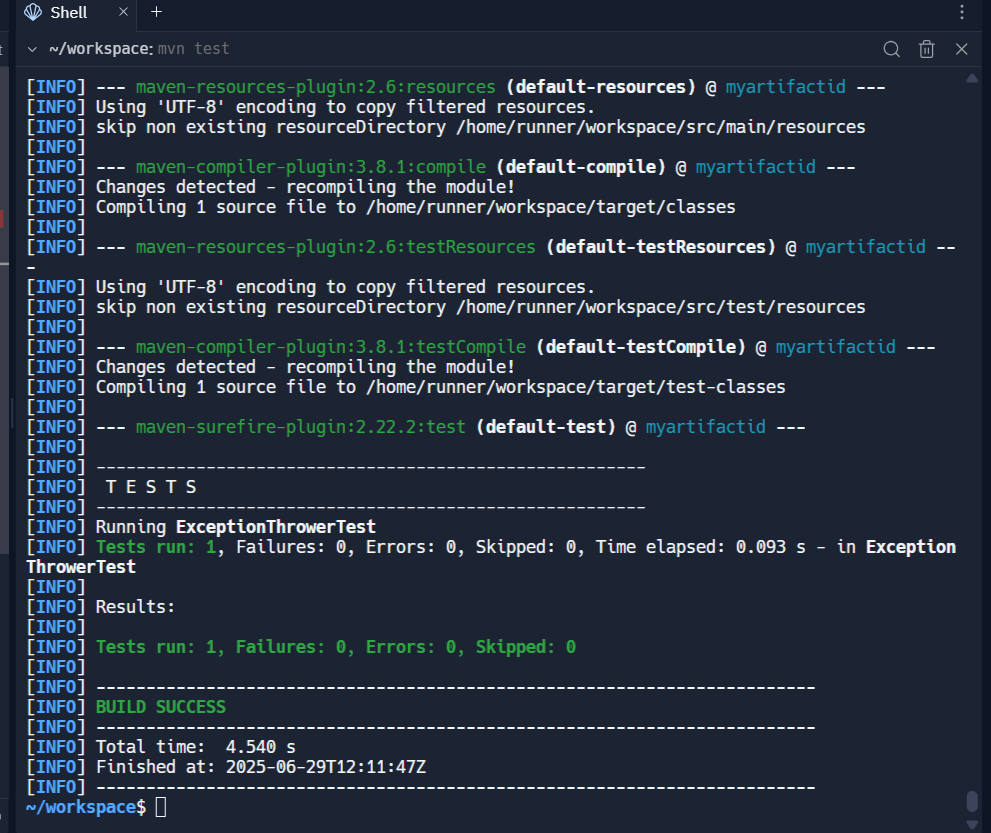
ExceptionThrower thrower = new ExceptionThrower();

assertThrows(IllegalArgumentException.class, thrower::throwException);

}

}

**OUTPUT:**



**Exercise 5: Timeout and Performance Testing**

**CODE:**

**PerformanceTester.java:**

public class PerformanceTester {

public void performTask() {

try {

Thread.sleep(500);

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

}

}

**PerformanceTesterTest.java:**

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertTimeout;

import java.time.Duration;

public class PerformanceTesterTest {

@Test

void testPerformTaskCompletesWithinOneSecond() {

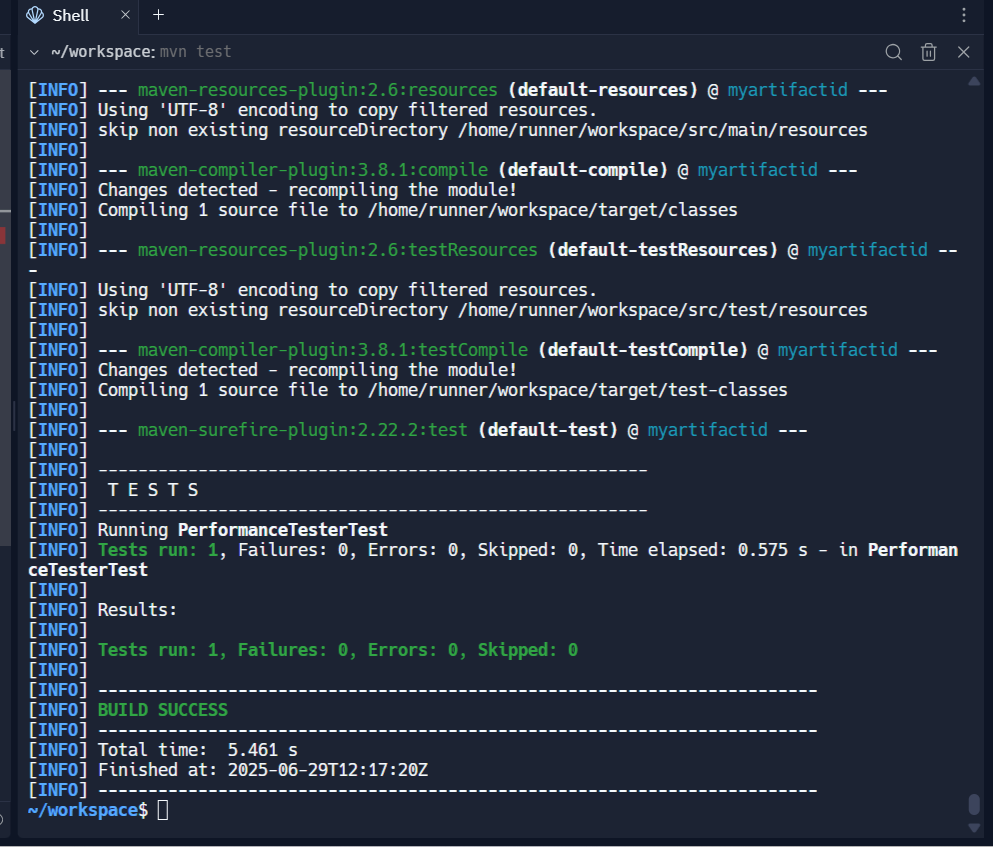
PerformanceTester tester = new PerformanceTester();

assertTimeout(Duration.ofSeconds(1), tester::performTask);

}

}

**OUTPUT:**

****