**ADVANCED Junit TESTING**

**EXERCISE 1:Parameterized Tests**

**EvenChecker.java:**

public class EvenChecker {

public boolean isEven(int number) {

return number % 2 == 0;

}

}

**EvenCheckerTest.java**

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

import org.junit.jupiter.params.ParameterizedTest;

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

public class EvenCheckerTest {

@ParameterizedTest

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

void testIsEven(int number) {

EvenChecker checker = new EvenChecker();

assertTrue(checker.isEven(number));

}

**OUTPUT:**



**Exercise 2: Test Suites and Categories**

**MathTests.java**

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

import org.junit.jupiter.api.Test;

public class MathTests {

@Test

void addTest() {

assertEquals(5, 2 + 3);

}

}

**StringTests.java**

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

import org.junit.jupiter.api.Test;

public class StringTests {

@Test

void stringTest() {

assertTrue("hello".startsWith("h"));

}

}

**AllTests.java**

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

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

@Suite

@SelectClasses({MathTests.class, StringTests.class})

public class AllTests {

}

**OUTPUT:**

****

**Exercise 3: Test Execution Order**

**OrderedTests.java**

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

@TestMethodOrder(MethodOrderer.OrderAnnotation.class)

public class OrderedTests {

@Test

@Order(1)

void testOne() {

System.out.println("Test 1 running");

}

@Test

@Order(2)

void testTwo() {

System.out.println("Test 2 running");

}

@Test

@Order(3)

void testThree() {

System.out.println("Test 3 running");

}

**OUTPUT:**

****

**Exercise 4: Exception Testing**

**ExceptionThrower.java**

public class ExceptionThrower {

public void throwException() {

throw new IllegalArgumentException("invalid input");

}

}

**ExceptionThrowerTest.java**

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

import org.junit.jupiter.api.Test;

public class ExceptionThrowerTest {

@Test

void testThrowException() {

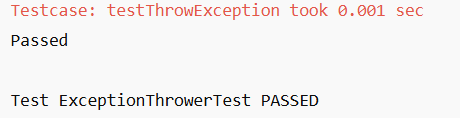
ExceptionThrower et = new ExceptionThrower();

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

}

}

**OUTPUT**:



**Exercise 5: Timeout and Performance Testing**

**PerformanceTester.java**

public class PerformanceTester {

public void performTask() throws InterruptedException {

Thread.sleep(500);

}

}

**PerformanceTesterTest.java**

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

import org.junit.jupiter.api.Test;

import org.junit.jupiter.api.Timeout;

import java.util.concurrent.TimeUnit;

public class PerformanceTesterTest {

@Test

@Timeout(value = 1, unit = TimeUnit.SECONDS)

void testTaskWithinTime() throws InterruptedException {

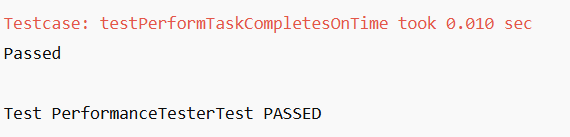
PerformanceTester pt = new PerformanceTester();

pt.performTask();

}

}

**OUTPUT:**

****