**Exercise 1: Setting Up Junit**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

#test class

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

@Test

public void testAdd() {

Calculator calc = new Calculator();

int result = calc.add(2, 3);

assertEquals(5, result);

}

}

**Exercise 3: Assertions in Junit**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

assertEquals("Sum should be 5", 5, 2 + 3);

assertTrue("5 is greater than 3", 5 > 3);

assertFalse("5 is not less than 3",5<3);

assertNull("Expected null value", null);

assertNotNull("Expected non-null object", new Object());

}

}

**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

#setup and teardown methods

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calculator;

@Before

public void setUp() {

calculator = new Calculator();

System.out.println("Setting up Calculator instance...");

}

@After

public void tearDown() {

calculator = null;

System.out.println("Tearing down Calculator instance...");

}

@Test

public void testAddition() {

int a = 5;

int b = 3;

int result = calculator.add(a, b);

assertEquals("5 + 3 should equal 8", 8, result);

}

@Test

public void testSubtraction() {

int a = 10;

int b = 4;

int result = calculator.subtract(a, b);

assertEquals("10 - 4 should equal 6", 6, result);

}

}

**Mockito Exercises**

**Exercise 1: Mocking and Stubbing**

🔹 External API Interface

public interface ExternalApi {

String getData();

}

🔹 Service Class that Uses the API

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

🔹 JUnit 5 Test Class with Mockito

import org.junit.jupiter.api.Test;

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

import static org.mockito.Mockito.\*;

public class MyServiceTest {

@Test

public void testExternalApi() {

// Step 1: Create mock

ExternalApi mockApi = mock(ExternalApi.class);

// Step 2: Stub method

when(mockApi.getData()).thenReturn("Mock Data");

// Step 3: Inject mock into service

MyService service = new MyService(mockApi);

// Act and Assert

String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

**Exercise 2: Verifying Interactions**

🔹 External API Interface

public interface ExternalApi {

String getData(); }

🔹 Service Class

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData(); // This method should be verified

}

}

JUnit 5 Test: Verifying Interaction with Mockito

import org.junit.jupiter.api.Test;

import static org.mockito.Mockito.\*;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Step 1: Create mock

ExternalApi mockApi = mock(ExternalApi.class);

// Step 2: Inject into service and call method

MyService service = new MyService(mockApi);

service.fetchData();

// Step 3: Verify that getData() was called exactly once

verify(mockApi, times(1)).getData(); // or simply: verify(mockApi).getData();

}

}

**SLJ4 LOGGING EXERCISES**

**Exercise 1: Logging Error Messages and Warning Levels Task: Write a Java application that demonstrates logging error messages and warning levels using SLF4J**

Add Dependencies to pom.xml

<dependencies>

<!-- SLF4J API -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.30</version>

</dependency>

<!-- Logback Classic (SLF4J Implementation) -->

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.2.3</version>

</dependency>

</dependencies>

Create the Logging Java Class

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class LoggingExample {

private static final Logger logger = LoggerFactory.getLogger(LoggingExample.class);

public static void main(String[] args) {

logger.error("❌ This is an error message");

logger.warn("⚠️ This is a warning message");

logger.info("ℹ️ This is an info message");

logger.debug("🐞 This is a debug message");

}

}