JUnit Testing Exercises

**Exercise 1: Setting Up Junit**

**Step 1: Create a New Java Project in Your IDE**

For IntelliJ IDEA:

1. Open IntelliJ IDEA.
2. Click on New Project.
3. Select Maven on the left sidebar.
4. Check Create from archetype if needed (optional).
5. Click Next.
6. Fill in:
   * GroupId: com.example
   * ArtifactId: junit-demo
7. Click Next, then Finish.
8. Your project will open with a basic pom.xml file.

**Step 2: Add JUnit Dependency to pom.xml**

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

**Step 3: Create a Java Class to Test**

1. Right-click on src/main/java.
2. Go to New > Java Class.
3. Name it Calculator.

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**Step 4: Create a JUnit Test Class**

1. Right-click on src/test/java.
2. Go to New > Java Class.
3. Name it CalculatorTest.

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

@Test

public void testAdd() {

Calculator calc = new Calculator();

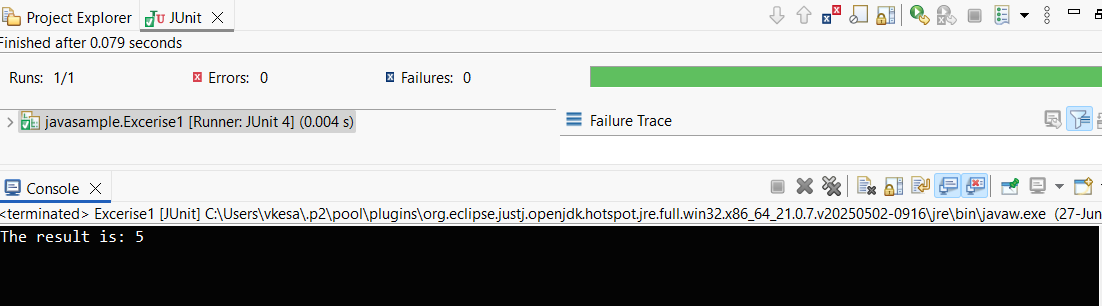
int result = calc.add(2, 3);

assertEquals(5, result);

}

}

**OUTPUT :**

****

**Exercise 3: Assertions in Junit**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

// Assert that 2 + 3 equals 5

assertEquals(5, 2 + 3);

// Assert that condition is true

assertTrue(5 > 3);

// Assert that condition is false

assertFalse(5 < 3);

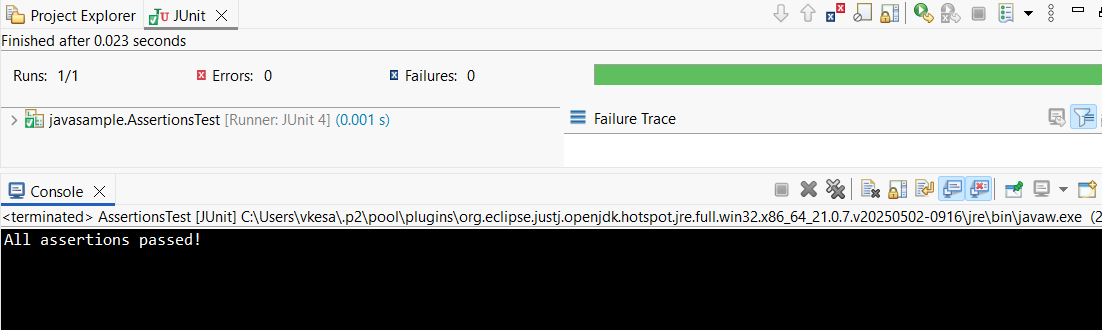
assertNull(null);

assertNotNull(new Object());

}

}

**OUTPUT :**

****

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

**Calculator.java**

package ArrangeActAssert;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

**CalculatorTest.java**

import org.junit.Before;

import org.junit.After;

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 test…");

}

// Teardown method – runs after each test

@After

public void tearDown() {

calculator = null;

System.out.println("Cleaning up after test…");

}

@Test

public void testAddition() {

int result = calculator.add(10, 20);

assertEquals(30, result);

}

@Test

public void testSubtraction() {

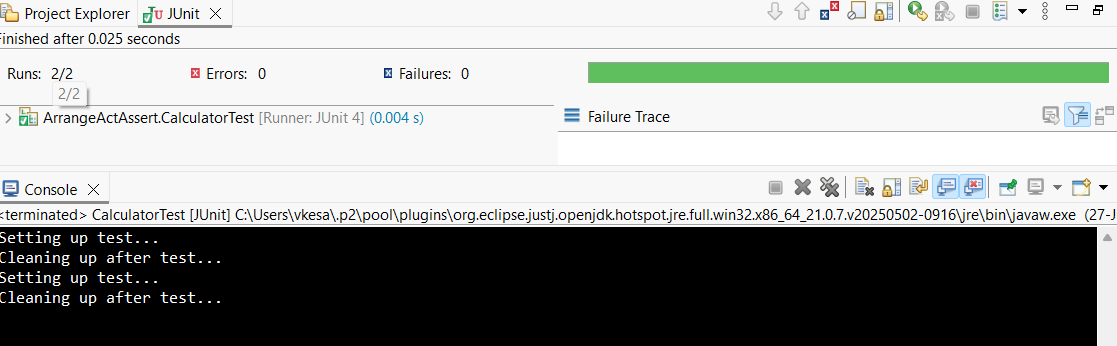
int result = calculator.subtract(50, 20);

assertEquals(30, result);

}

}

**OUTPUT :**

****

**MOCKITO EXERCISES**

**Exercise 1: Mocking and Stubbing**

**ExternalApi.java**

package com.example.demo;

public interface ExternalApi {

String getData();

}

**MyService.java**

package com.example.demo

public class MyService {

private ExternalApi externalApi;

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

public String fetchData() {

return externalApi.getData();

}

}

**MyServiceTest.java**

package com.example.demo;

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

import static org.mockito.Mockito.\*

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testExternalApi() {

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

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

MyService service = new MyService(mockApi);

String result = service.fetchData();

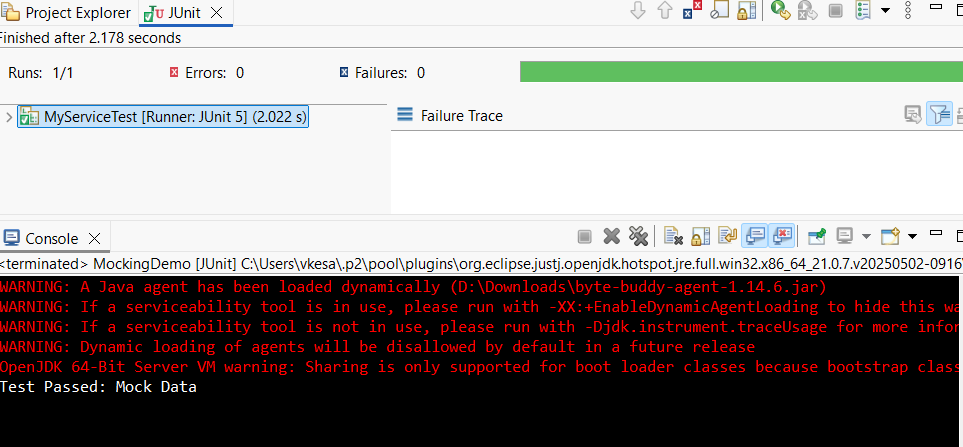
assertEquals("Mock Data", result);

System.out.println("Test Passed: " + result);

}

}

**OUTPUT :**



**Exercise 2: Verifying Interactions**

**ExternalApi.java**

package VERIFYINTERACTION;

public interface ExternalApi {

String getData();

}

**MyService.java**

package VERIFYINTERACTION;

public class MyService {

private ExternalApi externalApi;

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

public String fetchData() {

return externalApi.getData();

} }

**MyServiceTest.java**

package VERIFYINTERACTION;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Step 1: Create a mock object

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

// Step 2: Call the method using MyService

MyService service = new MyService(mockApi);

service.fetchData();

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

verify(mockApi).getData();

System.out.println("Verified: getData() was called on mockApi");

}

}

**OUTPUT :**

