1. JUnit\_Basic Testing Exercises

Exercise 1: Setting Up JUnit

Calculator.java:

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

CalculatorTest.java:

import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class CalculatorTest {

public void testAddition() {

Calculator calc = new Calculator();

int result = calc.add(5, 3);

assertEquals(8, result);

}

}

Output:

Test passed.

Exercise 3: Assertions in Junit

AssertionsTest.java:

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

// Assert equals

assertEquals(5, 2 + 3);

// Assert true

assertTrue(5 > 3);

// Assert false

assertFalse(5 < 3);

// Assert null

assertNull(null);

// Assert not null

assertNotNull(new Object());

}

}

Output:

Test passed.

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

CalculatorTest.java:

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class CalculatorTest {

private Calculator calculator;

@Before

public void setUp() {

calculator = new Calculator();

System.out.println("Setup: Calculator object created.");

}

@After

public void tearDown() {

calculator = null;

System.out.println("Teardown: Calculator object cleaned up.");

}

@Test

public void testAddition() {

int a = 10;

int b = 5;

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

assertEquals(15, result);

}

@Test

public void testSubtraction() {

int a = 10;

int b = 4;

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

assertEquals(6, result);

}

}

Calculator.java:

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

Output:

Setup: Calculator object created.

Teardown: Calculator object cleaned up.

Setup: Calculator object created.

Teardown: Calculator object cleaned up.

3. Mockito exercises

Exercise 1: Mocking and Stubbing

import static org.mockito.Mockito.\*;

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

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

interface ExternalApi {

String getData();

}

class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

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);

}

}

Output:

Test passed: testExternalApi

Exercise 2: Verifying Interactions

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

interface ExternalApi {

String getData();

}

class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

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

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

}

}

Output :

Test passed: testVerifyInteraction