**JUnit Testing Exercises**

**Exercise 1: Setting Up JUnit Scenario: You need to set up JUnit in your Java project to start writing unit tests.**

pom.xml:

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

**CalculatorTest.java**

package com.example.junit\_demo;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

@Test

public void testAdd() {

Calculator calc = new Calculator();

assertEquals(5, calc.add(2, 3));

}

}

**Calculator.java**

package com.example.junit\_demo;

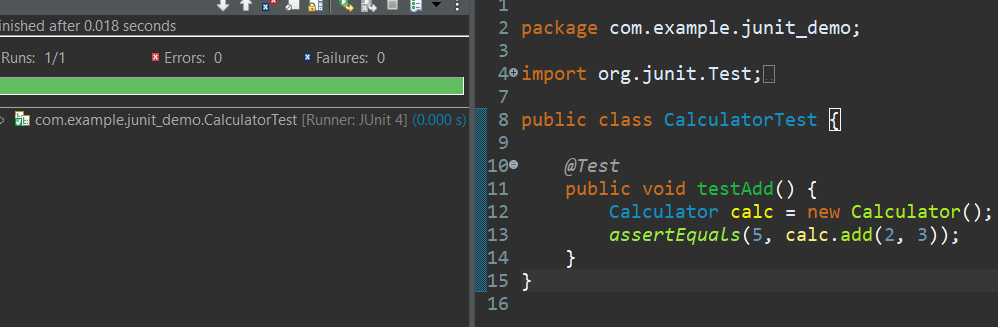
public class Calculator {

public int add(int a, int b) {

return a + b;

}

}



**Exercise 2: Writing Basic JUnit Tests Scenario: You need to write basic JUnit tests for a simple Java class.**

**SOLUTION**

**1.Base.java**

package Utilities;

public class Base {

public static boolean isEven(int number) {

return number % 2 == 0;

}

}

**2.BaseTest.java**

package Utilities;

import org.junit.jupiter.api.Test;

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

public class BaseTest {

@Test

void testIsEven1() {

System.out.println("Checking Test Case 1");

assertFalse(Base.isEven(19));

}

@Test

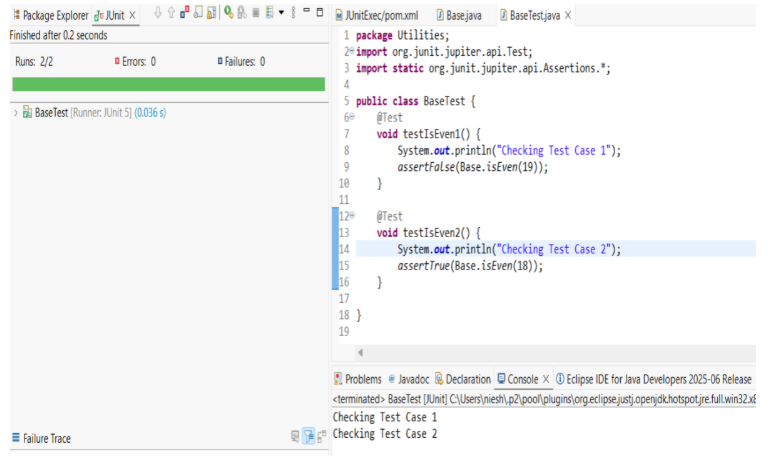
void testIsEven2() {

System.out.println("Checking Test Case 2");

assertTrue(Base.isEven(18));

}

}



**Exercise 3: Assertions in JUnit Scenario: You need to use different assertions in JUnit to validate your test results.**

**SOLUTION**

**1.AssertionsTest.java**

package Utilities;

import org.junit.jupiter.api.Test;

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

public class AssertionsTest {

@Test

public void testAssertions() {

System.out.println("Running assertion test...");

// Assert equals

assertEquals(5, 2 + 3, "2 + 3 should equal 5");

// Assert true

assertTrue(5 > 3, "5 should be greater than 3");

//assert false

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

//assert null

assertNull(null, "Should be null");

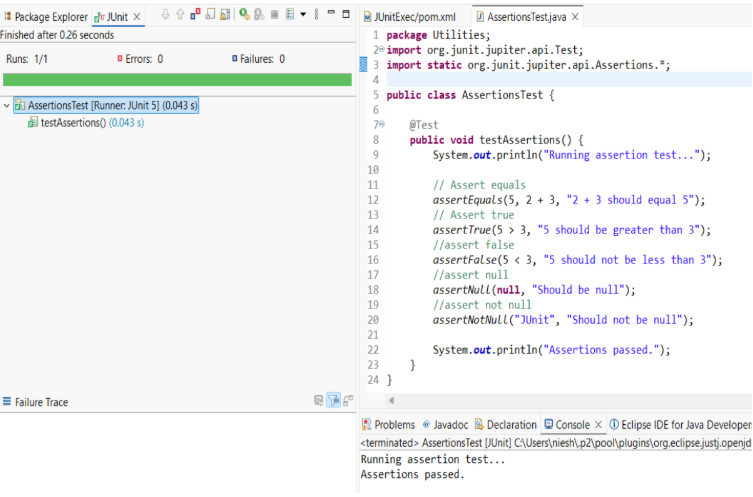
//assert not null

assertNotNull("JUnit", "Should not be null");

System.out.println("Assertions passed.");

}

}

****

**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit Scenario: You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup and teardown methods.**

**SOLUTION**

**1.BankAccount.java**

package Utilities;

public class BankAccount {

private int balance;

public BankAccount(int initialBalance) {

this.balance = initialBalance;

}

public void deposit(int amount) {

balance += amount;

}

public boolean withdraw(int amount) {

if (balance >= amount) {

balance -= amount;

return true;

} else {

return false;

}

}

public int getBalance() {

return balance;

}

}

**2.BankAccountTest.java**

package Utilities;

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.\*;

public class BankAccountTest {

private BankAccount account;

@BeforeEach

void setUp() {

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

account = new BankAccount(100); // Arrange: start with ₹100

}

@AfterEach

void tearDown() {

System.out.println("Cleaning up...");

account = null;

}

@Test

void testDeposit() {

// Act

account.deposit(50);

// Assert

assertEquals(150, account.getBalance(), "Balance should be 150 after deposit");

System.out.println("Running Deposit Method");

}

@Test

void testWithdrawSuccess() {

// Act

boolean result = account.withdraw(60);

// Assert

assertTrue(result, "Withdrawal should be successful");

assertEquals(40, account.getBalance(), "Balance should be 40 after withdrawal");

System.out.println("Running WithDrawSuccess Method");

}

@Test

void testWithdrawFail() {

// Act

boolean result = account.withdraw(200);

// Assert

assertFalse(result, "Withdrawal should fail");

assertEquals(100, account.getBalance(), "Balance should remain 100");

System.out.println("Running Deposit Method");

System.out.println("Running WithDrawFail Method");

}

}

