**JUnit Testing Exercises**

**Exercise 1: Setting Up JUnit**

**Scenario:**

You need to set up JUnit in your Java project to start writing unit tests.

Steps:

1. **Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse).**

**Code:**

**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;

}

public int multiply(int a, int b) {

return a \* b;

}

public int divide(int a, int b) {

if (b == 0) throw new ArithmeticException("Cannot divide by zero");

return a / b;

}

}

**2. Add JUnit dependency to your project. If you are using Maven, add the following to your**

pom.xml:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.sohana</groupId>

<artifactId>junit-java-test</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

</properties>

<dependencies>

<!-- JUnit 4.13.2 -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

<!-- Required for assertThat, etc. -->

<dependency>

<groupId>org.hamcrest</groupId>

<artifactId>hamcrest-core</artifactId>

<version>1.3</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Enables running test classes using mvn test -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.2</version>

</plugin>

</plugins>

</build>

</project>

1. **Create a new test class in your project**

**CalculatorTest.java:**

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

Calculator calc = new Calculator();

@Test

public void testAdd() {

assertEquals(15, calc.add(10, 5));

}

@Test

public void testSubtract() {

assertEquals(5, calc.subtract(10, 5));

}

@Test

public void testMultiply() {

assertEquals(50, calc.multiply(10, 5));

}

@Test

public void testDivide() {

assertEquals(2, calc.divide(10, 5));

}

@Test(expected = ArithmeticException.class)

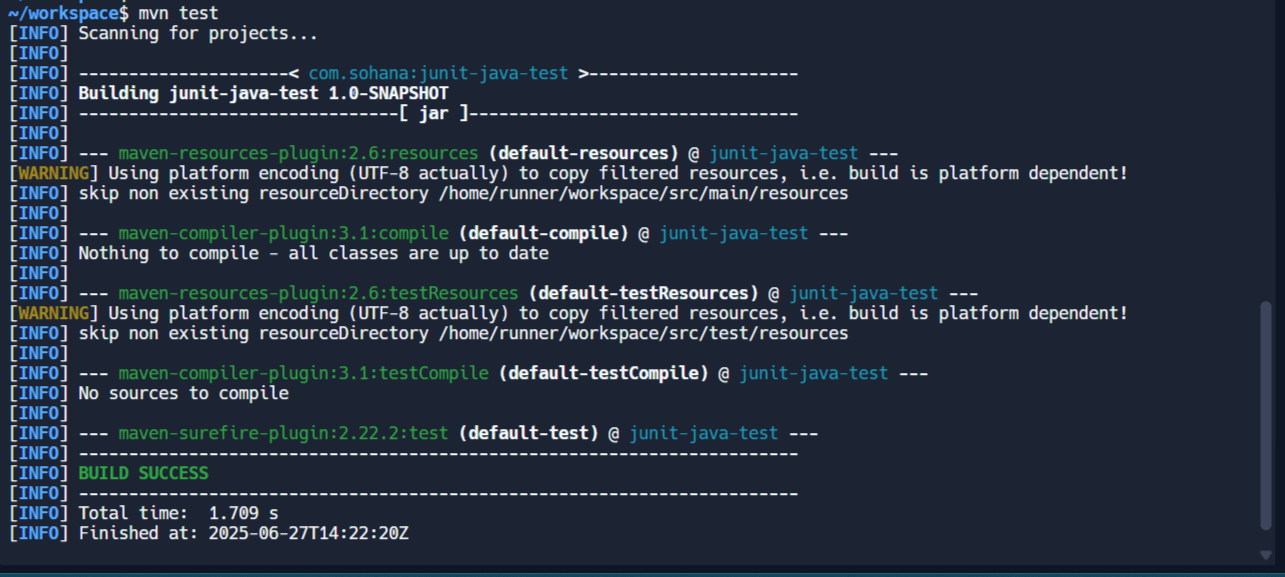
public void testDivideByZero() {

calc.divide(10, 0);

}

}

**Output:**

****

**Exercise 3: Assertions in JUnit**

**Scenario:**

**You need to use different assertions in JUnit to validate your test results.**

**Steps:**

**1. Write tests using various JUnit assertions.**

Solution Code:

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());

}

}

**AssertionsTest.java:**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

System.out.println("Running test: assertEquals(5, 2 + 3)");

assertEquals(5, 2 + 3);

System.out.println("✔ Passed");

System.out.println("Running test: assertTrue(5 > 3)");

assertTrue(5 > 3);

System.out.println("✔ Passed");

System.out.println("Running test: assertFalse(5 < 3)");

assertFalse(5 < 3);

System.out.println("✔ Passed");

System.out.println("Running test: assertNull(null)");

assertNull(null);

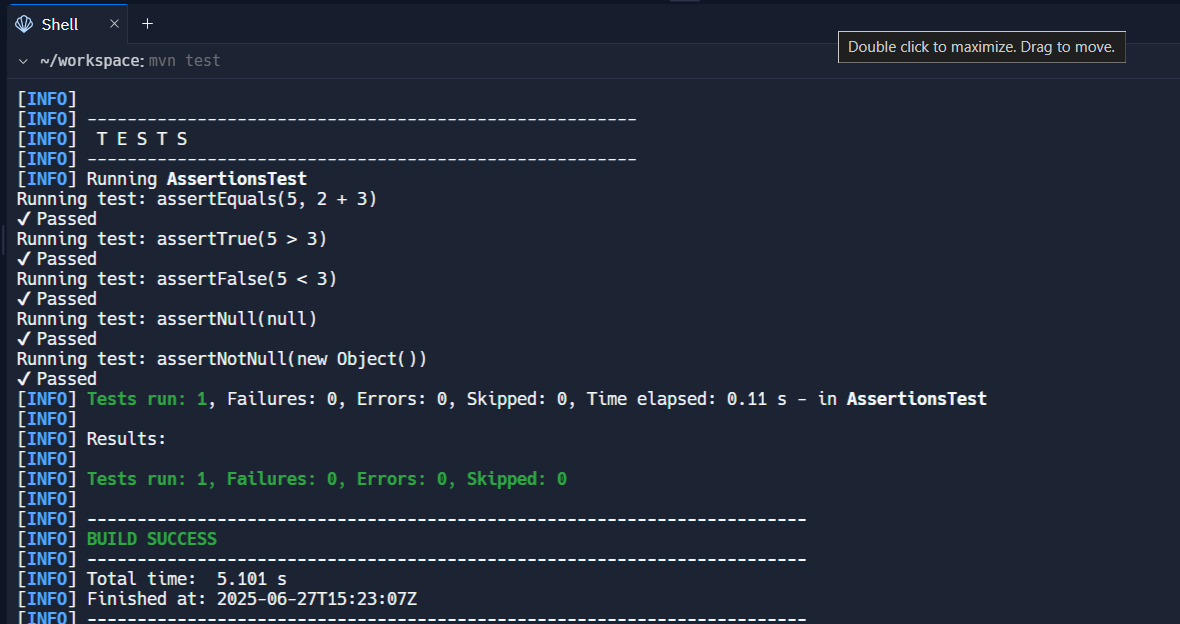
System.out.println("✔ Passed");

System.out.println("Running test: assertNotNull(new Object())");

assertNotNull(new Object());

System.out.println("✔ 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.**

**Steps:**

**1. Write tests using the AAA pattern.**

**2. Use @Before and @After annotations for setup and teardown methods.**

**Code:**

**CalculatorAAATest.java:**

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorAAATest {

Calculator calculator;

@Before

public void setUp() {

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

calculator = new Calculator();

}

@After

public void tearDown() {

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

calculator = null;

}

@Test

public void testAdd() {

// Arrange

int a = 10;

int b = 5;

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

assertEquals(15, result);

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

}

@Test

public void testMultiply() {

// Arrange

int a = 4;

int b = 6;

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

assertEquals(24, result);

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

}

}

**Output:**

