**JUnit, Mockito, SLF4J**

**JUnit - Basic Testing**

**Exercise 1: Setting Up JUnit**

**Step 1: Create a New Maven Project**

Open IntelliJ IDEA.

Click New Project.

In the left panel, select Maven.

Click Next.

**Step 2: Fill in Project Details**

GroupId: org.example

ArtifactId: JUnitBasics

Click Next, then Finish.

**Step 3: Add JUnit Dependency**

Open pom.xml.

Inside the <dependencies> section, paste this:

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

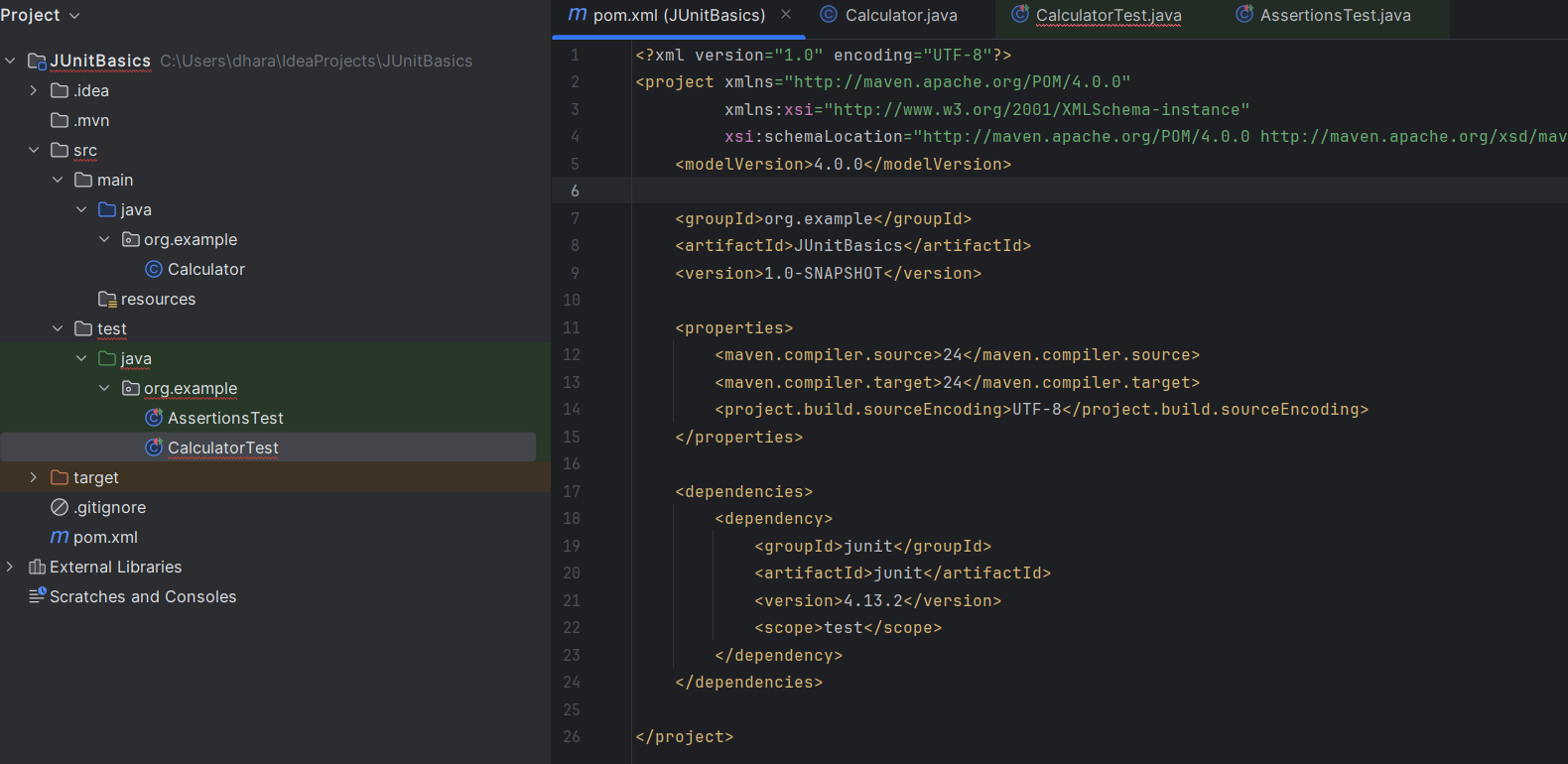
</dependency>

**Step 4: Creating new test class**

src/main/java

src/test/java

**Output:**



**Exercise 2: Writing Basic JUnit Tests**

**Step 1: Java test class**

Right-click src/main/java → New → Java Class → name it Calculator

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

**Step 2: JUnit test class**

Right-click src/test/java → New → Java Class → name it CalculatorTest

package com.example;

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));

}

@Test

public void testSubtract() {

Calculator calc = new Calculator();

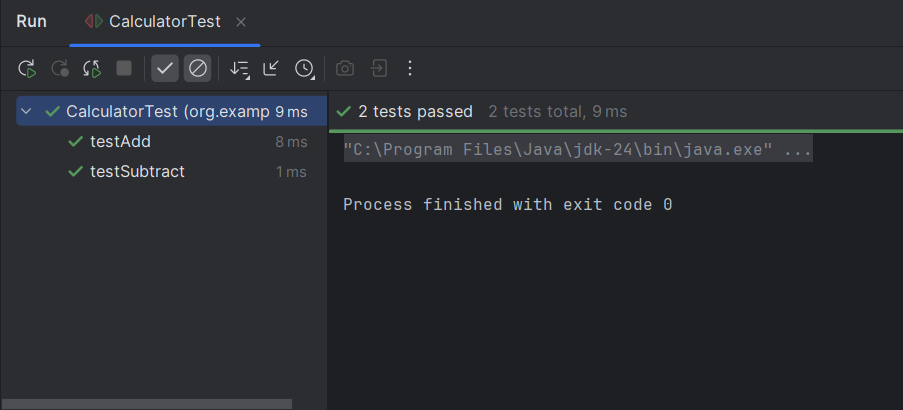
assertEquals(1, calc.subtract(4, 3));

}

}

**Step 3: Run the test**

**Output:**



**Exercise 3: Assertions in JUnit**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

// Act & Assert

// assertEquals: checks equality

assertEquals(5, 2 + 3);

// assertTrue: condition must be true

assertTrue(5 > 3);

// assertFalse: condition must be false

assertFalse(5 < 3);

// assertNull: checks if value is null

assertNull(null);

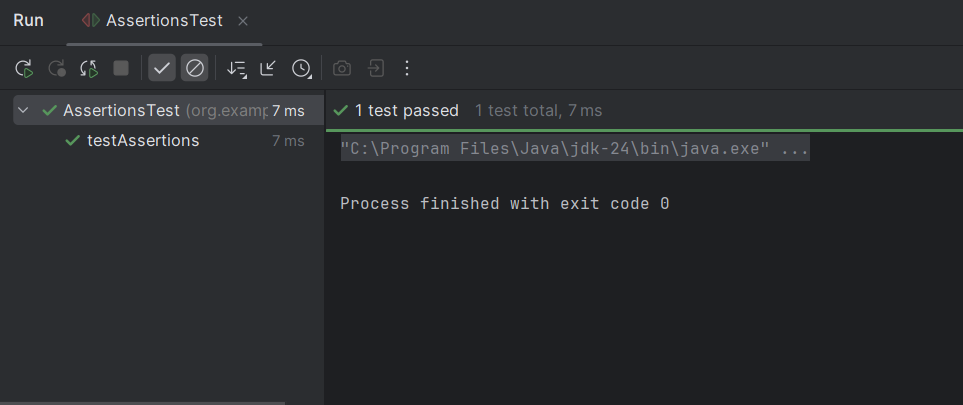
// assertNotNull: checks if value is not null

assertNotNull(new Object());

}

}

**Output:**



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

// CalculatorTest – JUnit test class

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calc;

// Setup method — runs before each test

@Before

public void setUp() {

calc = new Calculator();

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

}

// Teardown method — runs after each test

@After

public void tearDown() {

calc = null;

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

}

@Test

public void testAddition() {

// Arrange

int a = 4, b = 6;

// Act

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

// Assert

assertEquals(10, result);

}

@Test

public void testSubtraction() {

// Arrange

int a = 10, b = 3;

// Act

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

// Assert

assertEquals(7, result);

}

}

// Calculator – Java class

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

**Output:**

