**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.\*;

public class CalculatorTest {

@Test

public void testAdd() {

Calculator calc = new Calculator();

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

assertEquals(5, result);

}

}

**Pom.xml**

<dependency>

<groupId>junit</groupId>

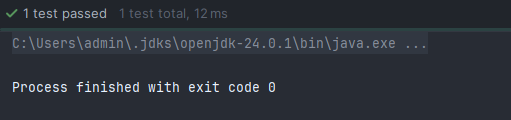
<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

**Output:**

****

**Exercise 3: Assertions in JUnit**

**AssertionsTest.java**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

System.out.println("Running testAssertions...");

assertEquals(5, 2 + 2);

assertTrue(5 > 3);

assertFalse(5 < 3);

assertNull(null);

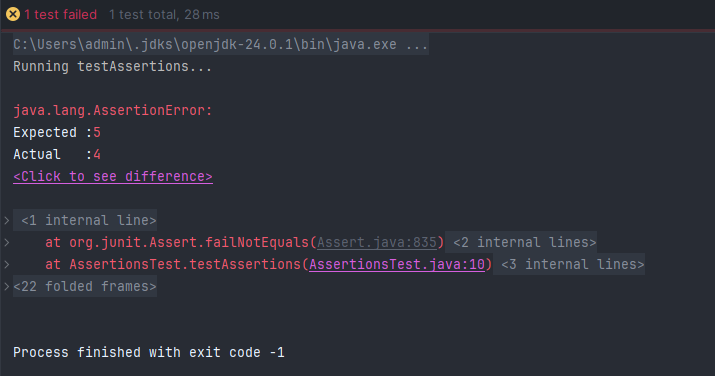
assertNotNull(new Object());

System.out.println("Test completed successfully.");

}

}

**Output:**

****

**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.\*;

public class CalculatorTest {

private Calculator calculator;

@Before

public void setUp() {

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

calculator = new Calculator();

}

@After

public void tearDown() {

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

calculator = null;

}

@Test

public void testAddition() {

// Arrange

int a = 5;

int b = 3;

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

// Assert

assertEquals(8, result);

}

@Test

public void testSubtraction() {

// Arrange

int a = 10;

int b = 4;

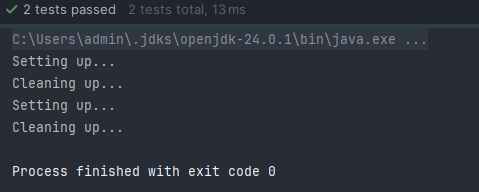
int result = calculator.subtract(a, b);

assertEquals(6, result);

}

}

**Output:**

****