JUnit Testing Exercises

**Exercise 1: Setting Up JUnit**

**Scenario:**

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

**Solution:**

**Tools Used:**

* Eclipse IDE
* Apache Maven
* JUnit 4

1. Created a Maven Project in Eclipse
2. Added JUnit Dependency

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

3 .Created a Sample JUnit Test Case

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

package com.example;

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

import org.junit.jupiter.api.Test;

class CalculatorTest {

@Test

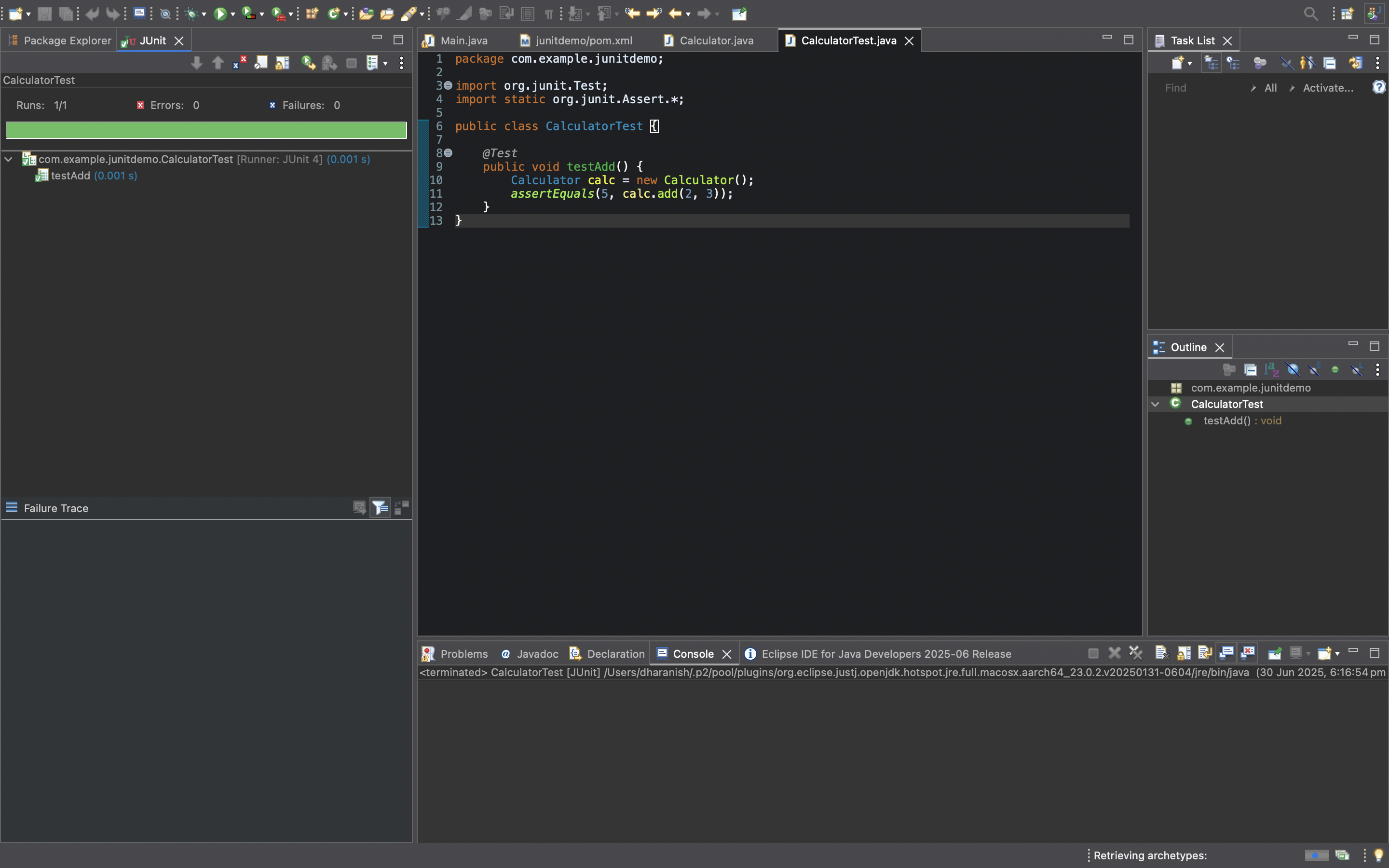
public void testAdd() {

Calculator calc = new Calculator();

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

}

}:

**Output:**

**Exercise 3: Assertions in JUnit**

**Scenario:**

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

**Solution:**

**Tools Used:** Eclipse IDE, JUnit 4, Java, Maven

**Steps Performed:**

1. Created a new test class named AssertionsTest inside the src/test/java folder.
2. Wrote unit tests using different assertion methods provided by JUnit.
3. Used assertions

package com.example.assertionsdemo;

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

*@Test*

public void testAssertions() {

// assertEquals: checks if two values are equal

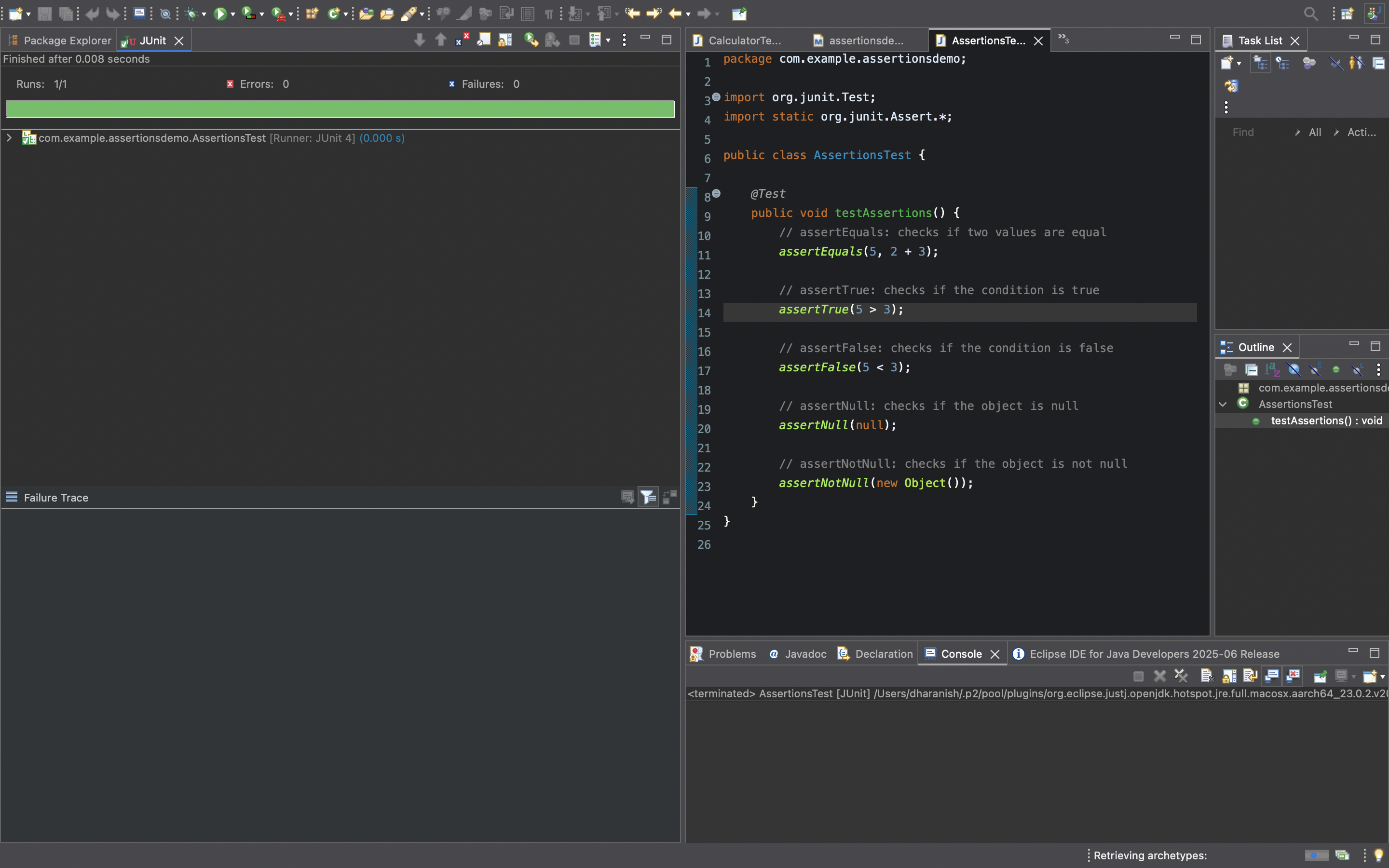
*assertEquals*(5, 2 + 3);

// assertTrue: checks if the condition is true

*assertTrue*(5 > 3);

// assertFalse: checks if the condition is false

*assertFalse*(5 < 3);

// assertNull: checks if the object is null

*assertNull*(null);

// assertNotNull: checks if the object is not null

*assertNotNull*(new Object());

}

}

**Output:**

**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:**

**Tools Used:** Eclipse IDE, JUnit 5, Java, Maven

**Steps Performed:**

1. Created a sample class Calculator with an add() method (in src/main/java).
2. Created a JUnit test class CalculatorTest (in src/test/java).
3. Used the AAA pattern in each test method.
4. Used @BeforeEach to initialize the Calculator object.

**Calculator.java:**

package com.example.junitdemo1;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

**CalculatorTest.java:**

package com.example.junitdemo1;

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() {

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

calculator = new Calculator();

}

*@After*

public void tearDown() {

System.***out***.println("Tearing down calculator...");

calculator = null;

}

*@Test*

public void testAdd() {

// Arrange

int a = 2;

int b = 3;

// Act

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

// Assert

*assertEquals*(5, result);

}

*@Test*

public void testSubtract() {

// Arrange

int a = 10;

int b = 4;

// Act

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

// Assert

*assertEquals*(6, result);

}

}

**Output:**

