

JUnit Testing Exercises

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.

AAA Pattern (Arrange–Act–Assert):

A standard structure for writing unit tests:

1. **Arrange** – Set up objects and variables.
2. **Act** – Invoke the method being tested.
3. **Assert** – Verify the result.

Test Fixtures, `@Before`, and `@After`:

- `@Before`: Runs **before each test method**. Used for common setup code.
- `@After`: Runs **after each test method**. Used for cleanup.

Calculator.java

```
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;  
    }  
}
```

CalculatorTest.java

```
package com.example;

import org.junit.Before;
import org.junit.After;
import org.junit.Test;
import static org.junit.Assert.*;

public class CalculatorTest {

    private Calculator calculator;

    // Setup method: runs before each test
    @Before
    public void setUp() {
        calculator = new Calculator(); // Arrange
        System.out.println("Setup: Calculator instance created");
    }

    // Teardown method: runs after each test
    @After
    public void tearDown() {
        calculator = null;
        System.out.println("Teardown: Calculator instance cleared");
    }

    @Test
    public void testAddition() {
        // Arrange (done in setUp)

        // Act
        int result = calculator.add(10, 5);

        // Assert
```

```
        assertEquals("10 + 5 should be 15", 15, result);
    }

    @Test
    public void testSubtraction() {
        // Arrange (done in setUp)

        // Act

        int result = calculator.subtract(10, 4);

        // Assert

        assertEquals("10 - 4 should be 6", 6, result);
    }
}
```

This confirms that:

- **@Before setUp()** ran before each test method.
- **@After tearDown()** ran after each test method.
- **Each test is isolated** — test environment is fresh for each.

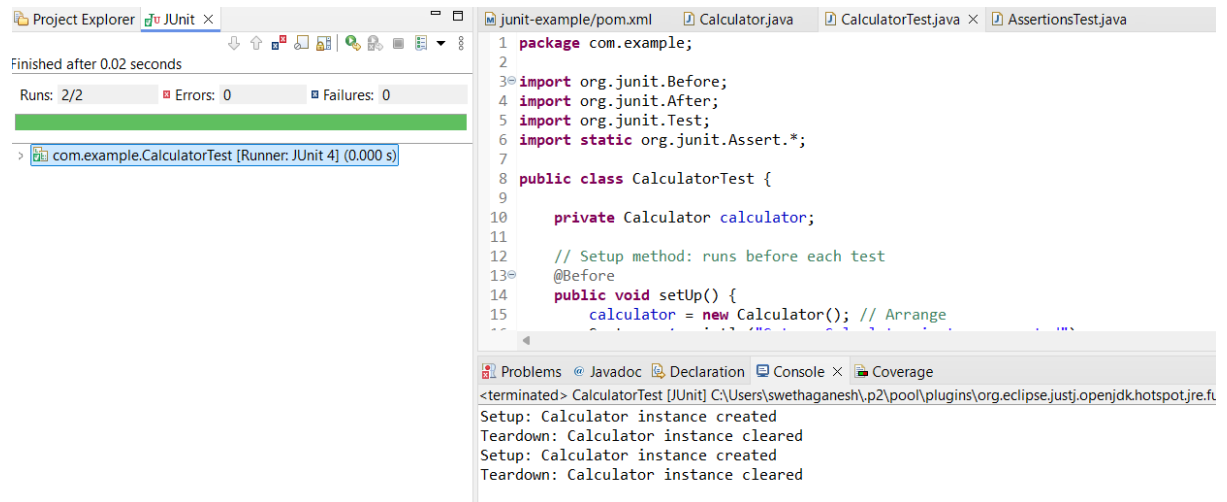
Explanation of AAA Pattern Used:

Arrange → Calculator object initialized in @Before method

Act → Operation (add, subtract) executed inside test

Assert → Result verified using assertEquals

OUTPUT:



The screenshot displays an IDE interface with the following components:

- Project Explorer:** Shows the project structure with files `junit-example/pom.xml`, `Calculator.java`, `CalculatorTest.java`, and `AssertionsTest.java`.
- JUnit Runner:** Indicates the test is finished after 0.02 seconds. The summary shows **Runs: 2/2**, **Errors: 0**, and **Failures: 0**. A green progress bar is visible.
- Test Results:** A list showing `> com.example.CalculatorTest [Runner: JUnit 4] (0.000 s)`.
- Source Editor:** Displays the `CalculatorTest.java` file with the following code:

```
1 package com.example;
2
3 import org.junit.Before;
4 import org.junit.After;
5 import org.junit.Test;
6 import static org.junit.Assert.*;
7
8 public class CalculatorTest {
9
10     private Calculator calculator;
11
12     // Setup method: runs before each test
13     @Before
14     public void setUp() {
15         calculator = new Calculator(); // Arrange
16     }
17 }
```
- Console:** Shows the execution output:

```
<terminated> CalculatorTest [JUnit] C:\Users\swethaganesh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.f
Setup: Calculator instance created
Teardown: Calculator instance cleared
Setup: Calculator instance created
Teardown: Calculator instance cleared
```