WEEK 2

Mandatory Hands-On

**Test Driven Development**

**EX: 4 Arrange-Act-Assert Pattern, Test Fixtures, Setup and Teardown Methods in JUnit**

**ARRANGE-ACT-ASSERT (AAA) PATTERN IN JUNIT:**

The **AAA pattern** is a widely used and clean structure for writing unit tests. It divides a test method into **three logical parts**:

**1. Arrange**

* Set up the test data and objects needed for the test.
* Example: Creating objects, initializing variables, setting conditions.

**2. Act**

* Perform the actual operation you're testing.
* Example: Call the method under test.

**3. Assert**

* Verify that the result of the operation matches the expected outcome.
* Example: Use assertEquals(), assertTrue(), etc.

### ****TEARDOWN METHODS IN JUNIT:****

**Teardown methods** are special methods in JUnit that are executed **after each test method** to perform **cleanup tasks**. These methods help restore the test environment to its original state so that each test runs in isolation without being affected by previous tests.

| **Method Type** | **Annotation** | **Purpose** |
| --- | --- | --- |
| Setup | @Before (JUnit 4) or @BeforeEach (JUnit 5) | Runs **before each test method** |
| Teardown | @After (JUnit 4) or @AfterEach (JUnit 5) | Runs **after each test method** |

## MAIN CLASS — 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;

}

}

## TEST CLASS — 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 calc;

@Before

public void setUp() {

calc = new Calculator();

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

}

@After

public void tearDown() {

calc = null;

System.out.println("Teardown: Calculator object cleared");

}

@Test

public void testAddition() {

int result = calc.add(4, 6);

assertEquals(10, result);

System.out.println("Test Addition Passed");

}

@Test

public void testSubtraction() {

int result = calc.subtract(9, 4);

assertEquals(5, result);

System.out.println("Test Subtraction Passed");

}

}

## SAMPLE OUTPUT

Setup: Calculator object created

Test Addition Passed

Teardown: Calculator object cleared

Setup: Calculator object created

Test Subtraction Passed

Teardown: Calculator object cleared

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

BUILD SUCCESSFUL