## **📄 1. pom.xml (with JUnit 4)**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>JUnitAAADemo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

## **📄 2. Calculator.java**

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int multiply(int a, int b) {

return a \* b;

}

public int divide(int a, int b) {

if (b == 0) throw new IllegalArgumentException("Division by zero");

return a / b;

}

}

## **📄 3. CalculatorTest.java (AAA Pattern + @Before/@After)**

package com.example;

import static org.junit.Assert.\*;

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

public class CalculatorTest {

private Calculator calc;

// Setup method - runs before every test

@Before

public void setUp() {

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

calc = new Calculator();

}

// Teardown method - runs after every test

@After

public void tearDown() {

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

calc = null;

}

@Test

public void testAddition() {

// Arrange

int a = 2, b = 3;

// Act

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

// Assert

assertEquals(5, result);

}

@Test

public void testMultiplication() {

// Arrange

int a = 4, b = 5;

// Act

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

// Assert

assertEquals(20, result);

}

@Test(expected = IllegalArgumentException.class)

public void testDivideByZero() {

// Arrange

int a = 10, b = 0;

// Act

calc.divide(a, b);

// Assert is implicit via expected exception

}

}

