**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.

**Solution:**

Open IntelliJ IDE and create a java project.

In the main class,create a java file Calculator.java

**Calculator.java:**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(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("Cannot divide by zero");

}

return a / b;

}

}

Next in the test class create a java file,CalculatorTest.java.

**CalculatorTest.java:**

package org.example;  
  
import org.junit.After;  
import org.junit.Before;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
  
public class CalculatorTest {  
  
 Calculator calculator;  
  
  
 @Before  
 public void setUp() {  
 calculator = new Calculator();  
 System.*out*.println("Setup: Calculator object created");  
 }  
  
  
 @After  
 public void tearDown() {  
 calculator = null;  
 System.*out*.println("Teardown: Calculator object destroyed");  
 }  
  
  
 @Test  
 public void testAddition() {  
  
 int a = 5;  
 int b = 3;  
  
  
 int result = calculator.add(a, b);  
  
   
 *assertEquals*(8, result);  
 }  
  
  
 @Test  
 public void testSubtraction() {  
  
 int a = 10;  
 int b = 4;  
  
   
 int result = calculator.subtract(a, b);  
  
  
 *assertEquals*(6, result);  
 }  
  
  
 @Test(expected = IllegalArgumentException.class)  
 public void testDivideByZero() {  
   
 calculator.divide(5, 0);  
  
   
 }  
}

A screenshot of a computer

AI-generated content may be incorrect.**Output:**