JUnit\_Basic Testing Exercises (Mandatory Hands On)

Exercise 1: Setting Up Junit

Code:

package com.example.junit;

import static org.junit.Assert.*assertEquals*;

import org.junit.Test;

public class CalculatorTest {

public int add(int a, int b) {

return a + b;

}

*@Test*

public void testAddition() {

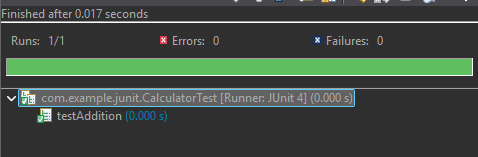
int result = add(2, 3);

*assertEquals*(5, result); // this test will pass

}

}

Output:



Exercise 3: Assertions in Junit

Code:

package com.example.junit;

import static org.junit.Assert.\*; // Required for assertions

import org.junit.Test;

public class AssertionsTest {

*@Test*

public void testAssertions() {

// Assert equals

*assertEquals*(5, 2 + 3);

// Assert true

*assertTrue*(5 > 3);

// Assert false

*assertFalse*(5 < 3);

// Assert null

*assertNull*(null);

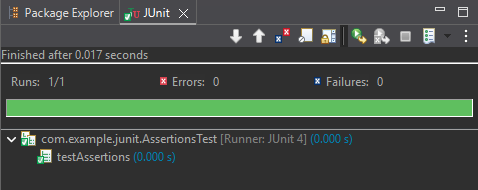
// Assert not null

*assertNotNull*(new Object());

}

}

Output:



Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and

Teardown Methods in Junit

Code:

package com.example.junit;

import static org.junit.Assert.\*;

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

public class CalculatorTest {

private Calculator calculator;

// Setup method (executed before each test)

*@Before*

public void setUp() {

calculator = new Calculator();

System.***out***.println("Setup: Calculator initialized.");

}

// Teardown method (executed after each test)

*@After*

public void tearDown() {

calculator = null;

System.***out***.println("Teardown: Calculator cleaned up.");

}

*@Test*

public void testAddition() {

// Arrange is handled by @Before

// Act

int result = calculator.add(5, 3);

// Assert

*assertEquals*(8, result);

}

*@Test*

public void testSubtraction() {

// Act

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

// Assert

*assertEquals*(6, result);

}

}

package com.example.junit;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

Output:

