**JUnit Testing Exercises**  
**Exercise 1:** Setting Up JUnit Scenario: You need to set up JUnit in your Java project to start writing unit tests.

Steps:

1. Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse).

2. Add JUnit dependency to your project. If you are using Maven, add the following to your pom.xml: junit junit 4.13.2 test

3. Create a new test class in your project.  
  
**Code Section:  
Pom.xml**<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>j\_unit\_example</artifactId>

<version>1.0-SNAPSHOT</version>

<properties>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**Main.java**package com.example;

public class Main {

public int add(int a, int b) {

return a + b;

}

}

**MainTest.java**

package com.example;

import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class MainTest {

@Test

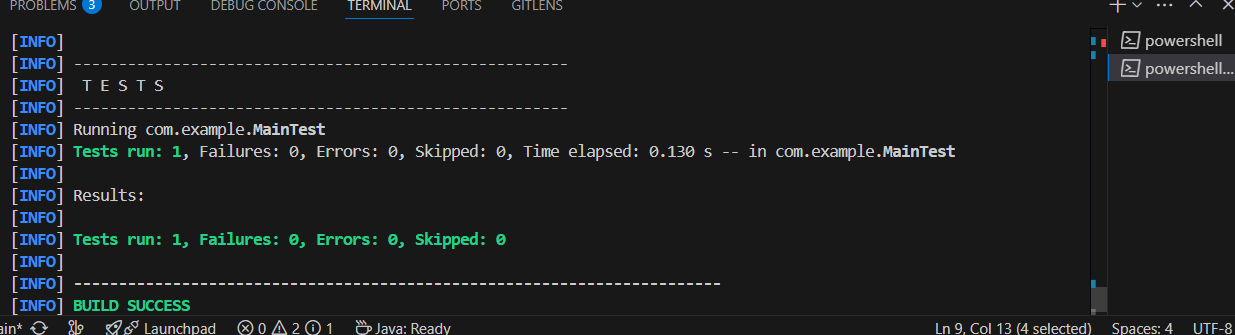
public void testAdd() {

Main main = new Main();

int result = main.add(3, 4);

assertEquals(7, result);

}

}  
**OUTPUT:**

**Exercise 3:**

**Assertions in JUnit Scenario: You need to use different assertions in JUnit to validate your test results.**

**Steps:**

**1. Write tests using various JUnit assertions.  
  
Code:**package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class MainTest {

    @Test

    public void testAssertions() {

        assertEquals(5, 2 + 3);

        assertTrue(5 > 3);

        assertFalse(5 < 3);

        assertNull(null);

        assertNotNull(new Object());

    }

}

**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.  
  
CODE SECTION  
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  
// CalculatorTest.java**

package com.example;

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

    private Calculator calculator;

    @Before

    public void setUp() {

        calculator = new Calculator();

    }

    @After

    public void tearDown() {

        calculator = null;

    }

    @Test

    public void testAddition() {

        // Arrange

        int a = 5, b = 3;

        // Act

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

        // Assert

        assertEquals(8, result);

    }

    @Test

    public void testSubtraction() {

        // Arrange

        int a = 10, b = 4;

        // Act

        int result = calculator.subtract(a, b);

        // Assert

        assertEquals(6, result);

    }

}  
  
**OUTPUT:**  
