**JUnit Testing Exercises**

**Exercise 1: Setting Up Junit**

pom.xml:

<?xml version="1.0" encoding="UTF-8"?>

<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>demo</artifactId>

  <version>1.0-SNAPSHOT</version>

  <properties>

    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

    <maven.compiler.source>21</maven.compiler.source>

    <maven.compiler.target>21</maven.compiler.target>

  </properties>

  <dependencies>

    <dependency>

      <groupId>junit</groupId>

      <artifactId>junit</artifactId>

      <version>4.13.2</version>

      <scope>test</scope>

    </dependency>

  </dependencies>

  <build>

    <plugins>

      <plugin>

        <artifactId>maven-clean-plugin</artifactId>

        <version>3.1.0</version>

      </plugin>

      <plugin>

        <artifactId>maven-resources-plugin</artifactId>

        <version>3.0.2</version>

      </plugin>

      <plugin>

        <artifactId>maven-compiler-plugin</artifactId>

        <version>3.11.0</version>

      </plugin>

      <plugin>

        <artifactId>maven-surefire-plugin</artifactId>

        <version>3.1.2</version>

      </plugin>

      <plugin>

        <artifactId>maven-jar-plugin</artifactId>

        <version>3.0.2</version>

      </plugin>

      <plugin>

        <artifactId>maven-install-plugin</artifactId>

        <version>2.5.2</version>

      </plugin>

      <plugin>

        <artifactId>maven-deploy-plugin</artifactId>

        <version>2.8.2</version>

      </plugin>

      <plugin>

        <artifactId>maven-site-plugin</artifactId>

        <version>3.7.1</version>

      </plugin>

      <plugin>

        <artifactId>maven-project-info-reports-plugin</artifactId>

        <version>3.0.0</version>

      </plugin>

    </plugins>

  </build>

</project>

**AppTest.java**

package com.example;

import static org.junit.Assert.assertTrue;

import org.junit.Test;

/\*\*

 \* Unit test for simple App.

 \*/

public class AppTest

{

    /\*\*

     \* Rigorous Test :-)

     \*/

    @Test

    public void shouldAnswerWithTrue()

    {

        assertTrue( true );

    }

}

**Ouput:**



**Exercise 2: Writing Basic JUnit Tests**

**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;

    }

    public int multiply(int a, int b) {

        return a \* b;

    }

}

**CalculatorTest.java:**

package com.example;

import static org.junit.Assert.assertEquals;

import org.junit.Test;

public class CalculatorTest {

    @Test

    public void testAdd() {

        Calculator calc = new Calculator();

        assertEquals(10, calc.add(7, 3));

    }

    @Test

    public void testSubtract() {

        Calculator calc = new Calculator();

        assertEquals(4, calc.subtract(7, 3));

    }

    @Test

    public void testMultiply() {

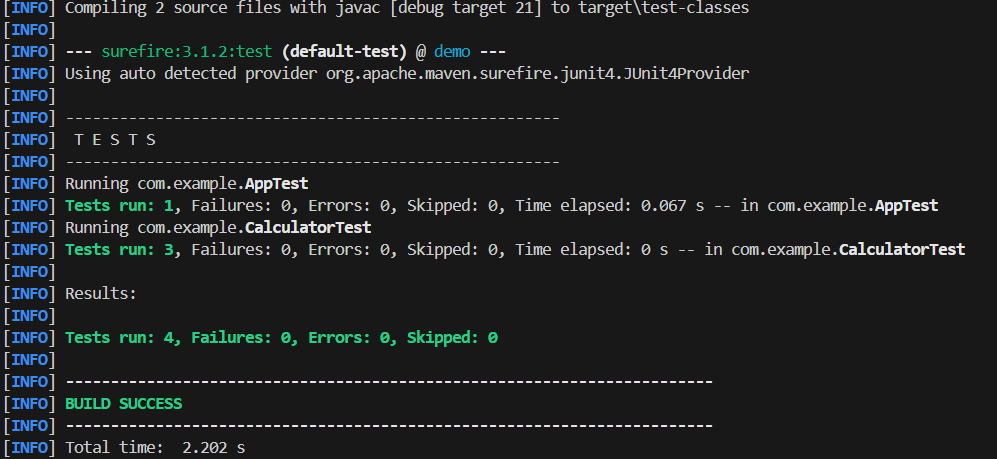
        Calculator calc = new Calculator();

        assertEquals(21, calc.multiply(7, 3));

    }

}

**Output:**

****

**Exercise 3: Assertions in Junit**

**AssertionsTest.java**

package com.example;

import static org.junit.Assert.assertEquals;

import static org.junit.Assert.assertFalse;

import static org.junit.Assert.assertNotNull;

import static org.junit.Assert.assertNull;

import static org.junit.Assert.assertTrue;

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);

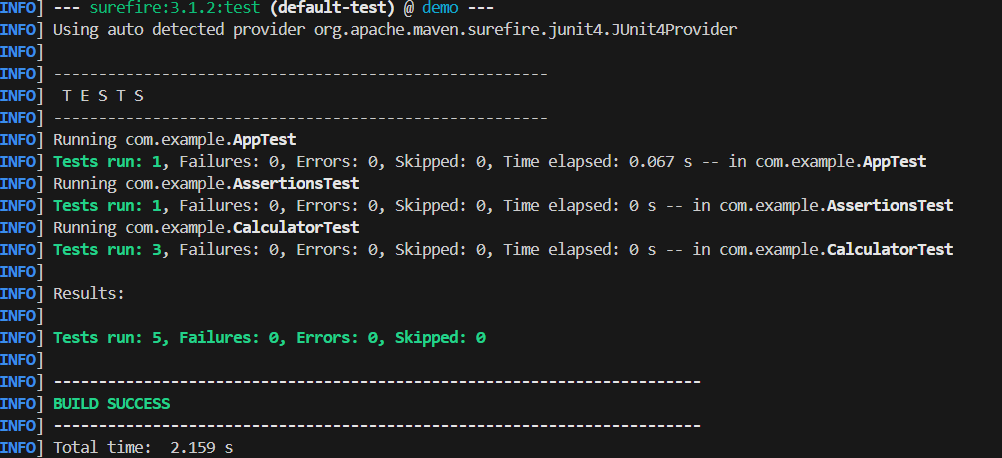
        // Assert not null

        assertNotNull(new Object());

    }

}

**Output:**



**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

**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;

    }

    public int multiply(int a, int b) {

        return a \* b;

    }

    public int divide(int a, int b) {

    if (b == 0) throw new ArithmeticException("Cannot divide by zero");

    return a / b;

}

}

**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;

    // Setup before each test

    @Before

    public void setUp() {

        calculator = new Calculator();

        System.out.println("Setup completed sucessfully");

    }

    // Teardown after each test

    @After

    public void tearDown() {

        System.out.println("Test completed sucessfully");

    }

    @Test

    public void shouldReturnSumOfTwoNumbers() {

        // Arrange done in setup

        // Act

        int result = calculator.add(10, 5);

        // Assert

        assertEquals(15, result);

    }

    @Test

    public void shouldReturnDifferenceOfTwoNumbers() {

        int result = calculator.subtract(20, 8);

        assertEquals(12, result);

    }

    @Test

    public void shouldReturnProductOfTwoNumbers() {

        int result = calculator.multiply(6, 7);

        assertEquals(42, result);

    }

    @Test

    public void shouldReturnQuotientOfTwoNumbers() {

        int result = calculator.divide(20, 4);

        assertEquals(5, result);

    }

    @Test(expected = ArithmeticException.class)

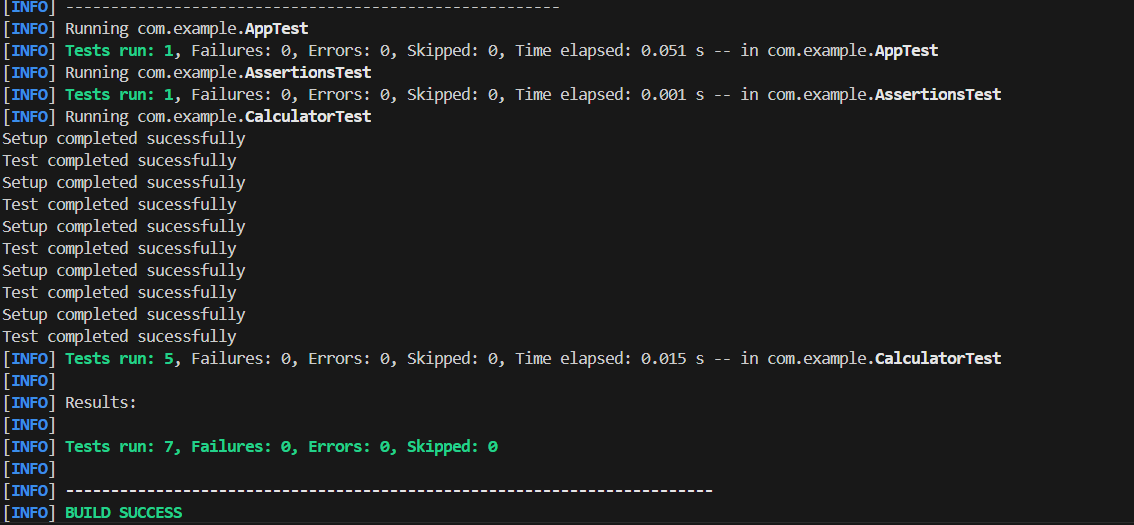
    public void shouldThrowExceptionWhenDivideByZero() {

        calculator.divide(10, 0);

    }

}

**Output:**

****