JUnit Testing Exercises

# Exercise 1: Setting Up JUnit

**Calculator.java**

package junitdemo;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**CalculatorTest.java**

package junitdemo;

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;

class CalculatorTest {

@Test

void testAdd() {

Calculator calc = new Calculator();

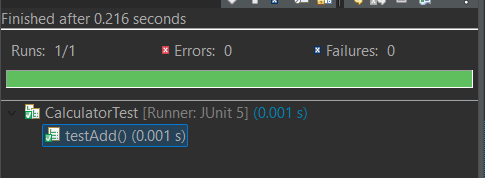
int result = calc.add(3, 4);

assertEquals(7, result);

}

}

**Output**



# Exercise 2: Writing Basic JUnit Tests

# MathUtils.java

package junitdemo;

public class MathUtils {

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("Division by zero not allowed.");

return a / b;

}

}

**MathUtilsTest.java**

package junitdemo;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

class MathUtilsTest {

MathUtils math = new MathUtils();

@Test

void testAdd() {

assertEquals(10, math.add(6, 4));

}

@Test

void testSubtract() {

assertEquals(2, math.subtract(6, 4));

}

@Test

void testMultiply() {

assertEquals(24, math.multiply(6, 4));

}

@Test

void testDivide() {

assertEquals(2, math.divide(8, 4));

}

@Test

void testDivideByZero() {

Exception exception = assertThrows(IllegalArgumentException.class, () -> {

math.divide(5, 0);

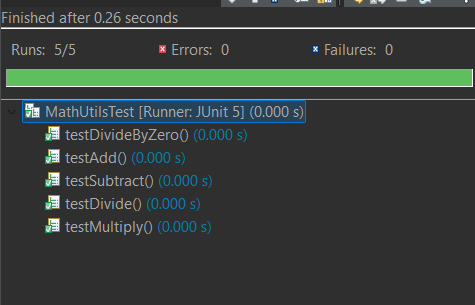
});

assertEquals("Division by zero not allowed.", exception.getMessage());

}

}

**Output**



# Exercise 3: Assertions in JUnit

**AssertionsTest.java**

package junitdemo;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

public class AssertionsTest {

@Test

public void testAssertions() {

assertEquals(5, 2 + 3);

assertTrue(5 > 3);

assertFalse(5 < 3);

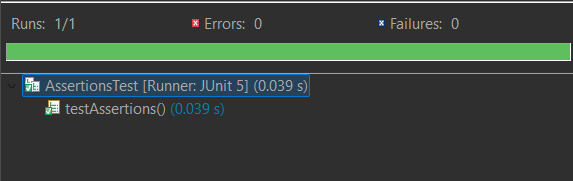
assertNull(null);

assertNotNull(new Object());

}

}

**Output**



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

**MathUtils.java**

package junitdemo;

public class MathUtils {

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("Division by zero not allowed.");

return a / b;

}

}

**MathUtilsTest.java**

package junitdemo;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.\*;

public class MathUtilsTest {

MathUtils mathUtils;

@BeforeEach

void setUp() {

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

mathUtils = new MathUtils();

}

@AfterEach

void tearDown() {

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

mathUtils = null;

}

@Test

void testAdd() {

int a = 10;

int b = 5;

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

assertEquals(15, result);

}

@Test

void testSubtract() {

int a = 10;

int b = 4;

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

assertEquals(6, result);

}

}

**Output**

