**Junit Frame Work & TDD**

**Exercise1 – Setting Up Junit**

1. **Created a new Java project in Eclipse**
2. **Created a new test class in your project.**

**Code:**

package testing;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**Test Class for Calculator**

package testing;

import org.junit.jupiter.api.Test;

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

public class CalculatorTest {

*@Test*

void testAdd() {

Calculator calc = new Calculator();

*assertEquals*(5, calc.add(2, 3));

}

A computer screen shot of a program

AI-generated content may be incorrect.}

**Output:**

Tests passed

**Exercise 2:** **Writing Basic JUnit Tests**

**&&**

**Exercise 3: Assertions in JUnit**

1. **Created calculator class with add(), subtract(), multiply(), divide(), ispositive() methods.**

**Code:**

package testing;

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) { return a / b; }

public boolean isPositive(int a) { return a > 0; }

}

1. **Wrote JUnit tests for these methods and also used different assertions to validate test results.**

package testing;

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

import org.junit.jupiter.api.Test;

public class CalculatorTest {

Calculator calc = new Calculator();

@Test

public void testAddition() {

assertEquals(5, calc.add(2, 3), "Addition should return 5");

}

@Test

public void testSubtraction() {

assertNotEquals(0, calc.subtract(5, 5), "Subtraction should not return 0");

}

@Test

public void testMultiplication() {

assertEquals(6, calc.multiply(2, 3));

}

@Test

public void testDivision() {

assertThrows(ArithmeticException.class, () -> calc.divide(5, 0));

}

@Test

public void testPositive() {

assertTrue(calc.isPositive(10));

assertFalse(calc.isPositive(-5));

}

@Test

public void testNull() {

String str = null;

assertNull(str);

}

@Test

public void testNotNull() {

String str = "JUnit";

assertNotNull(str);

}

}

**Output:**

**Subtract test failed and remaining tests passed**

A screenshot of a computer

AI-generated content may be incorrect.

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

1. **Writen tests using the AAA pattern.**
2. **Used @Before and @After annotations for setup and teardown methods.**

**Code:**

package AAA;

public class TemperatureConverter {

public double celsiusToFahrenheit(double celsius) {

return (celsius \* 9 / 5) + 32;

}

public double fahrenheitToCelsius(double fahrenheit) {

return (fahrenheit - 32) \* 5 / 9;

}

}

package AAA;

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

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

public class TemperatureConverterTest {

TemperatureConverter converter;

*@BeforeEach*

void setUp() {

converter = new TemperatureConverter();

}

*@AfterEach*

void tearDown() {

converter = null; }

*@Test*

void testCelsiusToFahrenheit() {

double result = converter.celsiusToFahrenheit(0);

*assertEquals*(32.0, result);

}

*@Test*

void testFahrenheitToCelsius() {

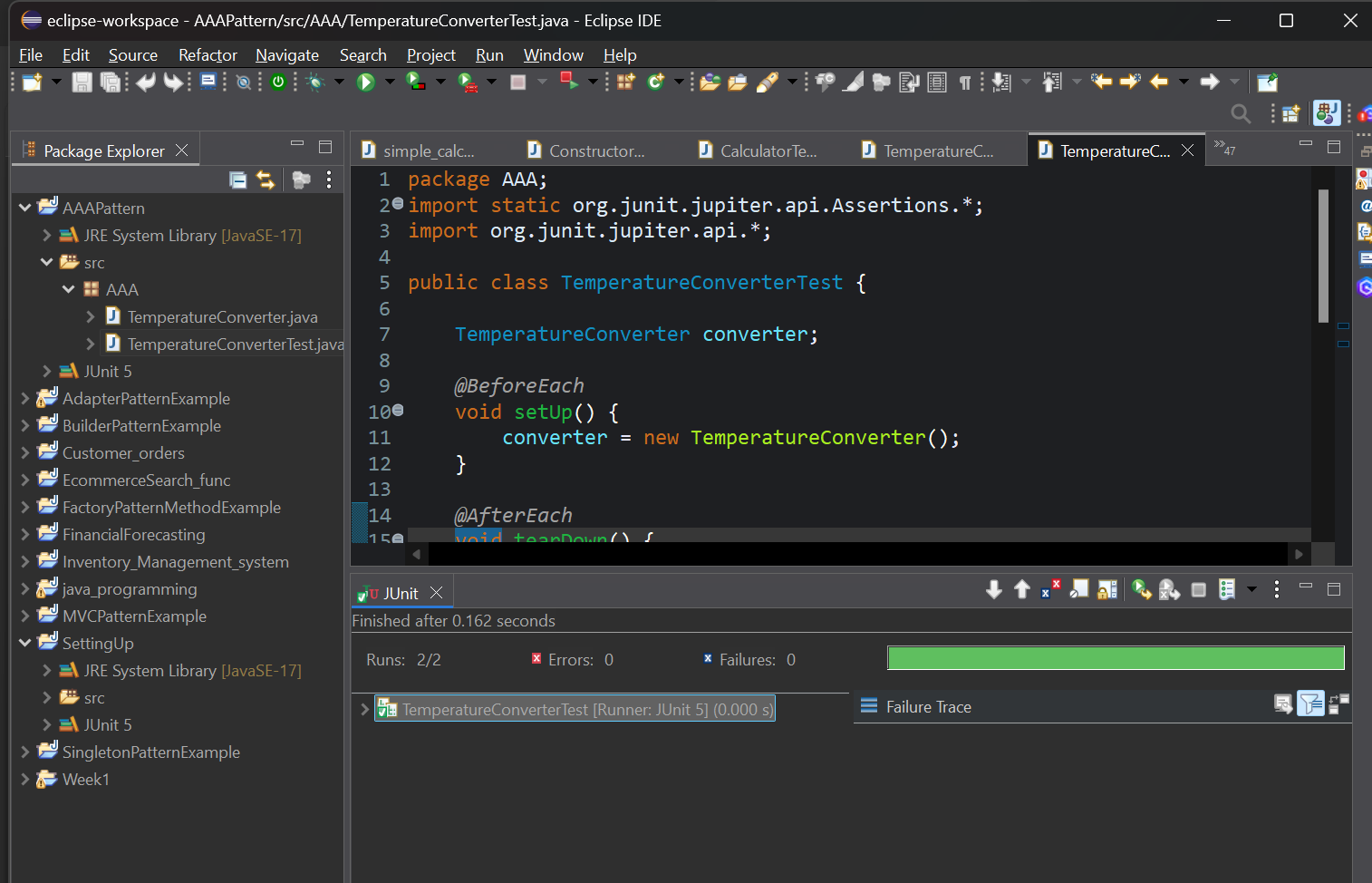
double result = converter.fahrenheitToCelsius(212);

*assertEquals*(100.0, result);

}

}

**Output:**



A screenshot of a computer

AI-generated content may be incorrect.