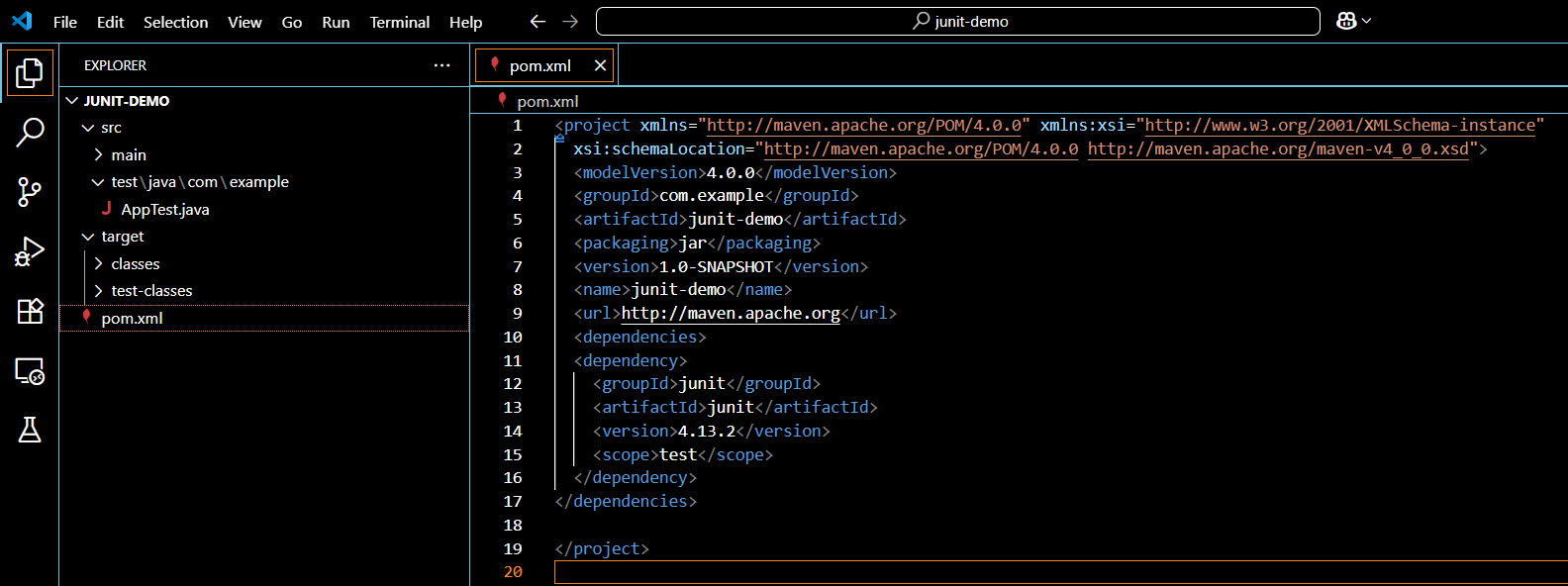
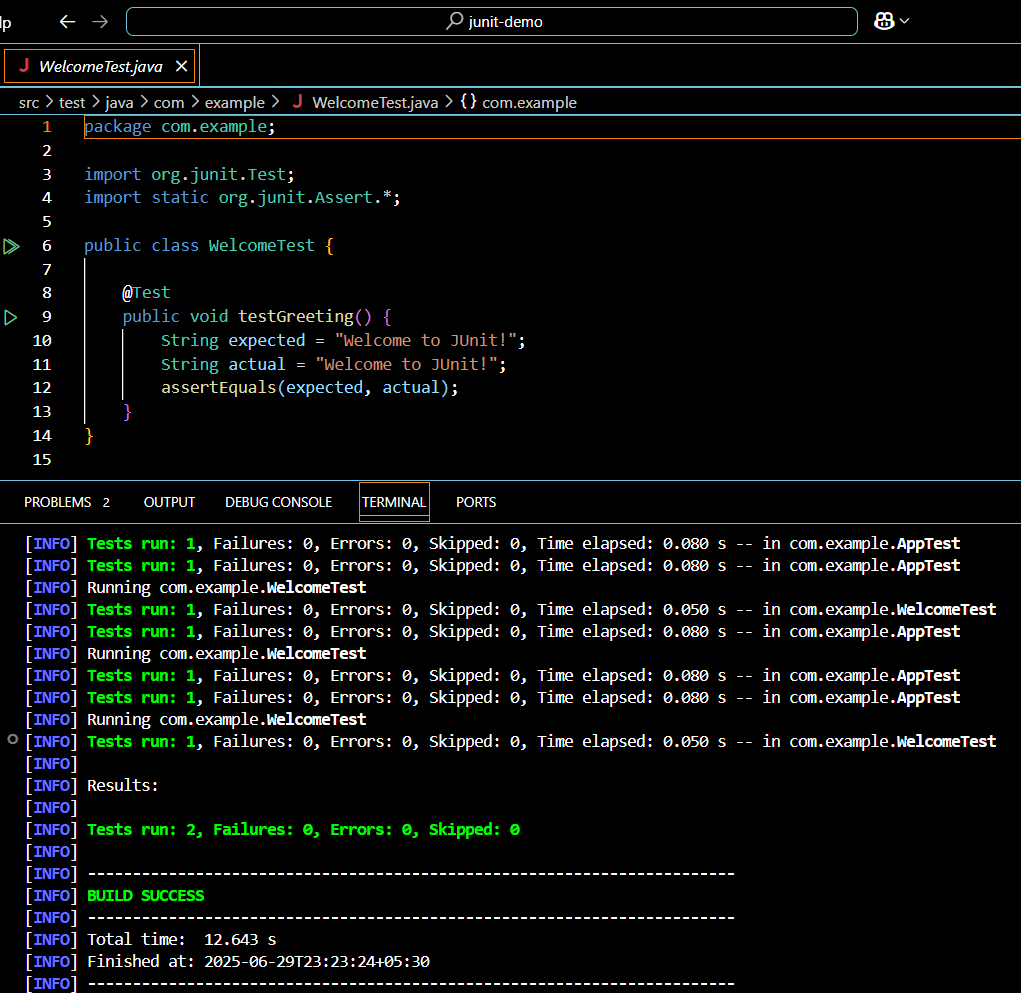
**JUnit Testing Exercises**

Exercise 1: Setting Up Junit





**Exercise 2**: Assertions in Junit

AssertionsTest.java:

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

assertEquals(5, 2 + 3);

assertTrue(10 > 5);

assertFalse(5 > 10);

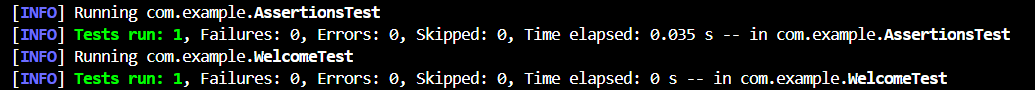
assertNull(null);

assertNotNull("JUnit");

}

}

Output:



**Exercise 3:** 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 multiply(int a, int b) {

return a \* b;

}

}

**CalculatorTest.java:**

package com.example;

import org.junit.\*;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calc;

@Before

public void setUp() {

calc = new Calculator(); // Arrange

System.out.println("Setup complete.");

}

@Test

public void testAddition() {

// Act

int result = calc.add(2, 3);

// Assert

assertEquals(5, result);

}

@Test

public void testMultiplication() {

int result = calc.multiply(4, 5);

assertEquals(20, result);

}

@After

public void tearDown() {

System.out.println("Teardown complete.");

}

}



**Mockito Hands-On Exercises**

**Exercise 4:** Mocking and Stubbing

**ExternalApi.java:**

Import package.com;

public interface ExternalApi {

String getData();

}

**Pom.xml:**

<!-- Mockito Core -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<!-- Add mockito-inline to work with Java 21 -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-inline</artifactId>

<version>5.2.0</version>

<scope>test</scope>

</dependency>

**MyService.java:**

package com.example;

public class MyService {

    private ExternalApi api;

    public MyService(ExternalApi api) {

        this.api = api;

    }

    public String fetchData() {

        return api.getData();

    }

}

**MyServiceTest.java:**

package com.example;

import org.junit.Test;

import static org.mockito.Mockito.\*;

import static org.junit.Assert.\*;

public class MyServiceTest {

    @Test

    public void testExternalApi() {

        ExternalApi mockApi = mock(ExternalApi.class);

        when(mockApi.getData()).thenReturn("Mock Data");

        MyService service = new MyService(mockApi);

        String result = service.fetchData();

        assertEquals("Mock Data", result);

    }

    @Test

    public void testVerifyInteraction() {

        ExternalApi mockApi = mock(ExternalApi.class);

        MyService service = new MyService(mockApi);

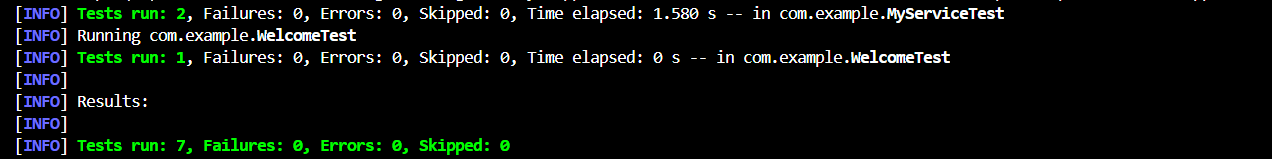
        service.fetchData();

        verify(mockApi).getData();

    }

}

Output:



**Exercise 5:** Verifying Interactions

**ErrorHandlingService.java:**

package com.example;

public class ErrorHandlingService {

    private ExternalApi api;

    public ErrorHandlingService(ExternalApi api) {

        this.api = api;

    }

    public String safeFetch() {

        try {

            return api.getData(); // This might throw

        } catch (RuntimeException e) {

            return "Error handled";

        }

    }

}

**ErrorHandlingServiceTest.java:**

package com.example;

import org.junit.Test;

import static org.mockito.Mockito.\*;

import static org.junit.Assert.\*;

public class ErrorHandlingServiceTest {

    @Test

    public void testApiThrowsException() {

        // Arrange

        ExternalApi mockApi = mock(ExternalApi.class);

        when(mockApi.getData()).thenThrow(new RuntimeException("API failure"));

        ErrorHandlingService service = new ErrorHandlingService(mockApi);

        // Act

        String result = service.safeFetch();

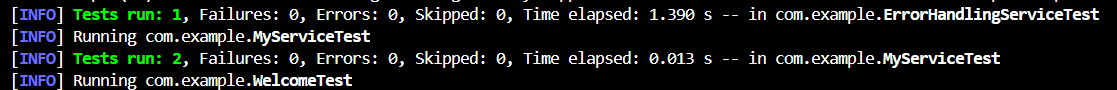
        // Assert

        assertEquals("Error handled", result);

    }

}

**Output:**

****

**Exercise 6:** Logging Error Messages and Warning Levels

**LoggingExample.java:**

package com.example;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class LoggingExample {

    private static final Logger logger = LoggerFactory.getLogger(LoggingExample.class);

    public static void main(String[] args) {

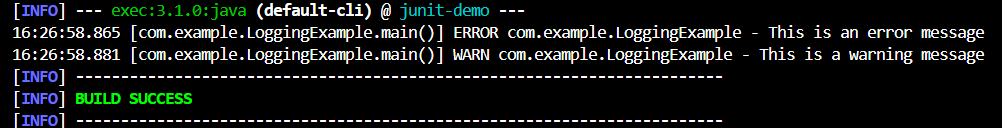
        logger.error("This is an error message");

        logger.warn("This is a warning message");

    }

}

**Output:**

****