**Exercise 1: Setting Up Junit :**

**Project Details:**

* Project Name: JUnitDemoProject
* Tool Used: Eclipse IDE
* Build System: Maven
* Java Version: JDK 20
* JUnit Version: 4.13.2

**Folder Structure:**

JUnitDemoProject/

├── src/

│ ├── main/java/com/example/Calculator.java

│ └── test/java/com/example/CalculatorTest.java

├── pom.xml

**Source Code :**

#### **Calculator.java :**

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**CalculatorTest.java :**

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

@Test

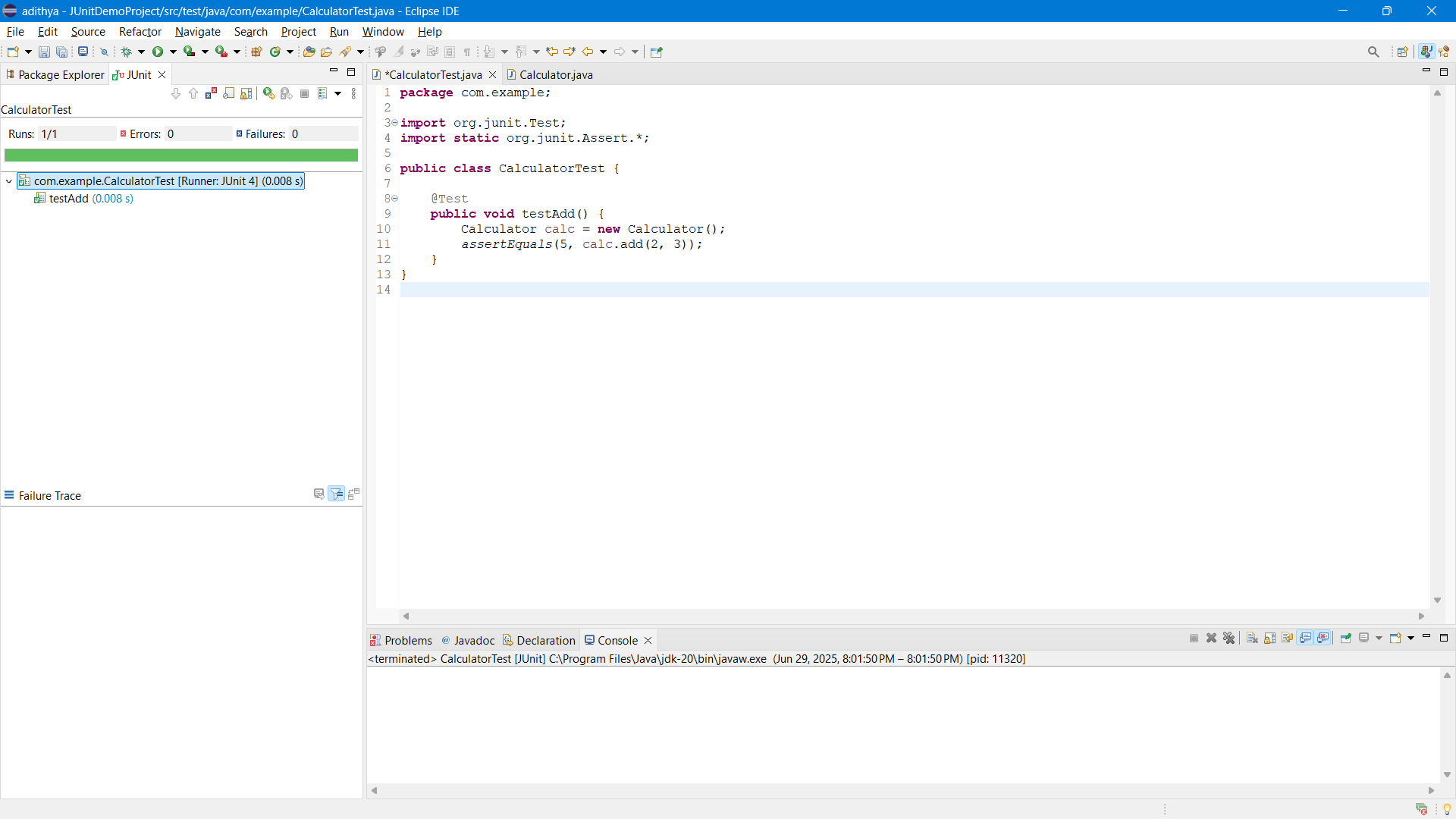
public void testAdd() {

Calculator calc = new Calculator();

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

}

}



**Exercise 3: Assertions in Junit**

**Source Code :**

**AssertionsTest.java :**

**package** com.example;

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

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

**public** **class** AssertionsTest {

@Test

**public** **void** testAssertions() {

*assertEquals*(5, 2 + 3);

*assertTrue*(10 > 1);

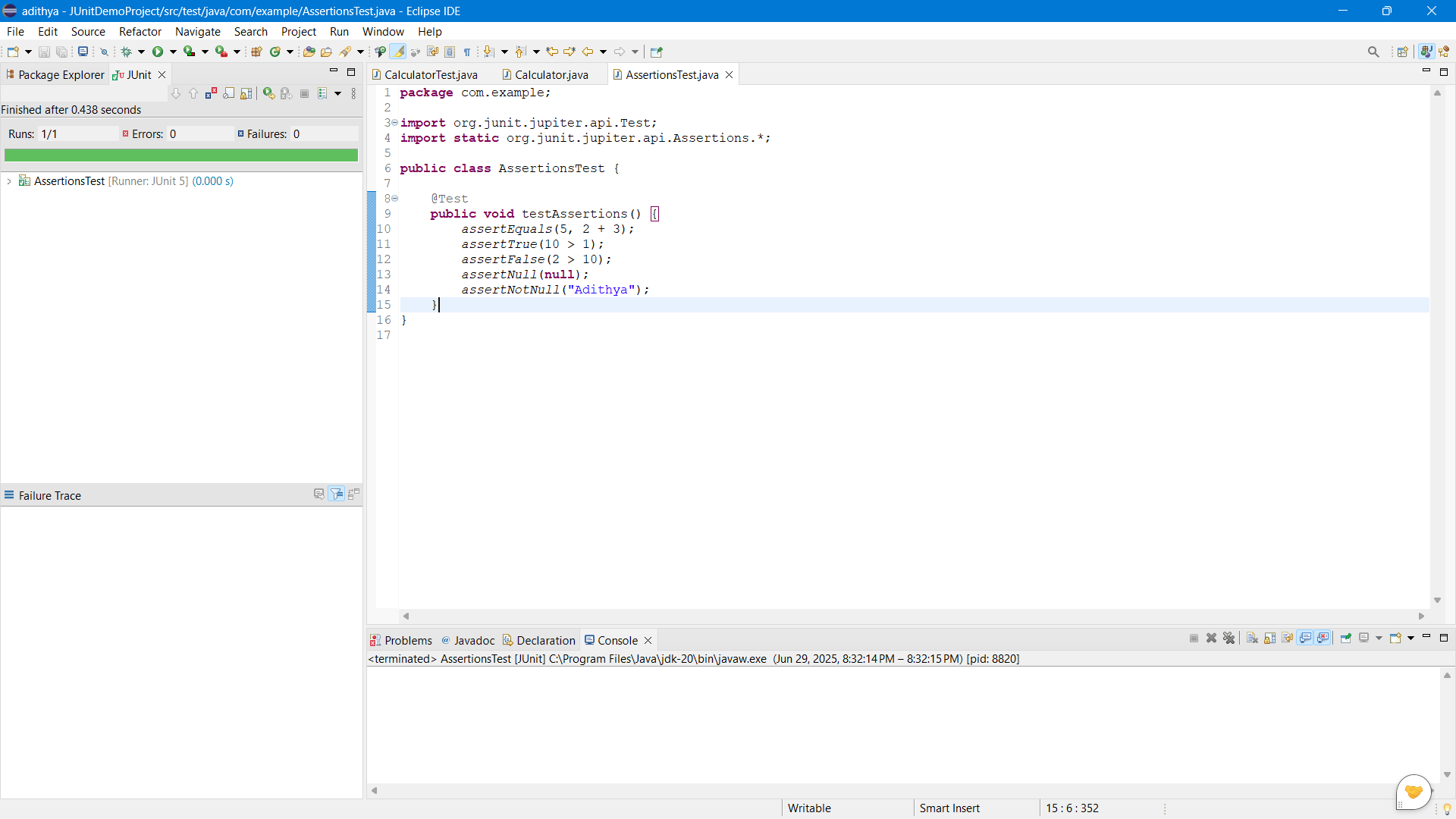
*assertFalse*(2 > 10);

*assertNull*(**null**);

*assertNotNull*("Adithya");

}

}



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

**Teardown Methods in Junit :**

**Source Code :**

**Account.java :**

**package** com.example;

**public** **class** Account {

**private** **int** balance;

**public** Account(**int** initialBalance) {

**this**.balance = initialBalance;

}

**public** **void** deposit(**int** amount) {

balance += amount;

}

**public** **int** getBalance() {

**return** balance;

}

**public** **void** withdraw(**int** amount) {

**if** (amount <= balance) {

balance -= amount;

} **else** {

**throw** **new** IllegalArgumentException("Insufficient funds");

}

}

}

**AccountTest.java :**

**package** com.example;

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

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

**public** **class** AccountTest {

**private** Account account;

@BeforeEach

**void** setUp() {

account = **new** Account(1000);

System.***out***.println("Setup: Account initialized");

}

@AfterEach

**void** tearDown() {

System.***out***.println("Teardown: Test completed\n");

}

@Test

**void** testDeposit() {

// Act

account.deposit(500);

// Assert

*assertEquals*(1500, account.getBalance());

}

@Test

**void** testWithdraw() {

// Act

account.withdraw(300);

// Assert

*assertEquals*(700, account.getBalance());

}

@Test

**void** testWithdraw\_InsufficientFunds() {

// Act + Assert

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

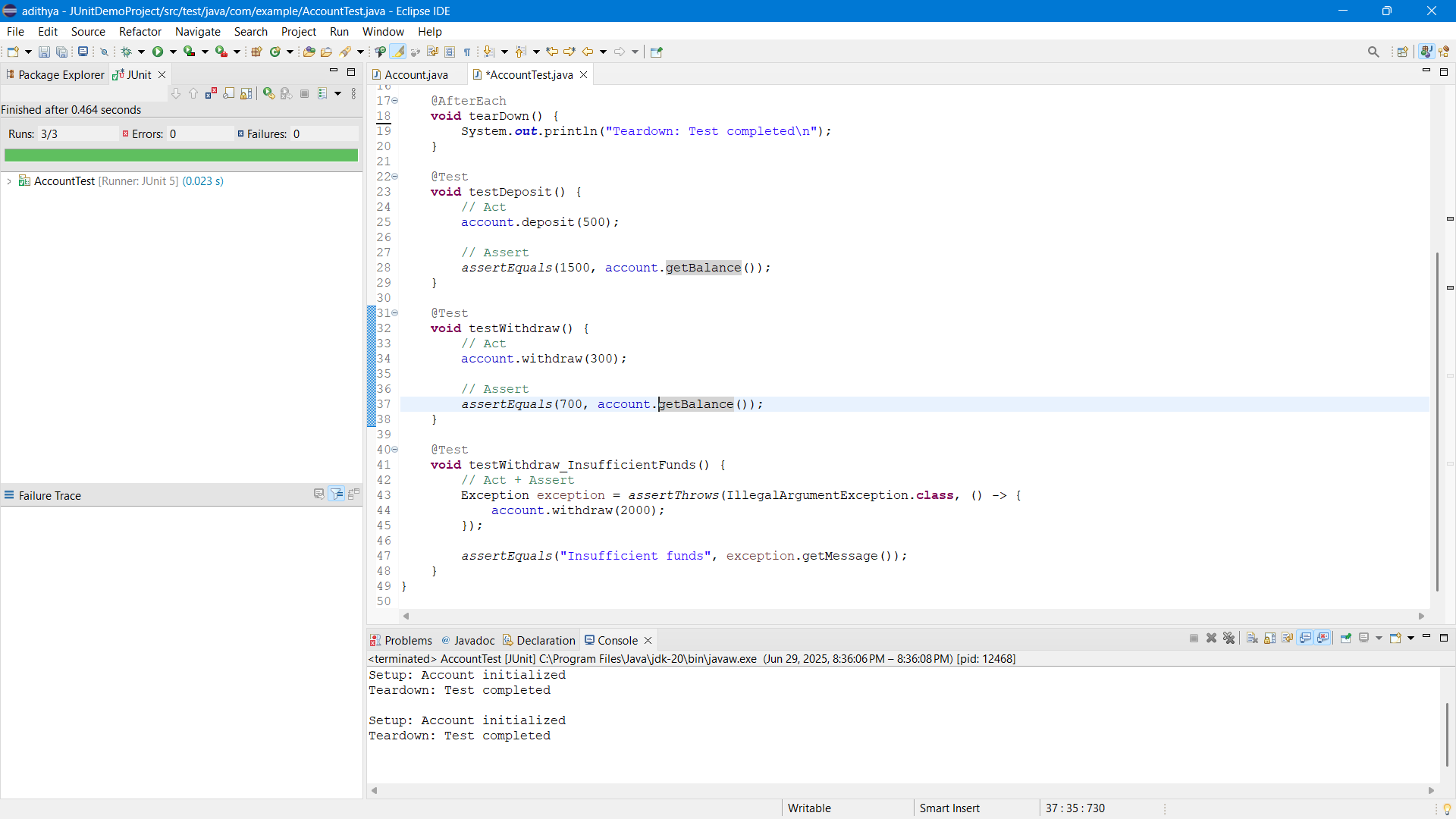
account.withdraw(2000);

});

*assertEquals*("Insufficient funds", exception.getMessage());

}

}



**Exercise 1: Mocking and Stubbing :**

**ExternalApi.java :**

**package** com.example.service;

**public** **interface** ExternalApi {

String getData();

}

**MyService.java:**

**package** com.example.service;

**public** **class** MyService {

**private** ExternalApi api;

**public** MyService(ExternalApi api) {

**this**.api = api;

}

**public** String fetchData() {

**return** api.getData();

}

}

**MyServiceTest.java:**

**package** com.example.test;

**import** com.example.service.ExternalApi;

**import** com.example.service.MyService;

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

**import** **static** org.mockito.Mockito.\*;

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

**public** **class** MyServiceTest {

@Test

**public** **void** testExternalApi() {

// Create mock

ExternalApi mockApi = *mock*(ExternalApi.**class**);

// Stub method

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

// Use in service

MyService service = **new** MyService(mockApi);

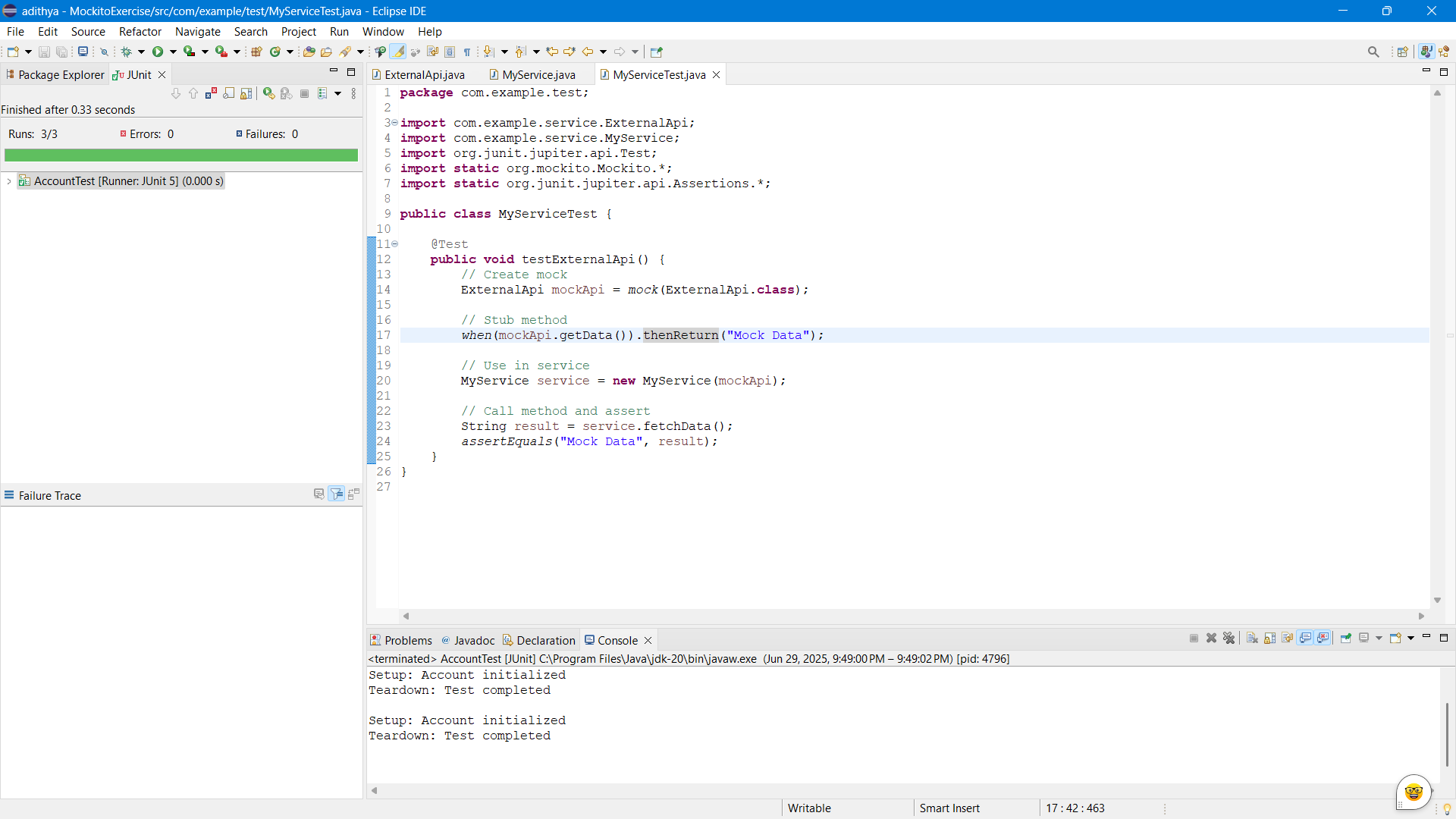
// Call method and assert

String result = service.fetchData();

*assertEquals*("Mock Data", result);

}

}



**Exercise 2: Verifying Interactions :**

**ExternalApi.java :**

**package** com.example.service;

**public** **interface** ExternalApi {

String getData();

}

**MyService.java:**

**package** com.example.service;

**public** **class** MyService {

**private** ExternalApi api;

**public** MyService(ExternalApi api) {

**this**.api = api;

}

**public** String fetchData() {

**return** api.getData();

}

}

**MyServiceTest.java:**

**package** com.example.test;

**import** com.example.service.ExternalApi;

**import** com.example.service.MyService;

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

**import** **static** org.mockito.Mockito.\*;

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

**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**);

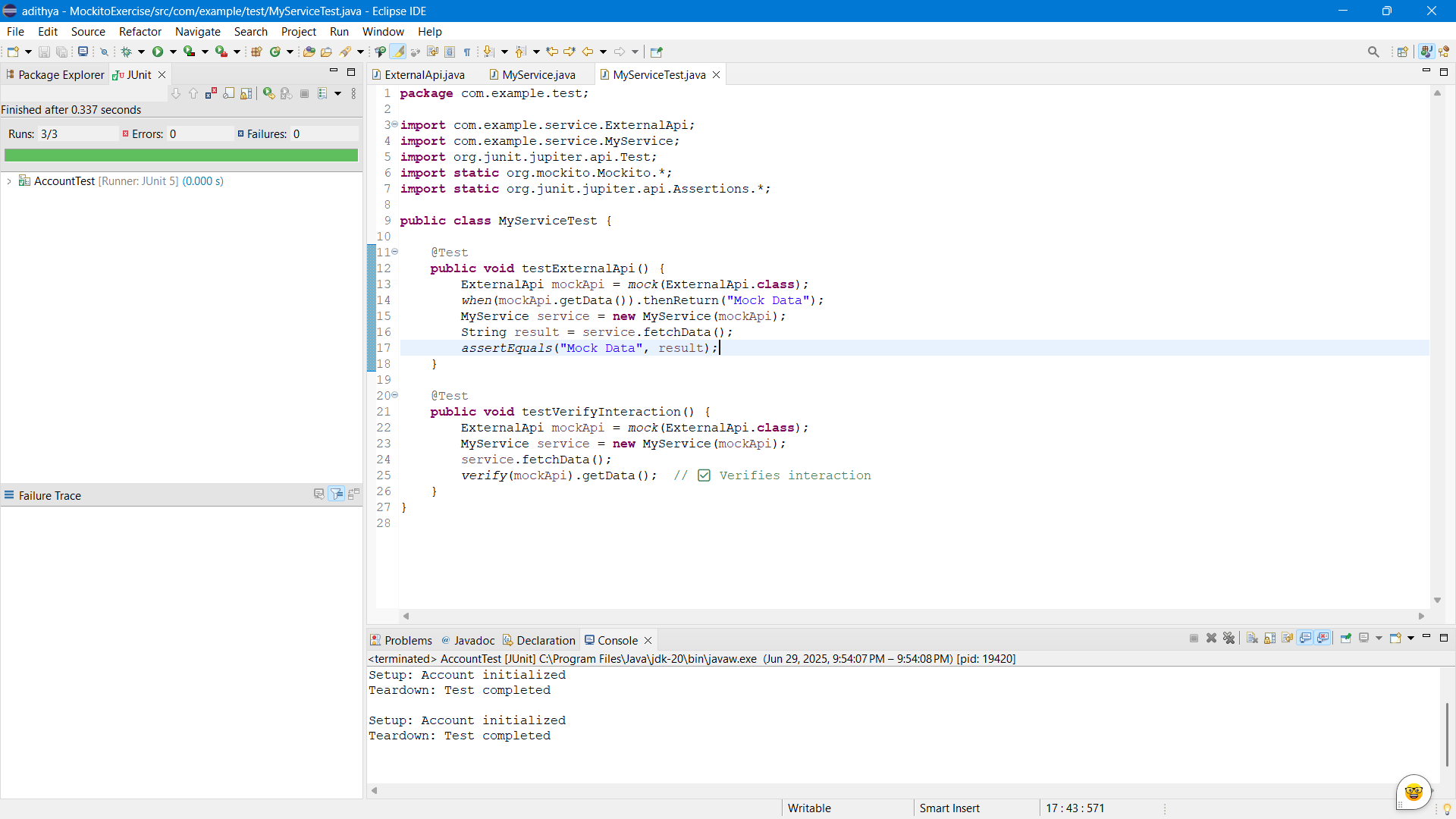
MyService service = **new** MyService(mockApi);

service.fetchData();

*verify*(mockApi).getData(); // ✅ Verifies interaction

}

}



**Exercise 1: Logging Error Messages and Warning Levels**

**LoggingExample.java :**

**package** com.example.logging;

**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***.info("This is an info log");

***logger***.debug("This is a debug log");

***logger***.error("This is an error log");

}

}

