**TASK 1:** **Setting Up Junit**

**THIS IS MY CODE:**

package com.example.test;

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

import org.junit.jupiter.api.Test;

class Calculatortest {

public int add(int a, int b) {

return a + b;

}

@Test

public void testAdd() {

Calculatortest calc = new Calculatortest();

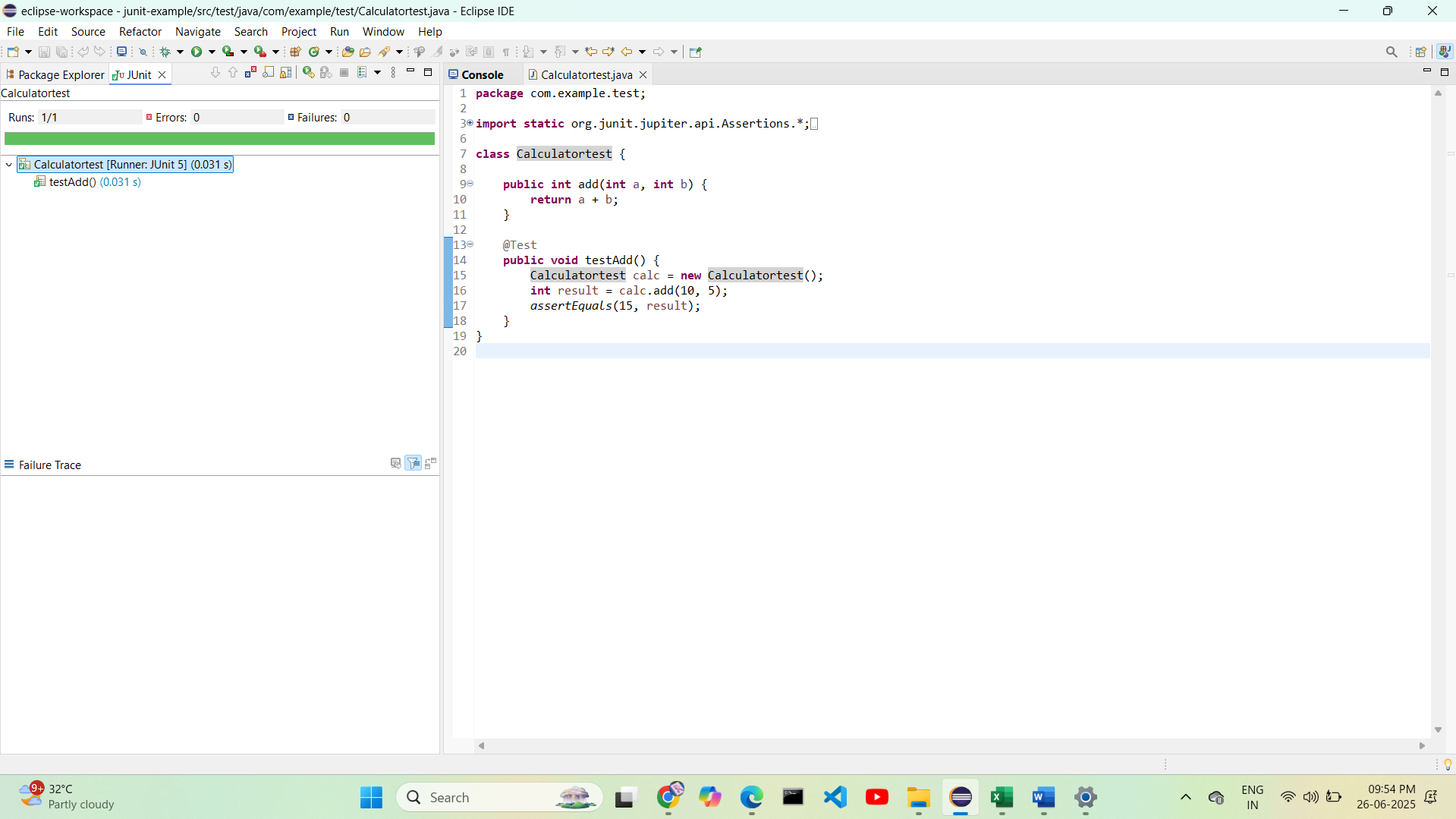
int result = calc.add(10, 5);

assertEquals(15, result);

}

}

**THIS IS MY OUTPUT:**

****

**TASK 2:ASSERTIONS IN JUNIT**

**THIS IS MY CODE:**

package com.example.test;

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);

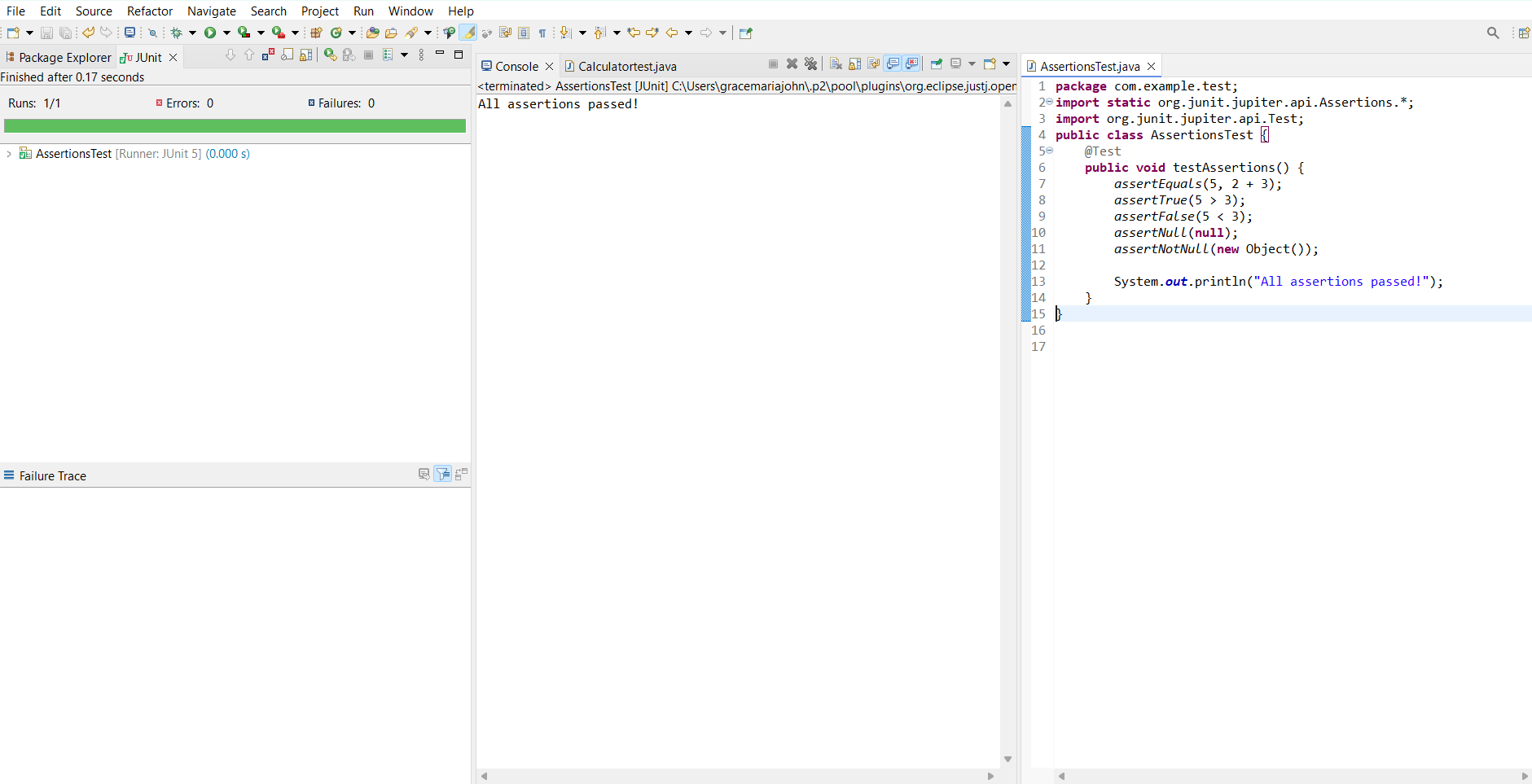
*assertNotNull*(new Object());

System.*out*.println("All assertions passed!");

}

}

**THIS IS MY OUTPUT:**



**TASK 3:** **Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

**THIS IS MY CODE:**

package com.example.test;

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

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

public class CalculatorLifecycleTest {

Calculator calculator;

@BeforeEach

public void setUp() {

calculator = new Calculator();

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

}

@AfterEach

public void tearDown() {

calculator = null;

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

}

@Test

public void testAddition() {

int a = 5, b = 10;

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

*assertEquals*(15, result);

}

@Test

public void testSubtraction() {

int a = 10, b = 3;

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

*assertEquals*(7, result);

}

class Calculator {

public int add(int x, int y) {

return x + y;

}

public int subtract(int x, int y) {

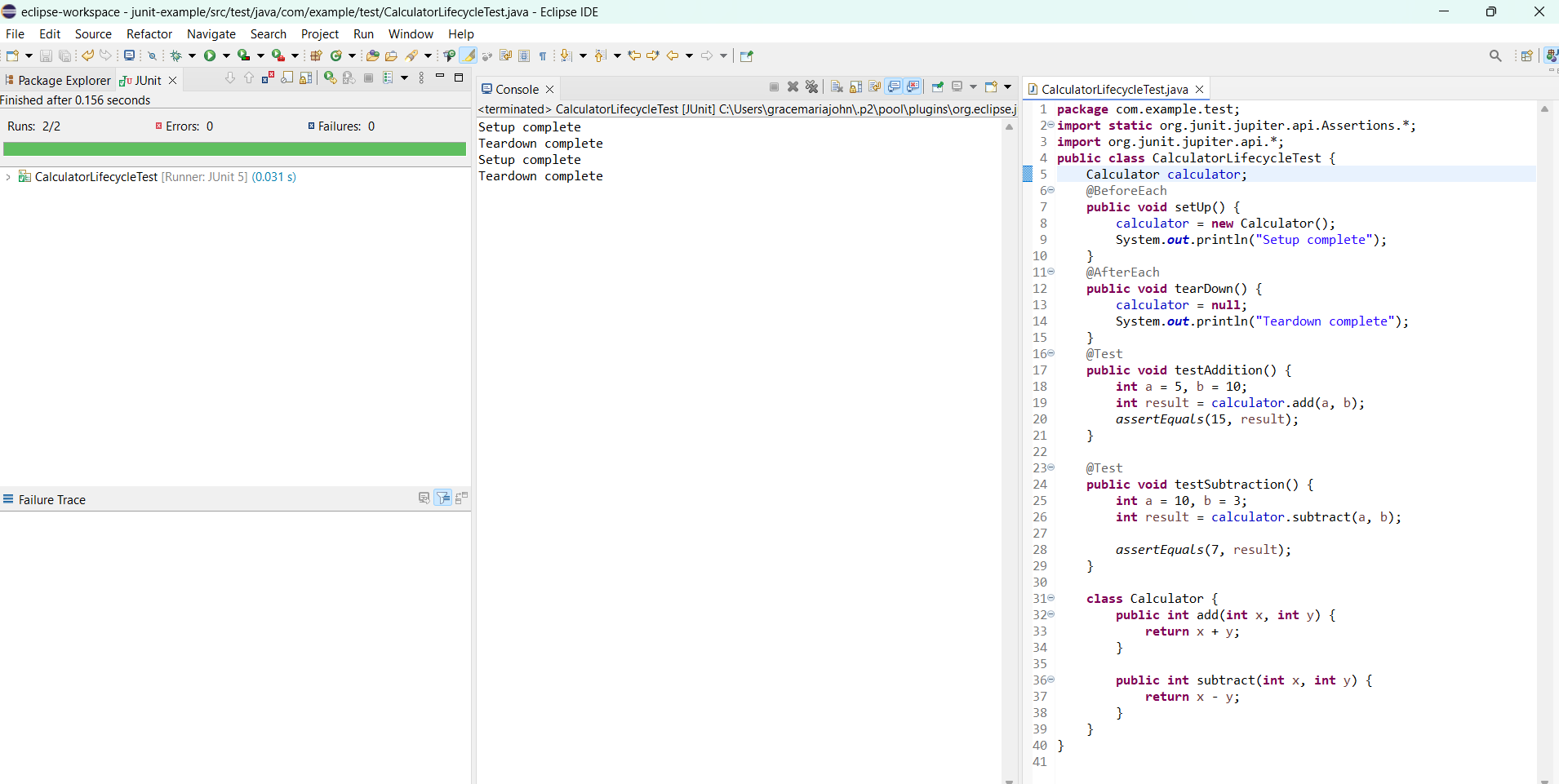
return x - y;

}

}

}

**THIS IS MY OUTPUT:**

****

**TASK 4:** **Mocking and Stubbing**

**THIS IS MY CODE:**

package com.example.test;

public interface ExternalApi {

String getData();

}

package com.example.test;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

package com.example.test;

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

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testExternalApi() {

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

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

MyService service = new MyService(mockApi);

String result = service.fetchData();

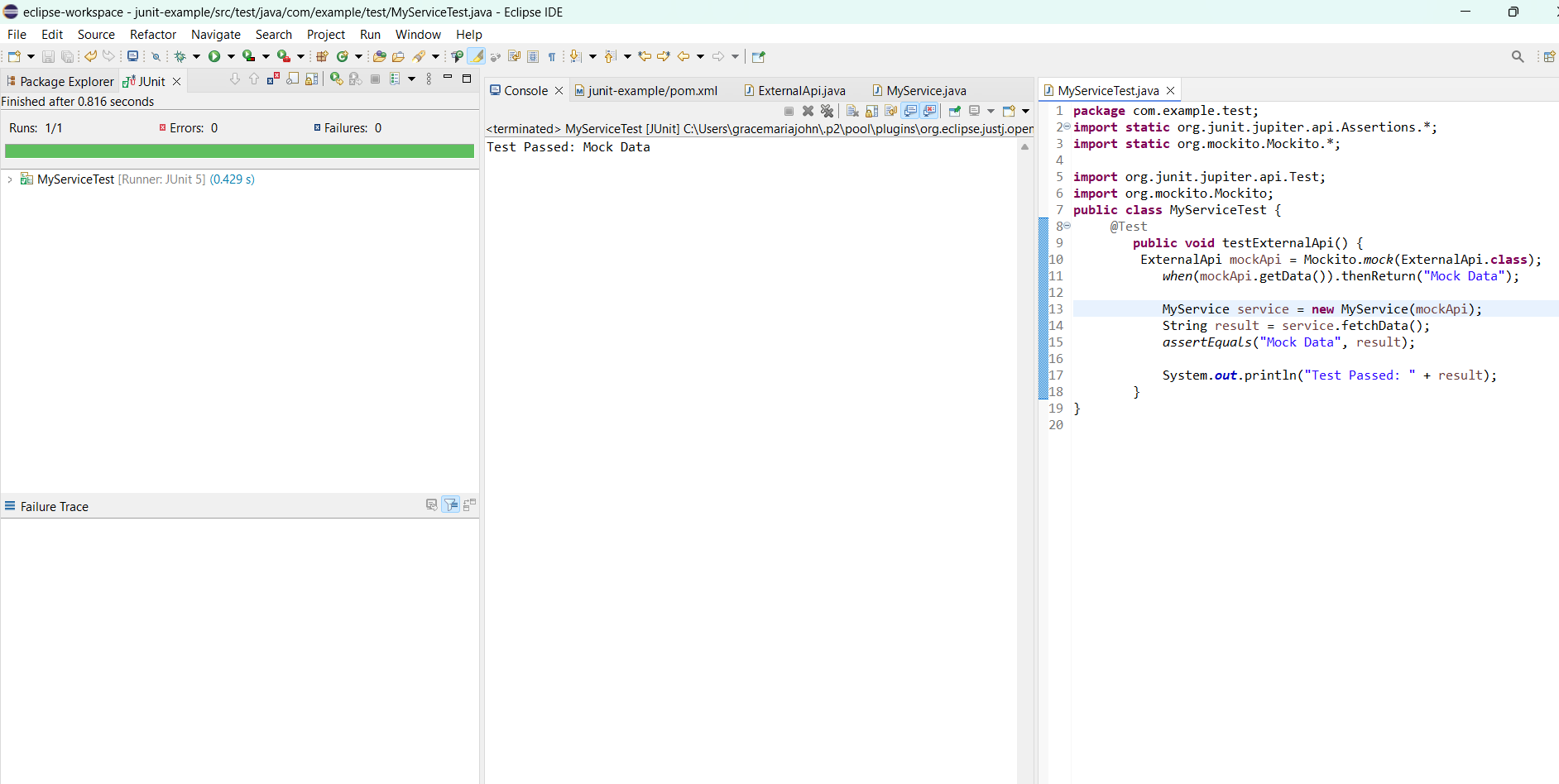
*assertEquals*("Mock Data", result);

System.*out*.println("Test Passed: " + result);

}

}

**THIS IS MY OUTPUT:**

****

**TASK 5:** **Verifying Interactions**

**THIS IS MY CODE:**

package com.example.test;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

public class ArgumentVerificationTest {

interface NotificationService {

void send(String message);

}

@Test

public void testSendCalledWithSpecificMessage() {

NotificationService mockService = *mock*(NotificationService.class);

mockService.send("Hello Alll!");

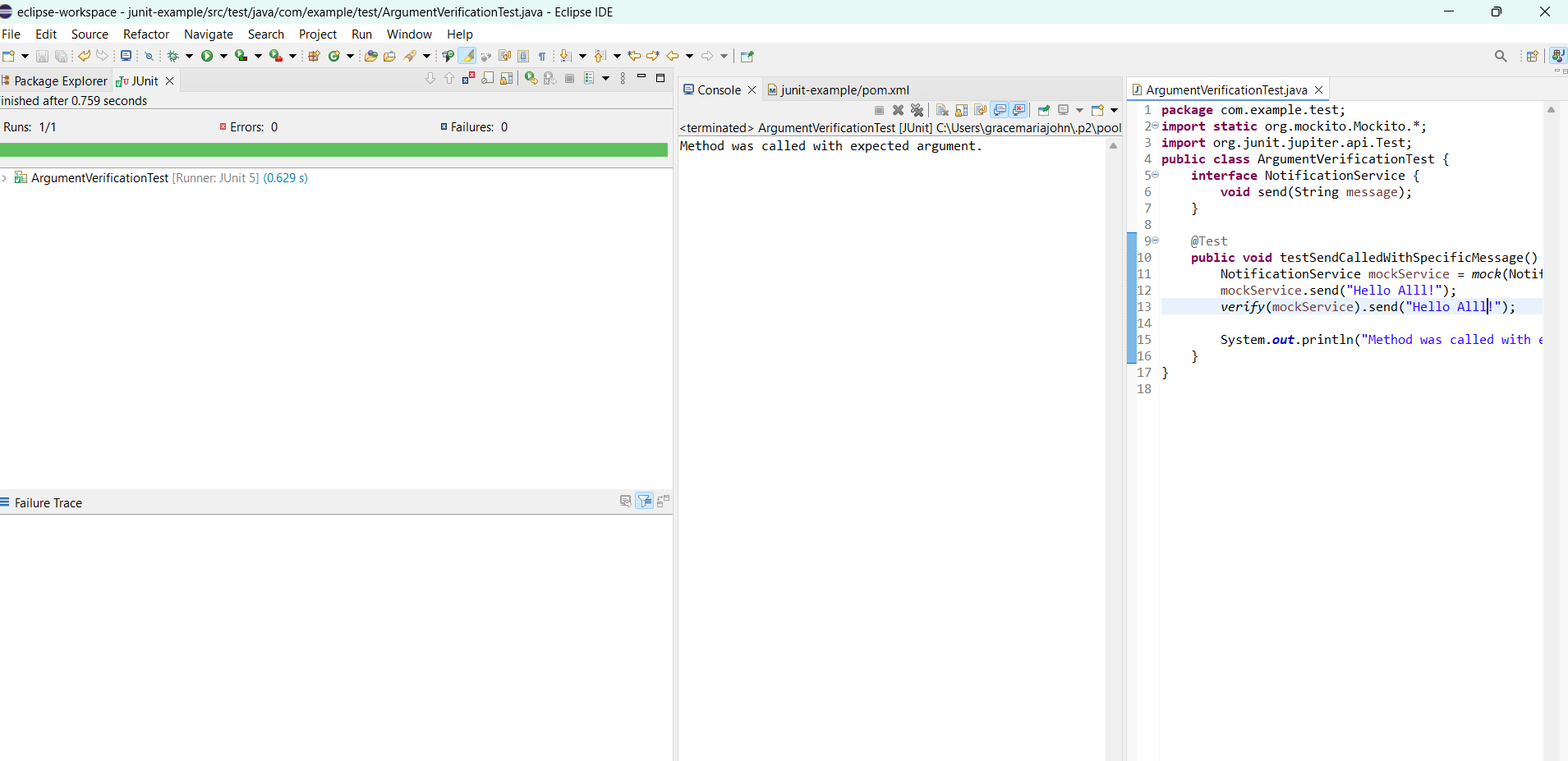
*verify*(mockService).send("Hello Alll!");

System.*out*.println("Method was called with expected argument.");

}

}

**THIS IS MY OUTPUT:**

****