**WEEK:2**

**JUnit Testing Exercises**

**Superset ID: 6419740**

**Exercise 1: Setting Up Junit**

**Code:**

**Pom.xml:**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>JUnitDemo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

<configuration>

<source>11</source>

<target>11</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**Greet.java:**

public class Greet {

public String says() {

return "Hello";

}

}

**GreetTest.java:**

import org.junit.\*;

import static org.junit.Assert.\*;

public class GreetTest {

@Test

public void test() {

Greet greet = new Greet();

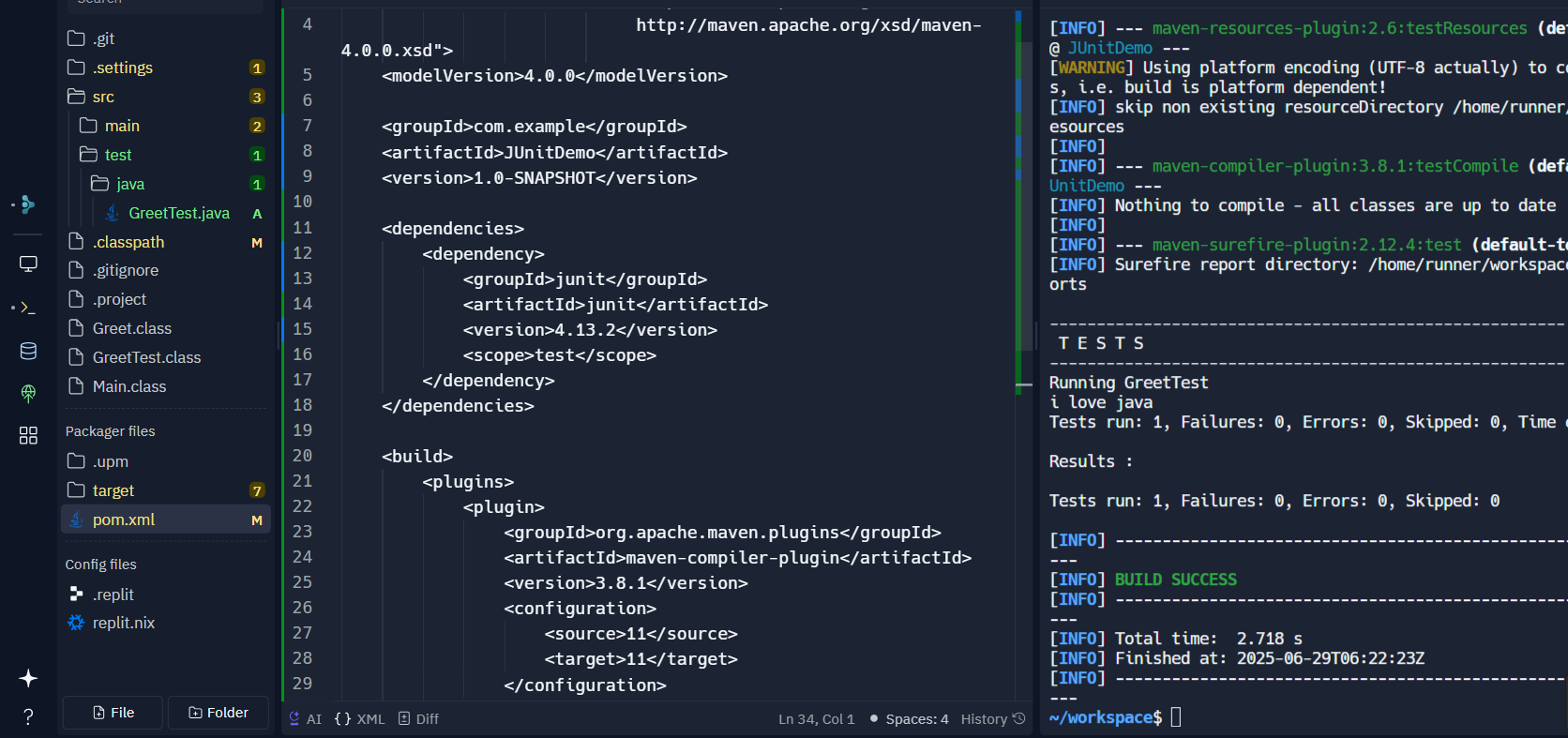
assertEquals("Hello", greet.says());

System.out.println("i love java");

}

}

**Output:**



**Exercise 2: Writing Basic JUnit Tests**

**Code:**

**Calculator.java:**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**CalculatorTest.java:**

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

@Test

public void testAddition() {

Calculator calc = new Calculator();

int result = calc.add(5, 3);

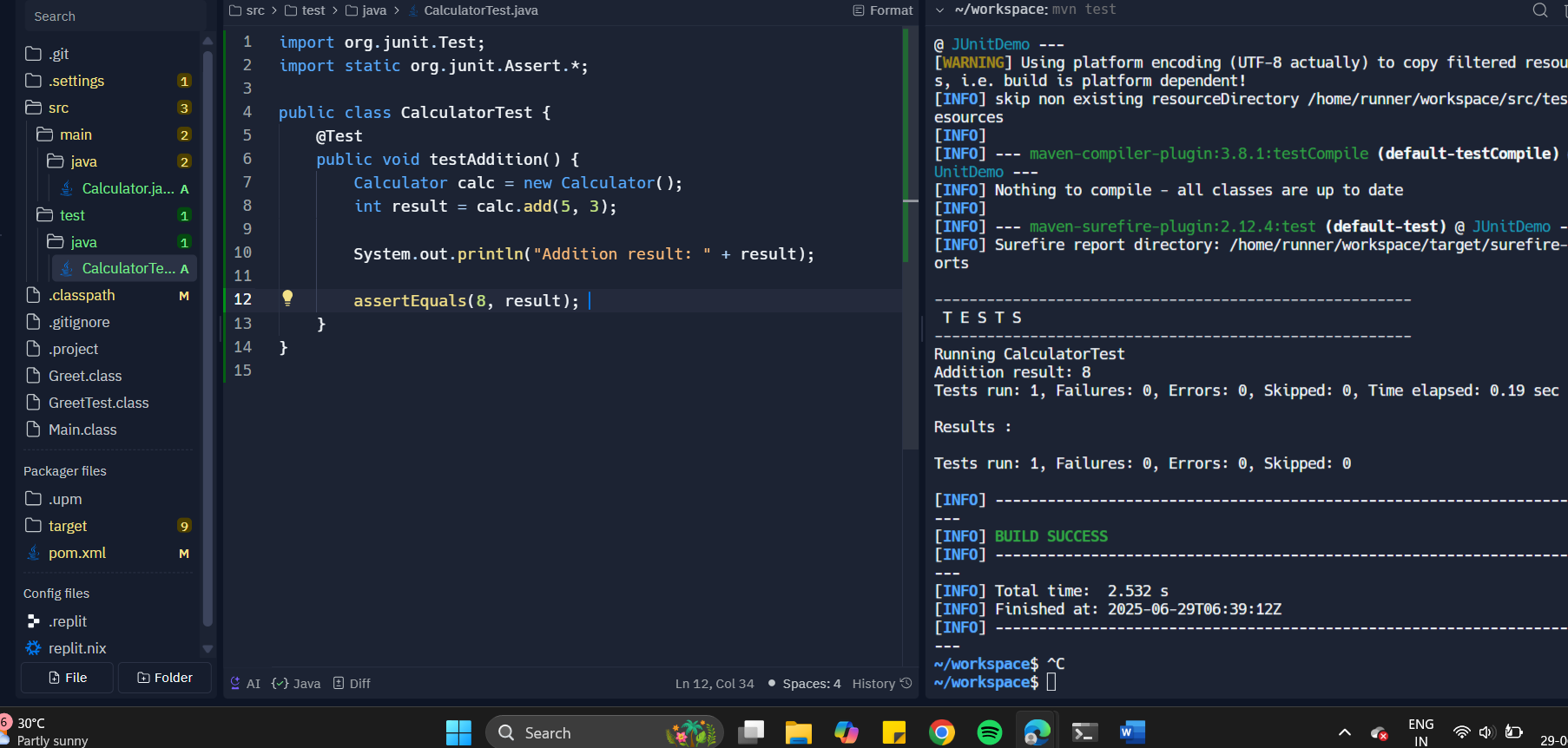
System.out.println("Addition result: " + result);

assertEquals(8, result);

}

}

**Output:**



**Exercise 3: Assertions in JUnit**

**Code:**

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionTest {

@Test

public void testAssertions() {

assertEquals(5, 2 + 3);

assertTrue(5 > 3);

assertFalse(5 < 3);

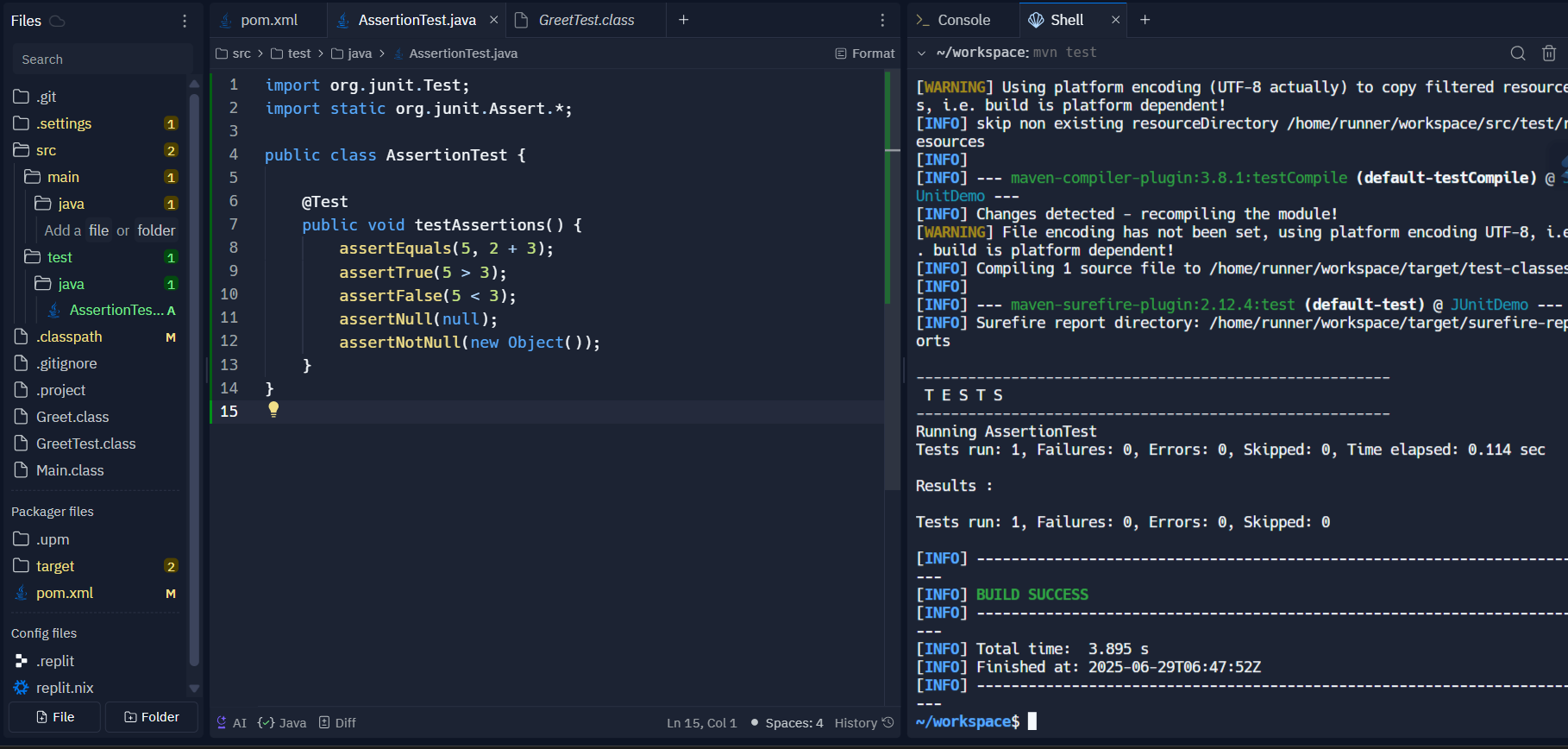
assertNull(null);

assertNotNull(new Object());

}

}

**Output:**



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

**Code:**

**UserService.java:**

public class UserService {

public boolean isValidUser(String username, String password) {

return "admin".equals(username) && "admin123".equals(password);

}

}

**UserServiceTest.java:**

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class UserServiceTest {

UserService service;

@Before

public void setUp() {

service = new UserService();

}

@After

public void tearDown() {

service = null;

}

@Test

public void testIsValidUser() {

boolean result = service.isValidUser("admin", "admin123");

assertTrue(result);

}

@Test

public void testIsInvalidUser() {

boolean result = service.isValidUser("admin", "wrongpass");

assertFalse(result);

}

}

**Output:**

