

Junit Basic Testing Exercises

Exercise 1:SETTING UP JUNIT:

Scenario:

You need to set up JUnit in your Java project to start writing unit tests.

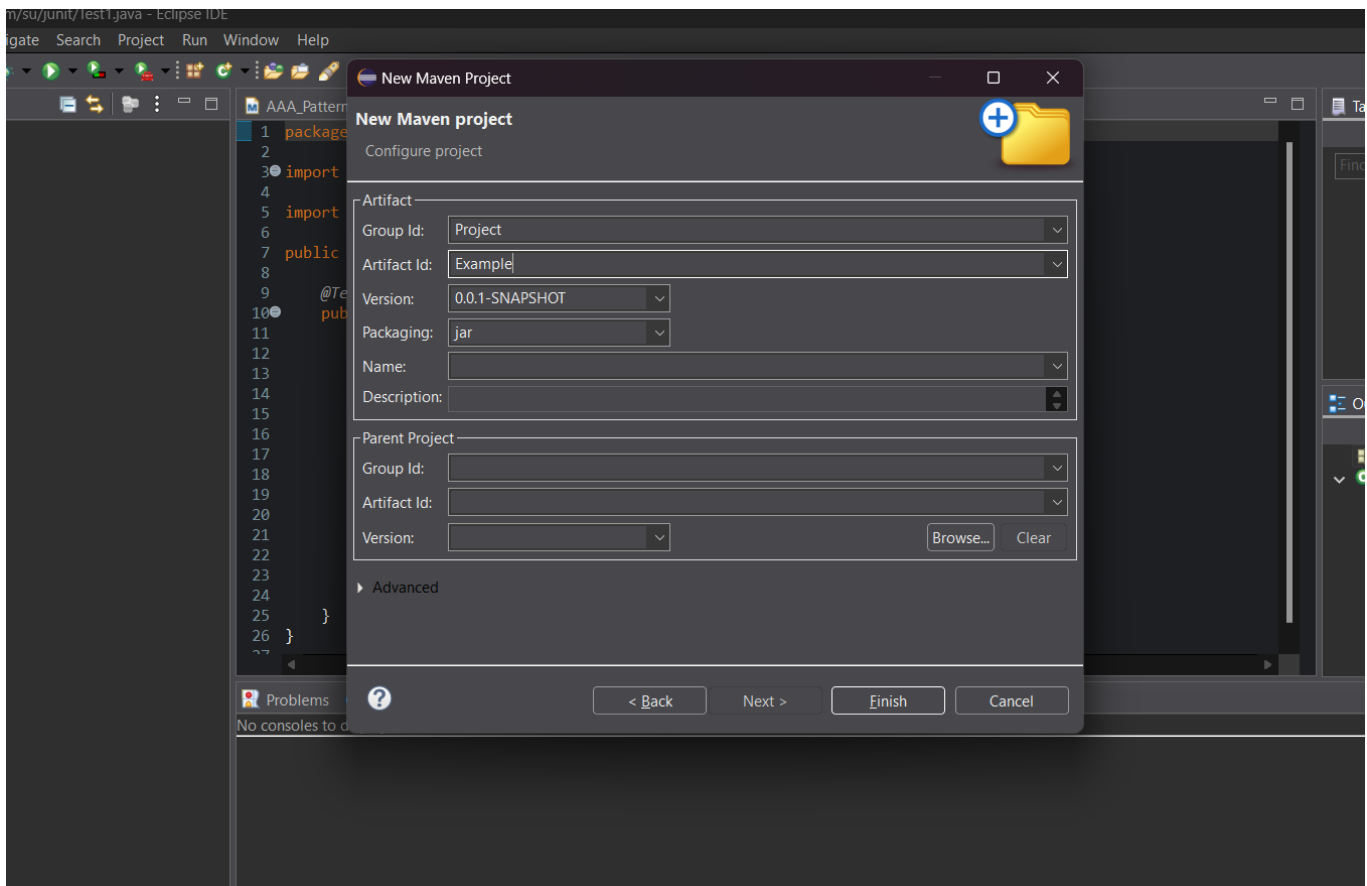
SOLUTION:

Step 1. Create or Open a **Maven Project** :

You can use an IDE like **IntelliJ IDEA**, **Eclipse**, or set it up manually.

Step 2. Click File-> New-> Maven Project(Click on create a simple project)

Then write the name of group id and artifact id and then click on finish button.



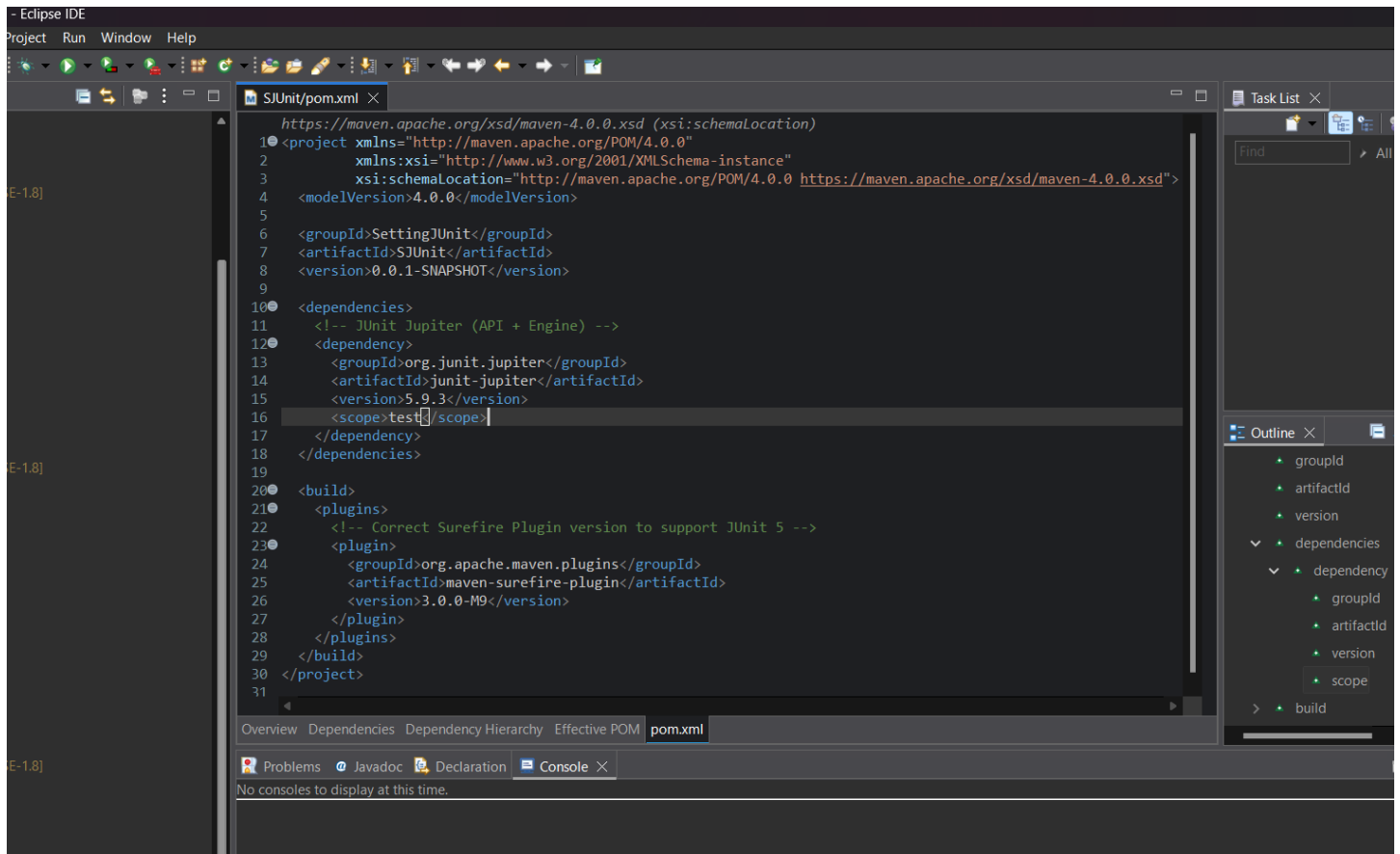
Step 3. Add **Junit Dependency** to your project . Add following to your **pom.xml** file:

```
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 https://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>Project</groupId>
  <artifactId>Example</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <dependencies>
    <!-- JUnit Jupiter (API + Engine) -->
    <dependency>
      <groupId>org.junit.jupiter</groupId>
      <artifactId>junit-jupiter</artifactId>
      <version>5.9.3</version>
      <scope>test</scope>
    </dependency>
  </dependencies>

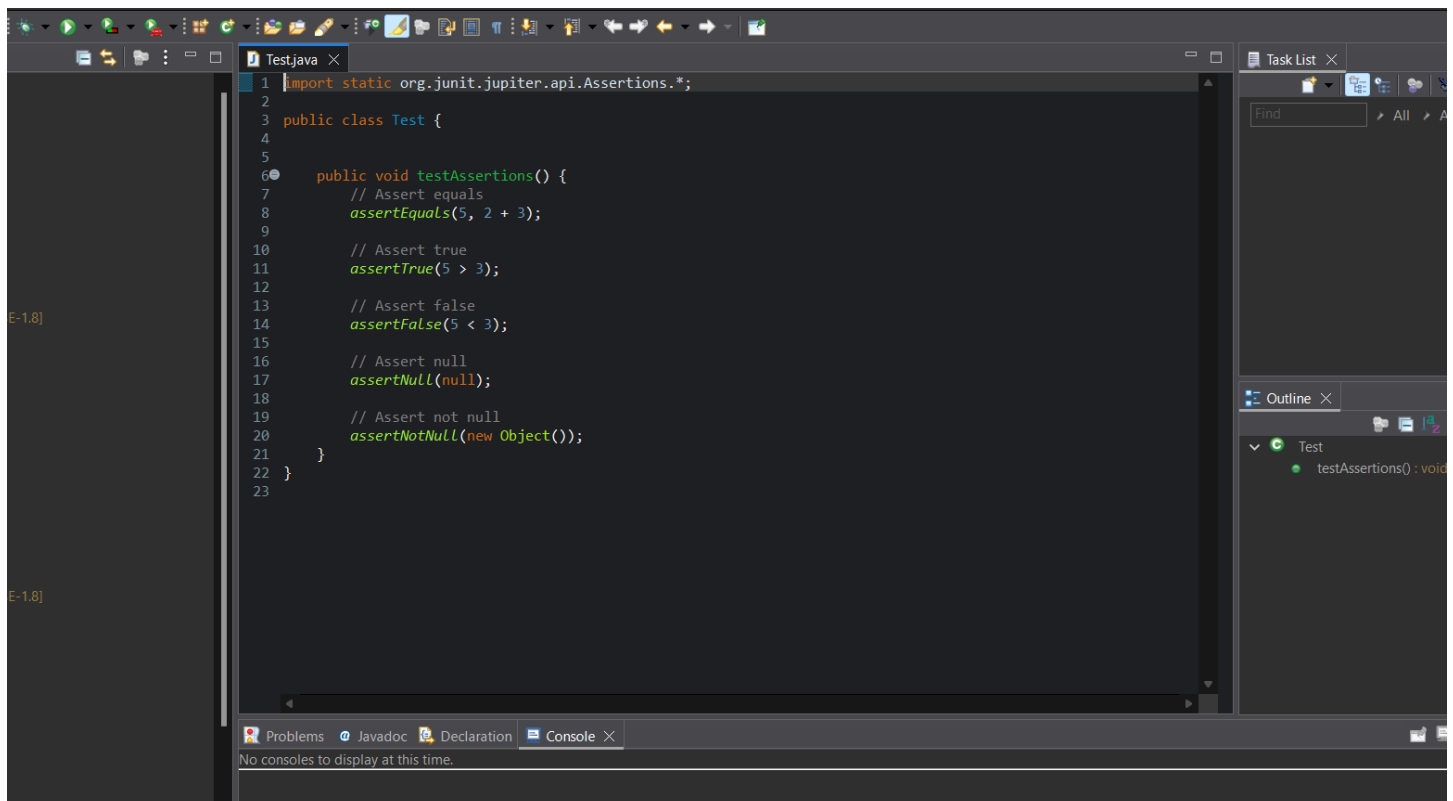
  <build>
    <plugins>
      <!-- Correct Surefire Plugin version to support JUnit 5 -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>3.0.0-M9</version>
      </plugin>
    </plugins>
  </build>
</project>
```

Then run this by maven test.



Step 4. Create your test classes in **src/test/java**.

Write a sample test case:



CODE:

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

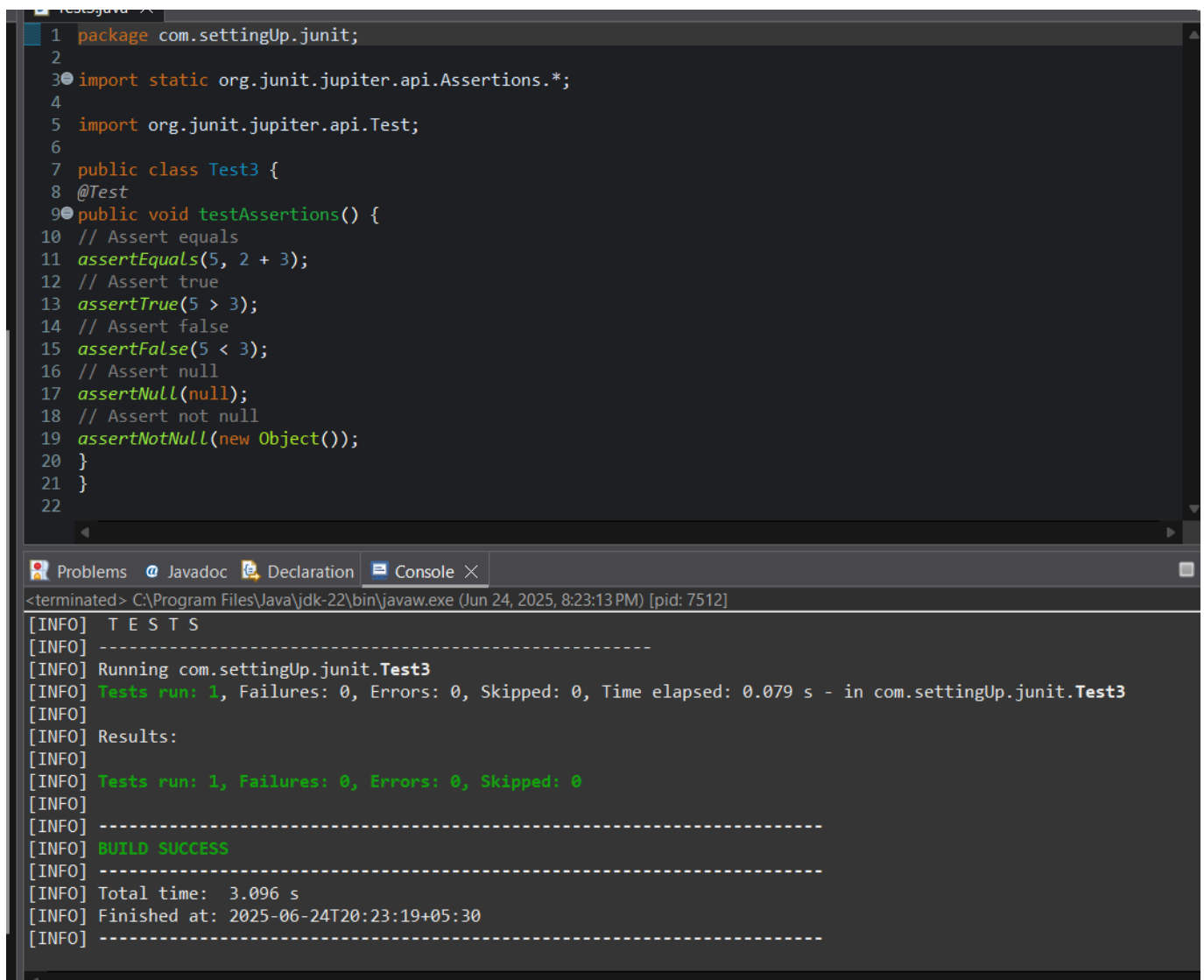
public class Test {

    public void testAssertions() {
        assertEquals(5, 2 + 3);
        assertTrue(5 > 3);
        assertFalse(5 < 3);
        assertNull(null);
        assertNotNull(new Object());
    }
}
```

Step 5. Then run the test class in **maven test** and then **Junit test** .

OUTPUT:

Maven Test:

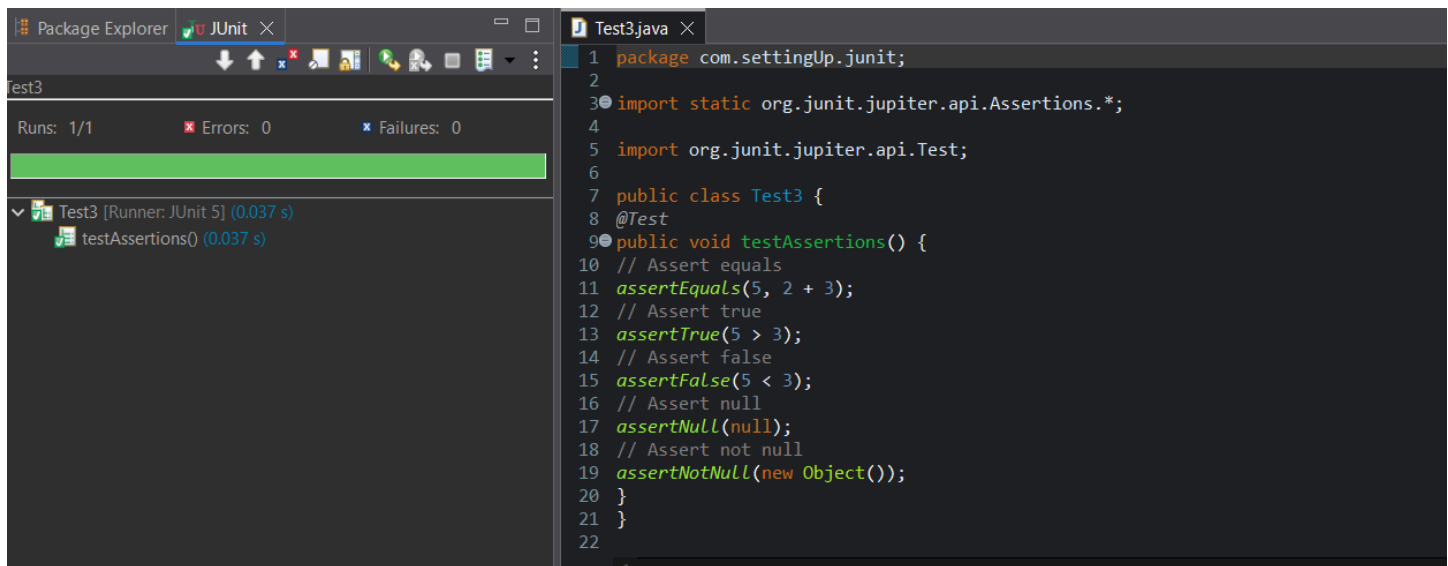


The screenshot shows an IDE with a Java file named `Test3.java` and its corresponding Maven test output in the console. The code defines a `Test3` class with a `@Test` annotated `testAssertions()` method. The console output shows the test running successfully, with 1 test passed, 0 failures, 0 errors, and 0 skipped tests. The build is successful.

```
1 package com.settingUp.junit;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 import org.junit.jupiter.api.Test;
6
7 public class Test3 {
8     @Test
9     public void testAssertions() {
10         // Assert equals
11         assertEquals(5, 2 + 3);
12         // Assert true
13         assertTrue(5 > 3);
14         // Assert false
15         assertFalse(5 < 3);
16         // Assert null
17         assertNull(null);
18         // Assert not null
19         assertNotNull(new Object());
20     }
21 }
22
```

```
<terminated> C:\Program Files\Java\jdk-22\bin\javaw.exe (Jun 24, 2025, 8:23:13 PM) [pid: 7512]
[INFO] T E S T S
[INFO] -----
[INFO] Running com.settingUp.junit.Test3
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.079 s - in com.settingUp.junit.Test3
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.096 s
[INFO] Finished at: 2025-06-24T20:23:19+05:30
[INFO] -----
```

JUnit Test:



Exercise 3: Assertions in Junit

Scenario:

You need to use different assertions in JUnit to validate your test results

SOLUTION:

Step 1: Add the Junit Dependency in pom.xml

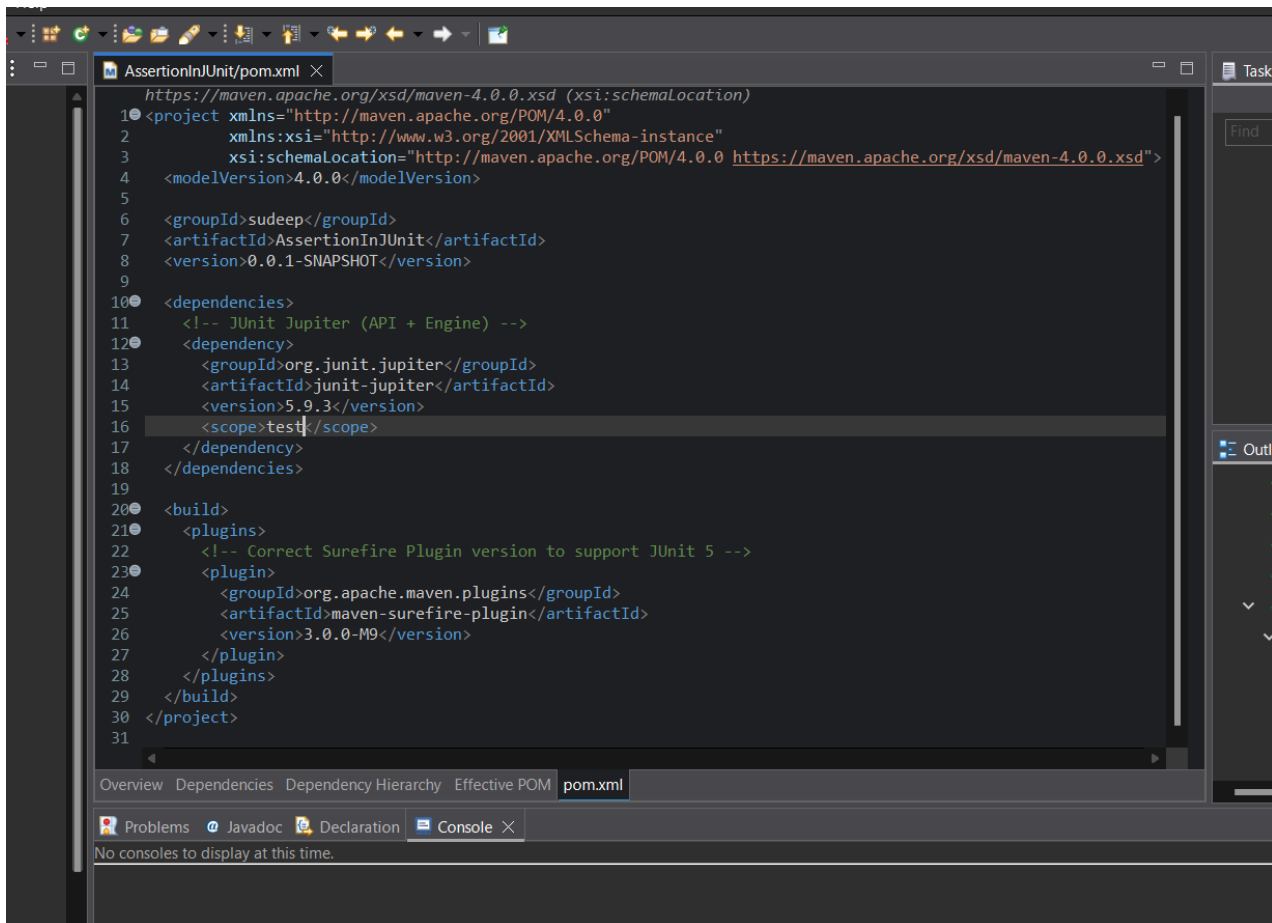
pom.xml :

```
<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">  
    <modelVersion>4.0.0</modelVersion>  
    <groupId>sudeep</groupId>  
    <artifactId>AssertionInJUnit</artifactId>  
    <version>0.0.1-SNAPSHOT</version>  
    <dependencies>  
        <!-- JUnit Jupiter (API + Engine) -->  
        <dependency>  
            <groupId>org.junit.jupiter</groupId>  
            <artifactId>junit-jupiter</artifactId>  
            <version>5.9.3</version>  
            <scope>test</scope>  
        </dependency>  
    </dependencies>  
    <build>  
        <plugins>
```

```

<!-- Correct Surefire Plugin version to support JUnit 5 -->
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>3.0.0-M9</version>
</plugin>
</plugins>
</build>
</project>

```



Step 2: Create your test classes in **src/test/java**.

Write the required test case and run as maven test and Junit test:

Code of Assertion.java:

```

package com.AssertionInJUnit.junit;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;
public class Assertion {
  @Test
  public void testAssertions() {
    // Assert equals

```

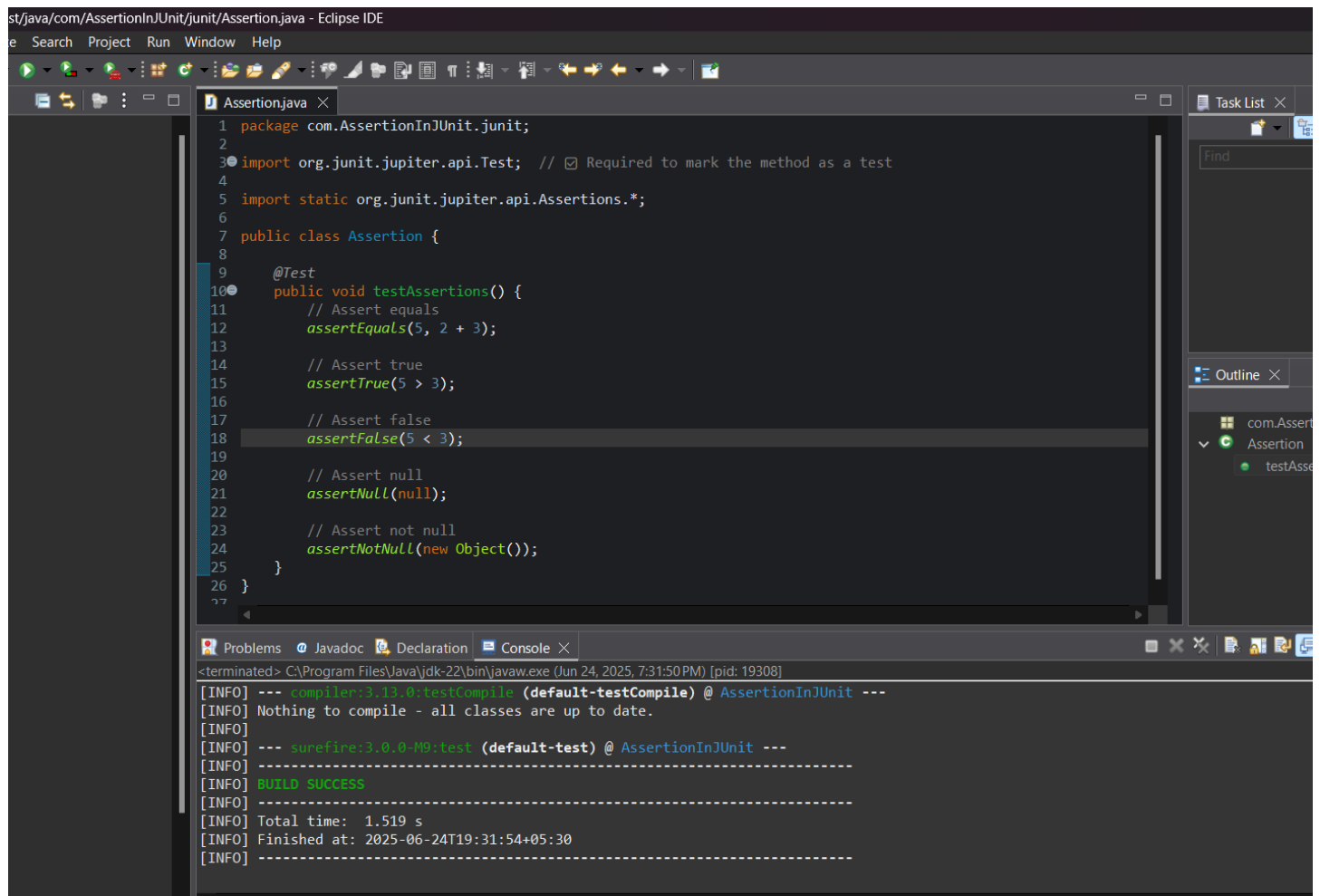
```

    assertEquals(5, 2 + 3);
    // Assert true
    assertTrue(5 > 3);
    // Assert false
    assertFalse(5 < 3);
    // Assert null
    assertNull(null);
    // Assert not null
    assertNotNull(new Object());
}
}

```

OUTPUT:

Maven test:



The screenshot shows the Eclipse IDE with the file `st/java/com/AssertionInJUnit/junit/Assertion.java` open. The code in the file is as follows:

```

1 package com.AssertionInJUnit.junit;
2
3 import org.junit.jupiter.api.Test; // Required to mark the method as a test
4
5 import static org.junit.jupiter.api.Assertions.*;
6
7 public class Assertion {
8
9     @Test
10    public void testAssertions() {
11        // Assert equals
12        assertEquals(5, 2 + 3);
13
14        // Assert true
15        assertTrue(5 > 3);
16
17        // Assert false
18        assertFalse(5 < 3);
19
20        // Assert null
21        assertNull(null);
22
23        // Assert not null
24        assertNotNull(new Object());
25    }
26 }
27

```

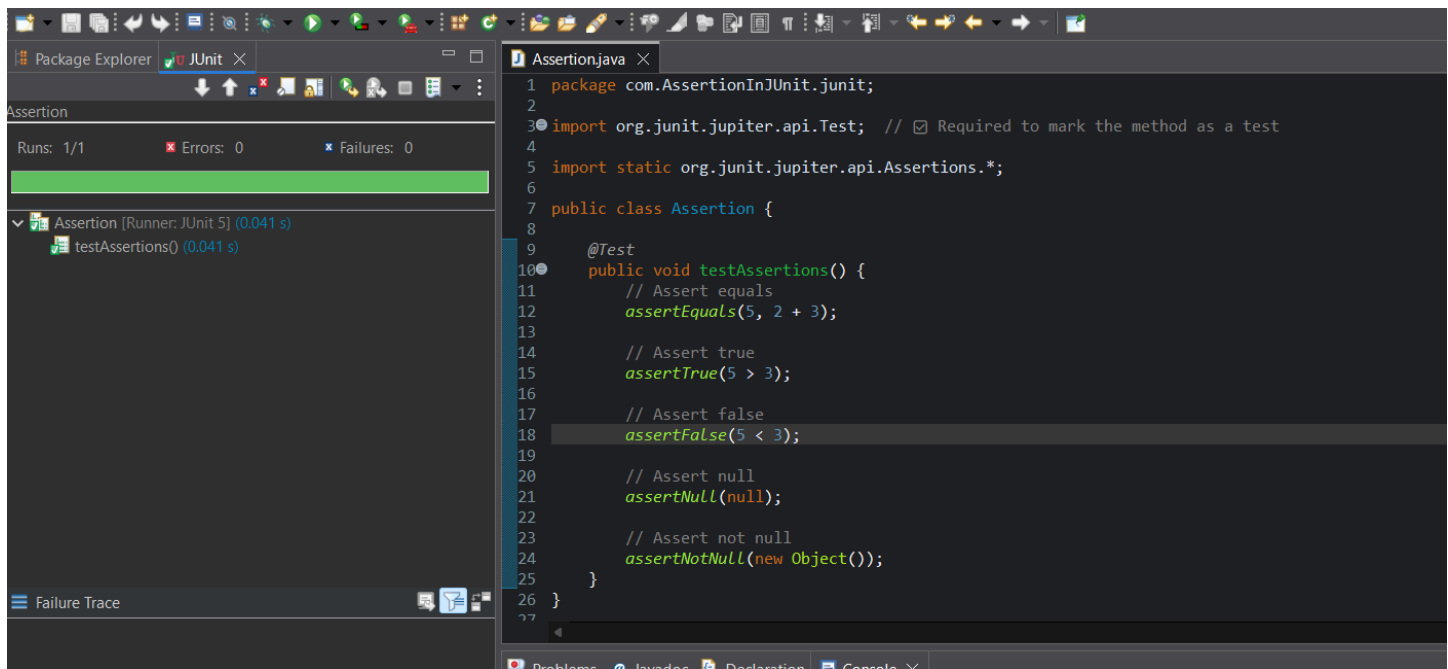
The console output at the bottom shows the results of the Maven test:

```

<terminated> C:\Program Files\Java\jdk-22\bin\javaw.exe (Jun 24, 2025, 7:31:50 PM) [pid: 19308]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ AssertionInJUnit ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- surefire:3.0.0-M9:test (default-test) @ AssertionInJUnit ---
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO]
[INFO] Total time: 1.519 s
[INFO] Finished at: 2025-06-24T19:31:54+05:30
[INFO] -----

```

JUnit test:



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

Scenario: You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup and teardown methods.

SOLUTION:

Step 1: Add the JUnit Dependency in pom.xml

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 https://maven.apache.org/xsd/maven-4.0.0.xsd>  
    <modelVersion>4.0.0</modelVersion>  
    <groupId>AAA</groupId>  
    <artifactId>AAA_Pattern</artifactId>  
    <version>0.0.1-SNAPSHOT</version>  
    <dependencies>  
        <!-- JUnit Jupiter (API + Engine) -->  
        <dependency>  
            <groupId>org.junit.jupiter</groupId>  
            <artifactId>junit-jupiter</artifactId>  
            <version>5.9.3</version>
```



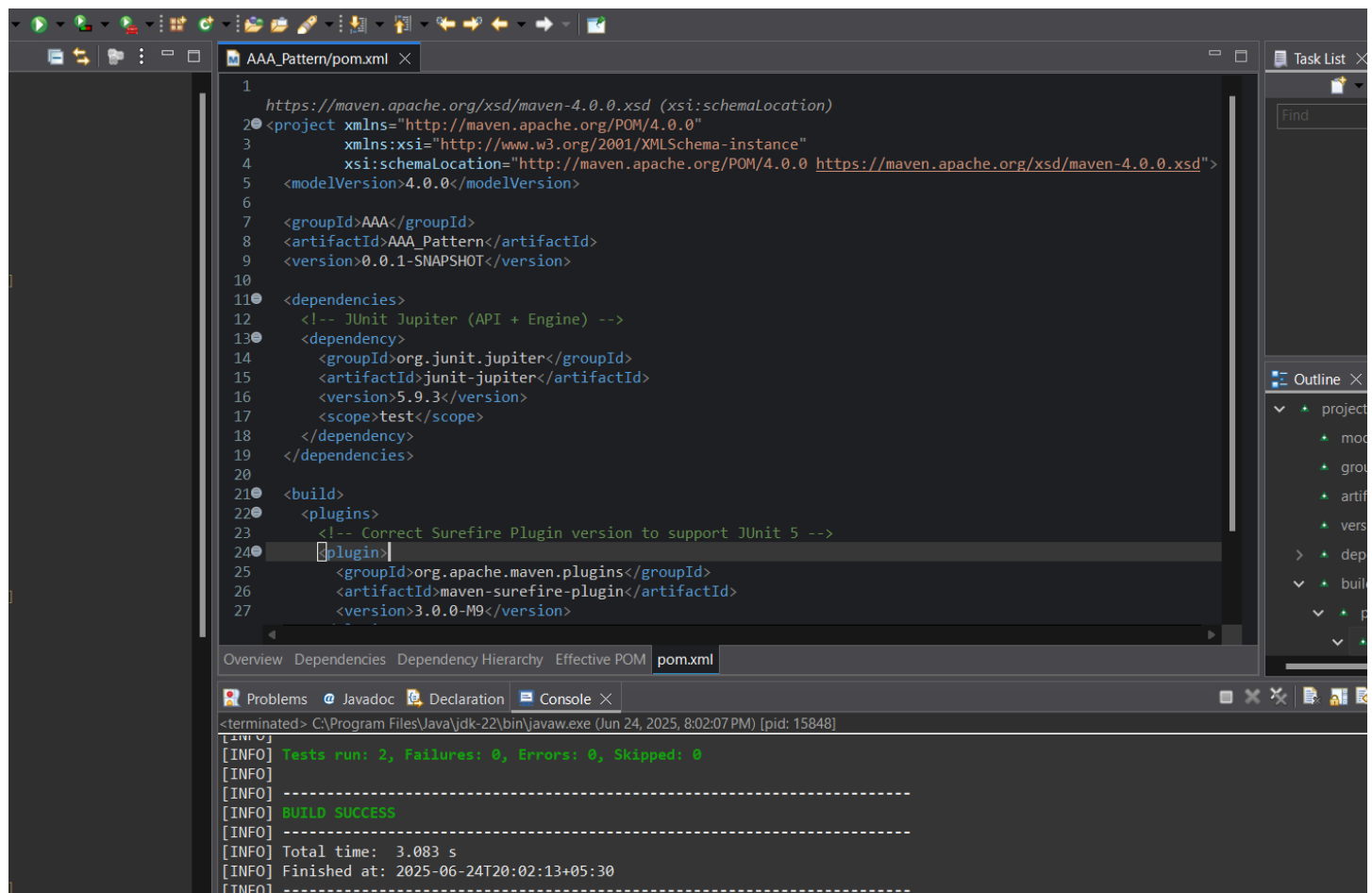
```

    <scope>test</scope>
  </dependency>
</dependencies>

<build>
  <plugins>
    <!-- Correct Surefire Plugin version to support JUnit 5 -->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>3.0.0-M9</version>
    </plugin>
  </plugins>
</build>
</project>

```

OUTPUT:



The screenshot shows an IDE with a Maven project. The main editor displays the `AAA_Pattern/pom.xml` file, which contains the XML configuration for the project. The configuration includes dependencies on JUnit Jupiter and the Maven Surefire plugin. The console at the bottom shows the output of a Maven build, indicating that the tests were run successfully.

```

1  https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
5    <modelVersion>4.0.0</modelVersion>
6
7    <groupId>AAA</groupId>
8    <artifactId>AAA_Pattern</artifactId>
9    <version>0.0.1-SNAPSHOT</version>
10
11    <dependencies>
12      <!-- JUnit Jupiter (API + Engine) -->
13      <dependency>
14        <groupId>org.junit.jupiter</groupId>
15        <artifactId>junit-jupiter</artifactId>
16        <version>5.9.3</version>
17        <scope>test</scope>
18      </dependency>
19    </dependencies>
20
21    <build>
22      <plugins>
23        <!-- Correct Surefire Plugin version to support JUnit 5 -->
24        <plugin>
25          <groupId>org.apache.maven.plugins</groupId>
26          <artifactId>maven-surefire-plugin</artifactId>
27          <version>3.0.0-M9</version>

```

```

[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO] BUILD SUCCESS
[INFO] Total time: 3.083 s
[INFO] Finished at: 2025-06-24T20:02:13+05:30
[INFO]

```

Step 2: Create your test classes in **src/test/java**.

Write the required test case and run as maven test and Junit test:

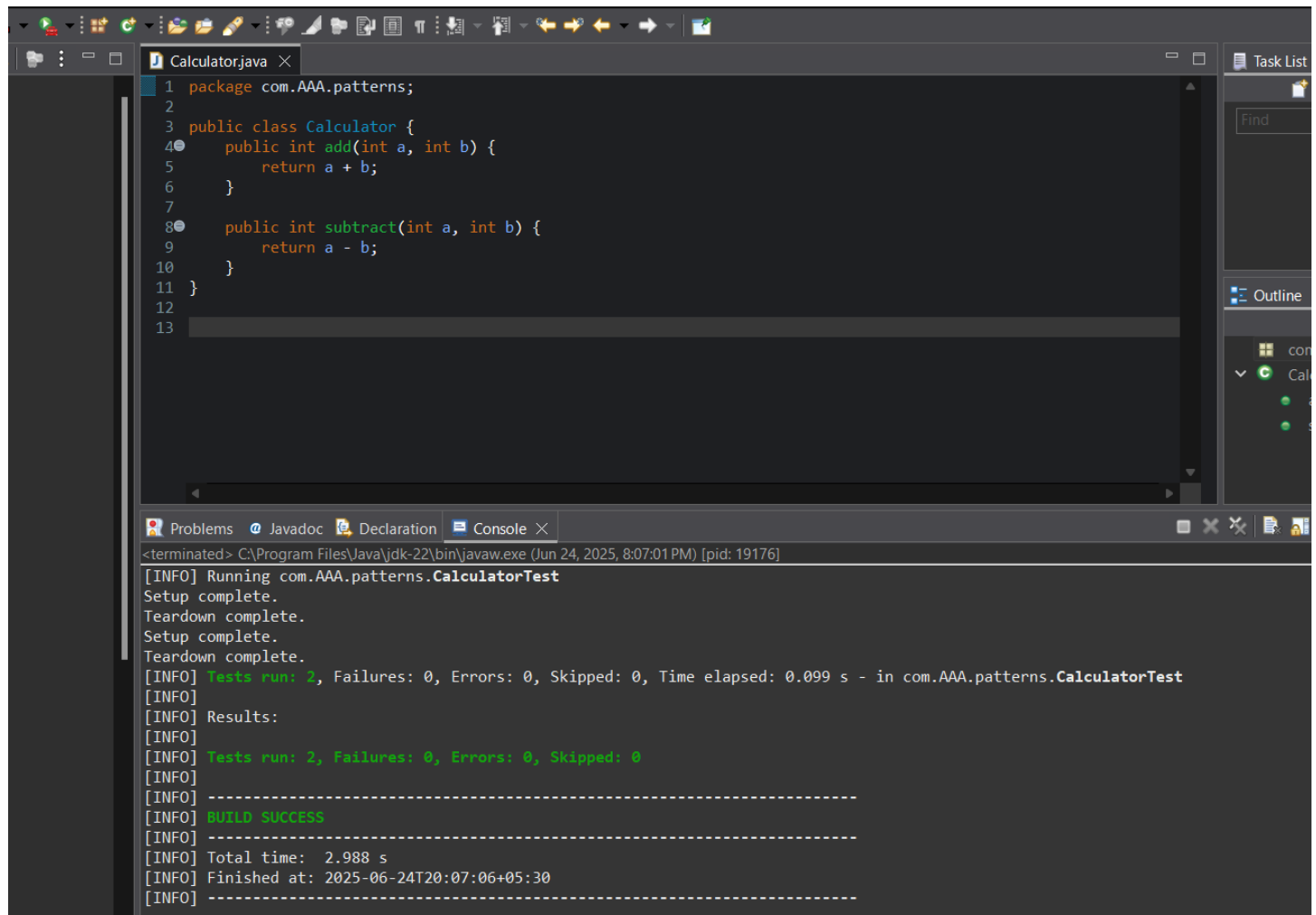
CODE for Calculator.java:

```
package com.AAA.patterns;

public class Calculator {
    public int add(int a, int b) {
        return a + b;
    }
    public int subtract(int a, int b) {
        return a - b;
    }
}
```

OUTPUT:

Maven Test for Calculator.java:

The screenshot shows an IDE window with a file named 'Calculator.java' open. The code in the editor is the same as provided in the previous block. Below the editor, the 'Console' tab is active, displaying the output of a Maven test run. The output shows that the tests passed successfully, with a 'BUILD SUCCESS' message. The console text is as follows:

```
<terminated> C:\Program Files\Java\jdk-22\bin\javaw.exe (Jun 24, 2025, 8:07:01 PM) [pid: 19176]
[INFO] Running com.AAA.patterns.CalculatorTest
Setup complete.
Teardown complete.
Setup complete.
Teardown complete.
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.099 s - in com.AAA.patterns.CalculatorTest
[INFO] Results:
[INFO]
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.988 s
[INFO] Finished at: 2025-06-24T20:07:06+05:30
[INFO] -----
```

CODE for CalculatorTest.java:

```
package com.AAA.patterns;

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

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

public class CalculatorTest {
    private Calculator calculator;

    @BeforeEach
    void setUp() {
        calculator = new Calculator();
        System.out.println("Setup complete.");
    }

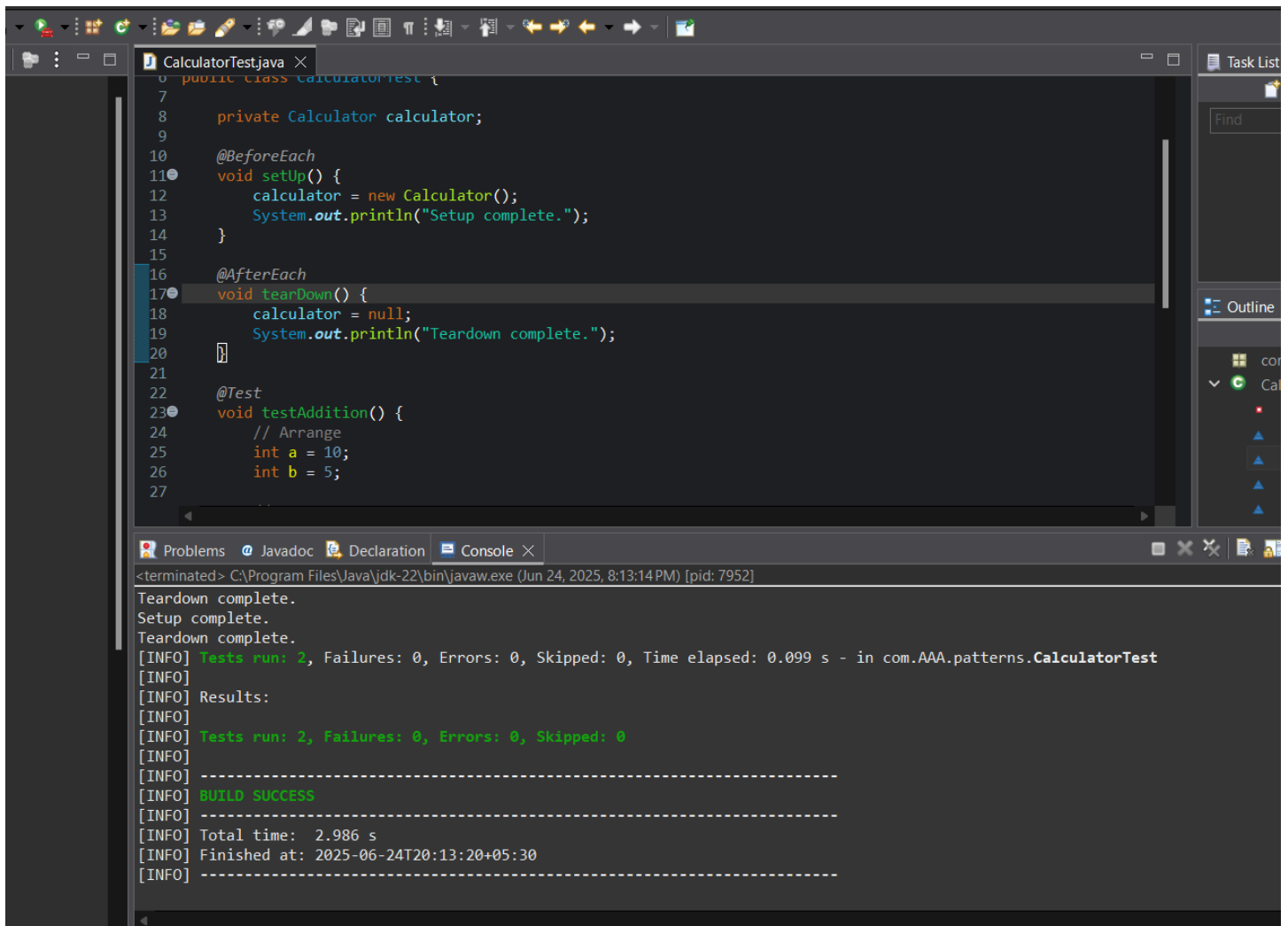
    @AfterEach
    void tearDown() {
        calculator = null;
        System.out.println("Teardown complete.");
    }

    @Test
    void testAddition() {
        // Arrange
        int a = 10;
        int b = 5;
        // Act
        int result = calculator.add(a, b);
        // Assert
        assertEquals(15, result);
    }

    @Test
    void testSubtraction() {
        // Arrange
        int a = 10;
        int b = 3;
        // Act
        int result = calculator.subtract(a, b);
        // Assert
        assertEquals(7, result);
    }
}
```

OUTPUT:

Maven Test for CalculatorTest.java:



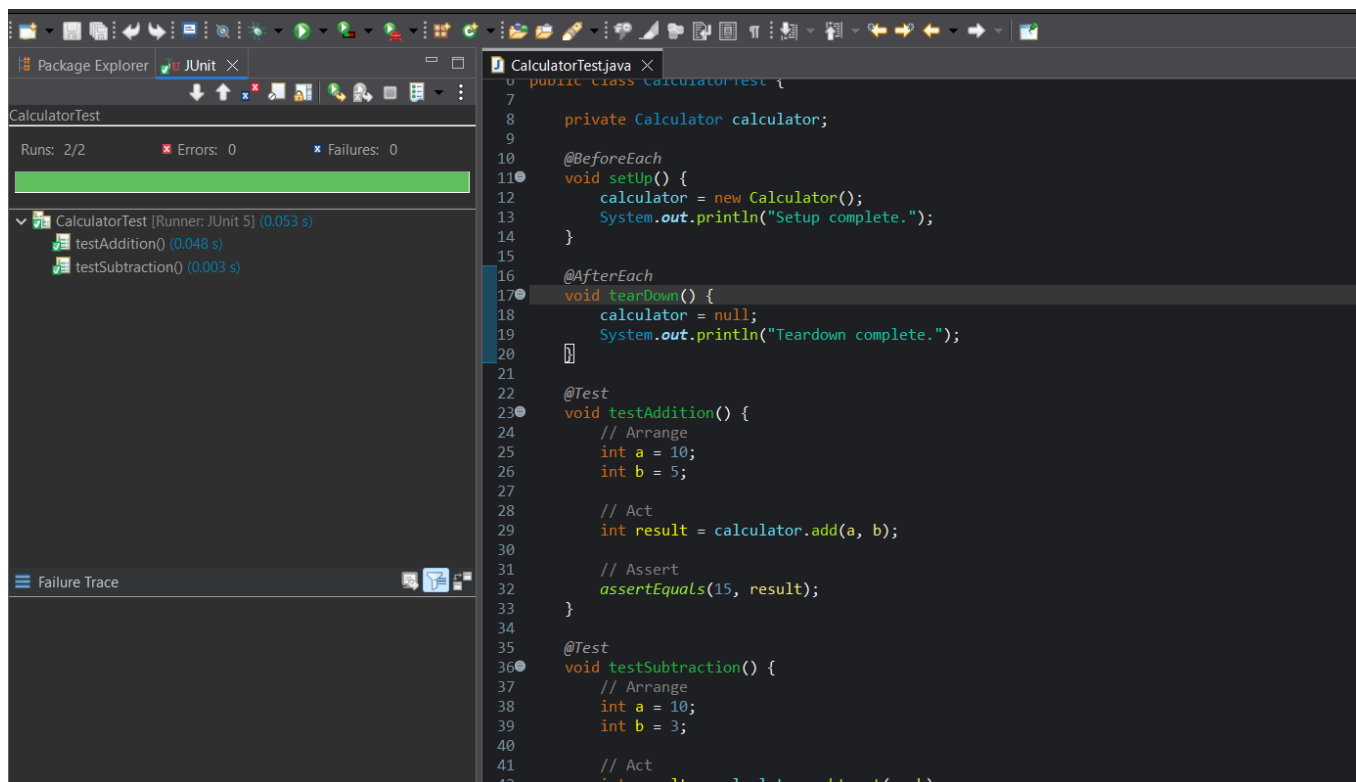
The screenshot displays an IDE window with the file `CalculatorTest.java` open. The code is as follows:

```
1 public class CalculatorTest {
2
3     private Calculator calculator;
4
5     @BeforeEach
6     void setUp() {
7         calculator = new Calculator();
8         System.out.println("Setup complete.");
9     }
10
11     @AfterEach
12     void tearDown() {
13         calculator = null;
14         System.out.println("Teardown complete.");
15     }
16
17     @Test
18     void testAddition() {
19         // Arrange
20         int a = 10;
21         int b = 5;
22     }
23 }
```

The console output at the bottom shows the results of the Maven test execution:

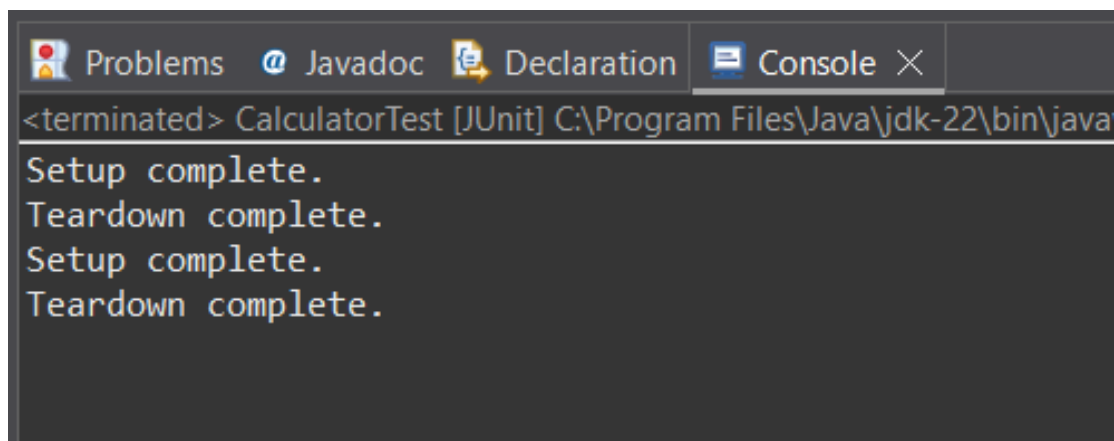
```
<terminated> C:\Program Files\Java\jdk-22\bin\javaw.exe (Jun 24, 2025, 8:13:14 PM) [pid: 7952]
Teardown complete.
Setup complete.
Teardown complete.
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.099 s - in com.AAA.patterns.CalculatorTest
[INFO] Results:
[INFO]
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.986 s
[INFO] Finished at: 2025-06-24T20:13:20+05:30
[INFO] -----
```

JUnit Test for CalculatorTest.java:



The screenshot shows an IDE with the JUnit test results on the left and the source code for `CalculatorTest.java` on the right. The test results show two tests passed: `testAddition()` (0.048 s) and `testSubtraction()` (0.003 s). The source code is as follows:

```
1 public class CalculatorTest {
2
3     private Calculator calculator;
4
5     @BeforeEach
6     void setUp() {
7         calculator = new Calculator();
8         System.out.println("Setup complete.");
9     }
10
11     @AfterEach
12     void tearDown() {
13         calculator = null;
14         System.out.println("Teardown complete.");
15     }
16
17     @Test
18     void testAddition() {
19         // Arrange
20         int a = 10;
21         int b = 5;
22
23         // Act
24         int result = calculator.add(a, b);
25
26         // Assert
27         assertEquals(15, result);
28     }
29
30     @Test
31     void testSubtraction() {
32         // Arrange
33         int a = 10;
34         int b = 3;
35
36         // Act
37         int result = calculator.subtract(a, b);
38     }
39 }
```



The screenshot shows the IDE's Console window with the following output:

```
<terminated> CalculatorTest [JUnit] C:\Program Files\Java\jdk-22\bin\java
Setup complete.
Teardown complete.
Setup complete.
Teardown complete.
```

NAME- G Kullayi Reddy

SUPERSET ID-6371952