Week 2- 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. Steps: 1. Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse). 2. Add JUnit dependency to your project. If you are using Maven, add the following to your pom.xml: junit junit 4.13.2 test 3. Create a new test class in your project.

Calculator.java

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

    public int subtract(int a, int b) {
        return a - b;
    }

    public int multiply(int a, int b) {
        return a * b;
    }

    public double divide(int a, int b) {
        if (b == 0) {
            throw new IllegalArgumentException("Division by zero is not allowed.");
        }
        return (double) a / b;
    }
}
```

CalculatorTest.java

```
import static org.junit.Assert.assertEquals;
import org.junit.Test;

public class CalculatorTest {

    @Test
    public void testAdd() {
        Calculator calc = new Calculator();
        assertEquals(5, calc.add(2, 3));
    }

    @Test
    public void testSubtract() {
        Calculator calc = new Calculator();
        assertEquals(1, calc.subtract(3, 2));
    }

    @Test
    public void testMultiply() {
        Calculator calc = new Calculator();
        assertEquals(6, calc.multiply(2, 3));
    }

    @Test(expected = IllegalArgumentException.class)
    public void testDivideByZero() {
        Calculator calc = new Calculator();
        calc.divide(5, 0);
    }
```

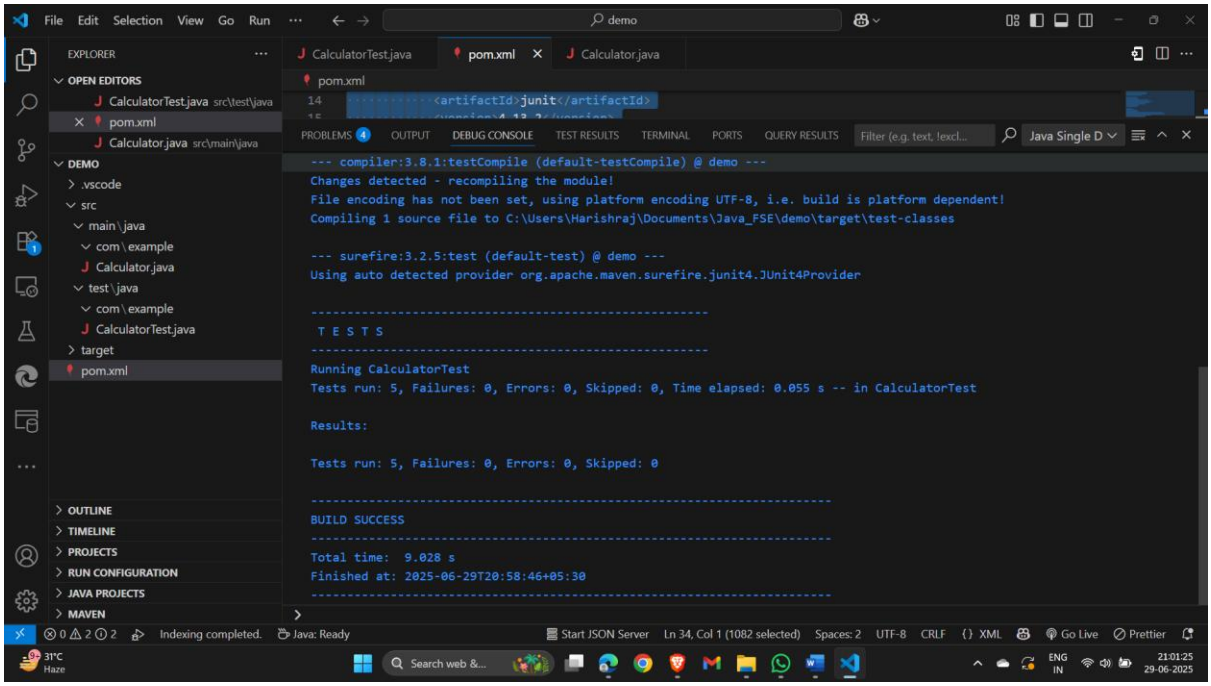
```
    @Test
    public void testDivide() {
        Calculator calc = new Calculator();
        assertEquals(2.0, calc.divide(6, 3), 0.001);
    }
}
```

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>demo</artifactId>
    <version>1.0</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>17</source>
                <target>17</target>
            </configuration>
        </plugin>
    </plugins>
  </build>
</project>
```

Output:



Exercise 3: Assertions in JUnit

Scenario:

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

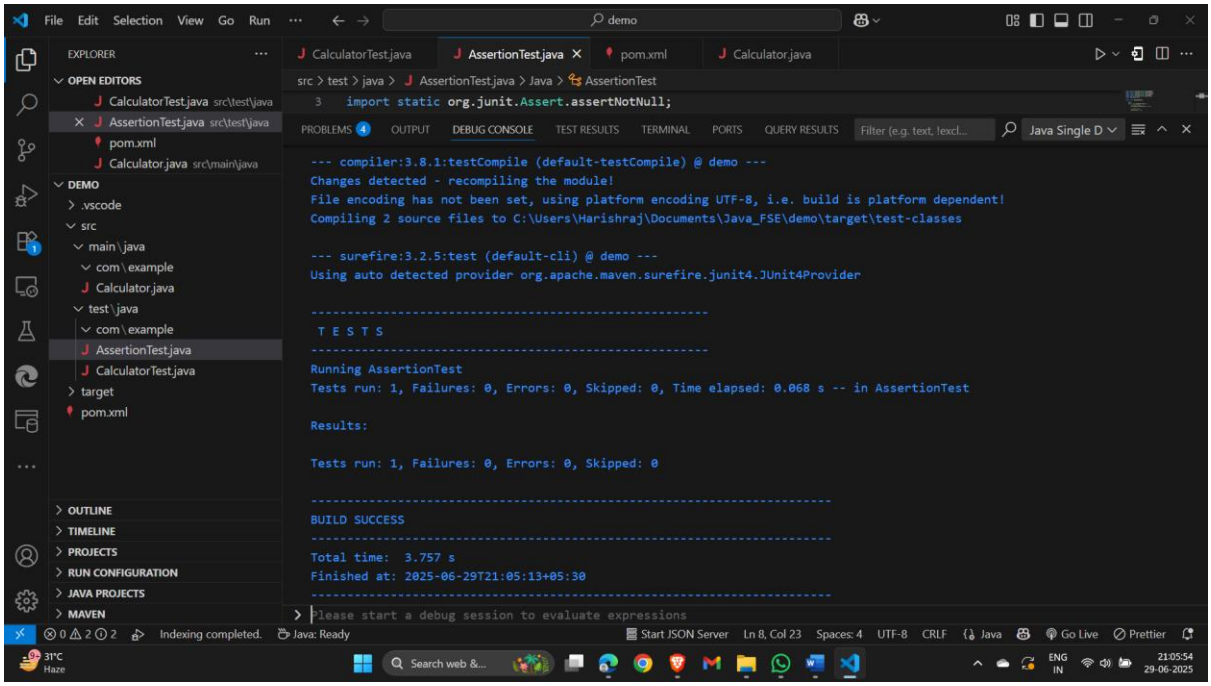
Steps:

1. Write tests using various JUnit assertions.

Solution Code:

```
public class AssertionsTest {
    @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:



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.

Steps:

1. Write tests using the AAA pattern.
2. Use @Before and @After annotations for setup and teardown methods.

Calculator.java

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

    public int subtract(int a, int b) {
        return a - b;
```

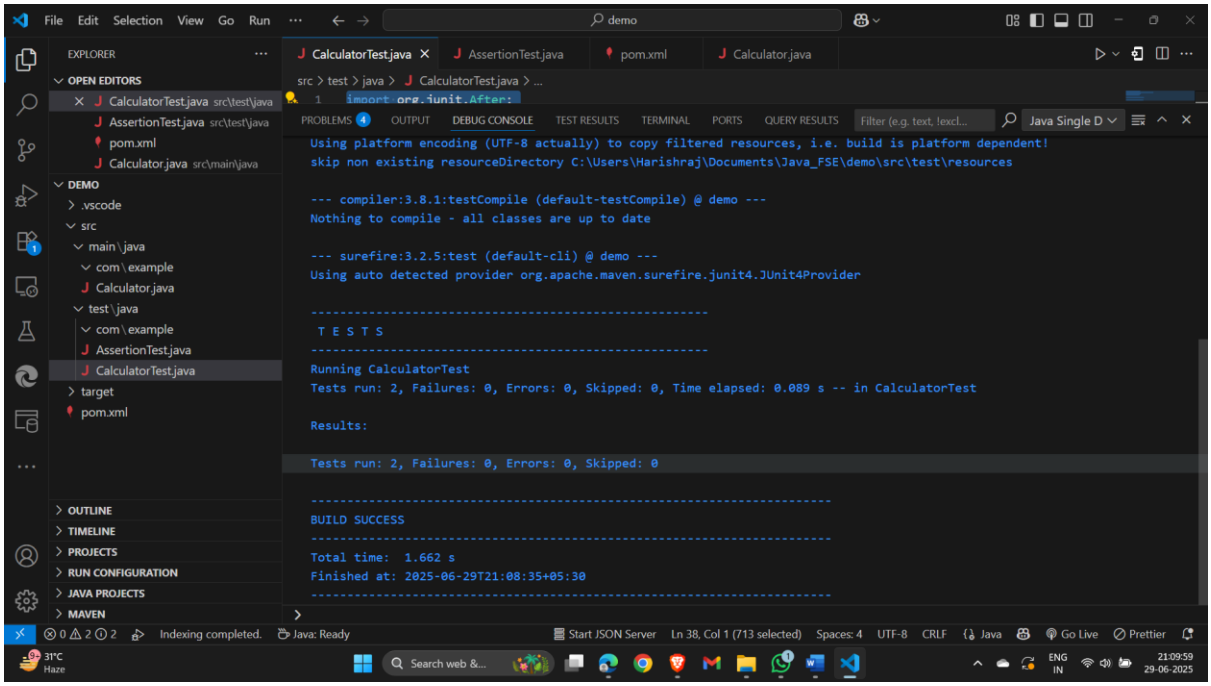
```
    }

    public int multiply(int a, int b) {
        return a * b;
    }

    public double divide(int a, int b) {
        if (b == 0) {
            throw new IllegalArgumentException("Division by zero is not allowed.");
        }
        return (double) a / b;
    }
}
```

CalculatorTest.java

```
import org.junit.After;
import static org.junit.Assert.assertEquals;
import org.junit.Before;
import org.junit.Test;

public class CalculatorTest {

    private Calculator calc;

    @Before
    public void setUp() {
        calc = new Calculator();  // Arrange (shared)
    }

    @After
    public void tearDown() {
```

```
        calc = null;  // Cleanup
    }

    @Test
    public void testAddition() {
        // Act
        int result = calc.add(4, 6);

        // Assert
        assertEquals(10, result);
    }

    @Test
    public void testSubtraction() {
        // Act
        int result = calc.subtract(9, 5);

        // Assert
        assertEquals(4, result);
    }
}
```

Output:


File Edit Selection View Go Run
demo
EXPLORER
OPEN EDITORS
CalculatorTest.java src\test\java
AssertionTest.java src\test\java
pom.xml
Calculator.java src\main\java
DEMO
.vscode
src
main\java
com\example
Calculator.java
test\java
com\example
AssertionTest.java
CalculatorTest.java
target
pom.xml
OUTLINE
TIMELINE
PROJECTS
RUN CONFIGURATION
JAVA PROJECTS
MAVEN
CalculatorTest.java
AssertionTest.java
pom.xml
Calculator.java
src > test > java > CalculatorTest.java > ...
1 import org.junit.After;
PROBLEMS 4 OUTPUT DEBUG CONSOLE TEST RESULTS TERMINAL PORTS QUERY RESULTS
Filter (e.g. text, !excl...
Java Single D
Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
skip non existing resourceDirectory C:\Users\Harishraj\Documents\Java_FSE\demo\src\test\resources
--- compiler:3.8.1:testCompile (default-testCompile) @ demo ---
Nothing to compile - all classes are up to date
--- surefire:3.2.5:test (default-cli) @ demo ---
Using auto detected provider org.apache.maven.surefire.junit4.JUnit4Provider
----
T E S T S

Running CalculatorTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.089 s -- in CalculatorTest
Results:
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

BUILD SUCCESS

Total time: 1.662 s
Finished at: 2025-06-29T21:08:35+05:30

Indexing completed.
Java: Ready
Start JSON Server
Ln 38, Col 1 (713 selected)
Spaces: 4
UTF-8
CRLF
Java
Go Live
Prettier
31°C
Haze
Search web &...
ENG
IN
21:09:59
29-06-2025