Perform Parameterized Testing for Calculator Program Using JUnit 5

Step 1: Class to Be Tested — Calculator.java

This class contains a simple method to perform addition.

```
public class Calculator {

    // Method to add two numbers
    public int add(int a, int b) {
        return a + b;
    }
}
```

Step 2: Using @CsvSource for Testing

This JUnit test uses @CsvSource to provide multiple sets of inputs directly within the test class.

```
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;
import static org.junit.jupiter.api.Assertions.assertEquals;

public class CalculatorTest {

    Calculator calculator = new Calculator();

    @ParameterizedTest
    @CsvSource({
        "2, 3, 5",
        "5, 5, 10",
        "10, 20, 30",
        "7, 8, 15",
        "15, 25, 40"
    })
    void testAdditionWithCsvSource(int num1, int num2, int expectedOutput) {
        int result = calculator.add(num1, num2);
        System.out.printf("Input: %d + %d | Expected Output: %d | Actual Output: %d%n",
num1, num2, expectedOutput, result);
        assertEquals(expectedOutput, result, "Addition result is incorrect!");
    }
}
```

Step 3: Using @CsvFileSource for External CSV Files

If you want to store the data in a CSV file instead of writing it inside the test class, follow this approach.

CSV File (src/test/resources/numbers.csv)

```
num1,num2,expectedOutput
2,3,5
5,5,10
10,20,30
7,8,15
15,25,40
```

JUnit Test Using @CsvFileSource

```
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvFileSource;
import static org.junit.jupiter.api.Assertions.assertEquals;

public class CalculatorTest {

   Calculator calculator = new Calculator();

   @ParameterizedTest
   @CsvFileSource(resources = "/numbers.csv", numLinesToSkip = 1)
   void testAdditionWithCsvFileSource(int num1, int num2, int expectedOutput) {
      int result = calculator.add(num1, num2);
      System.out.printf("Input: %d + %d | Expected Output: %d | Actual Output: %d%n",
num1, num2, expectedOutput, result);
      assertEquals(expectedOutput, result, "Addition result is incorrect!");
   }
}
```

Summary

Method	Description
@ValueSource	Used for single input values, like checking positivity.
@CsvSource	Provides test data directly within the test class.
@CsvFileSource	Reads test data from an external CSV file.