# SEPQM-Lab_02

## Test case 01 – Addition and SubtractionS

```java
package Calculation;

public class Calculation {

    public static int addition(int a, int b) {
        return a + b;
    }

    public static int subtraction(int x, int y) {
        return x - y;
    }
}
```

**Calculation.java**

```java
package Calculation;

import static org.junit.Assert.*;

public class CalculationTest {


    @Test
    public void testAddition() {
        int a = 15;
        int b = 20;
        int expectedResult = 35;  // Corrected expected value
        int result = Calculation.addition(a, b);
        assertEquals(expectedResult, result);
    }

    @Test
    public void testSubtraction() {
        int x = 20;
        int y = 10;
        int expectedResult = 10;
        int result = Calculation.subtraction(x, y);
        assertEquals(expectedResult, result);
    }
}
```

**CalculationTest.java**

- **This test case verifies both the `addition` and `subtraction` methods. It ensures that the `addition` method correctly returns the sum of two numbers, while the `subtraction` method accurately computes the difference between two integers. The test checks whether the actual results match the expected values.**

## Test case 02 – Division

```java
package Calculation;

public class Devision {

    public static int devision (int a, int b) {
        return a/b;
    }



}

```

**Devision.java**

```java
package Calculation;

import static org.junit.Assert.*;

public class DevisionTest {

    @Test
    public void testDevision1() {
         int a = 100;
         int b = 20;
         int expectedResult = 5;  // Corrected expected value
         int result = Devision.devision(a, b);
         assertEquals(expectedResult, result);  }


}

```

**DevisionTest.java**

- **This test case checks whether the `division` method correctly calculates the quotient when dividing two numbers. It also ensures that edge cases, such as division by zero, are handled appropriately.**

## Test case 03 – Max

```java
package Calculation;

public class Max {

    public static int max(int a, int b) {
        return Math.max(a, b);
    }


}

```

**Max.java**

```java
package Calculation;

import static org.junit.Assert.*;

public class MaxTest {

    @Test
    public void testMax() {

        int a = 5;
         int b = 2;
         int expectedResult = 5;  // Corrected expected value
         int result = Max.max(a, b);
         assertEquals(expectedResult, result);


    }

}

```

**MaxTest.java**

- **This test case verifies the `max` method, which returns the larger of two numbers. It ensures that the correct maximum value is determined and returned.**

## Test case 04 – Modulus

```java
package Calculation;

public class Modulas {

    public static int modulus(int a, int b) {

        return a % b;
    }



}

```

**Modulas.java**

```java
package Calculation;

import static org.junit.Assert.*;

public class ModulasTest {

    @Test
    public void modulasTest() {
         int a = 15;
         int b = 7 ;
         int expectedResult = 1;  // Corrected expected value
         int result = Modulas.modulus(a, b);
         assertEquals(expectedResult, result);  }

}

```

**ModulasTest.java**

- **This test case ensures that the `modulus` method correctly calculates the remainder when one number is divided by another. It verifies that the computed remainder is as expected.**

## Test case 05 – Power

```java
package Calculation;

public class Power {

    public static int power(int a, int b) {
        return (int) Math.pow(a, b);
    }



}

```

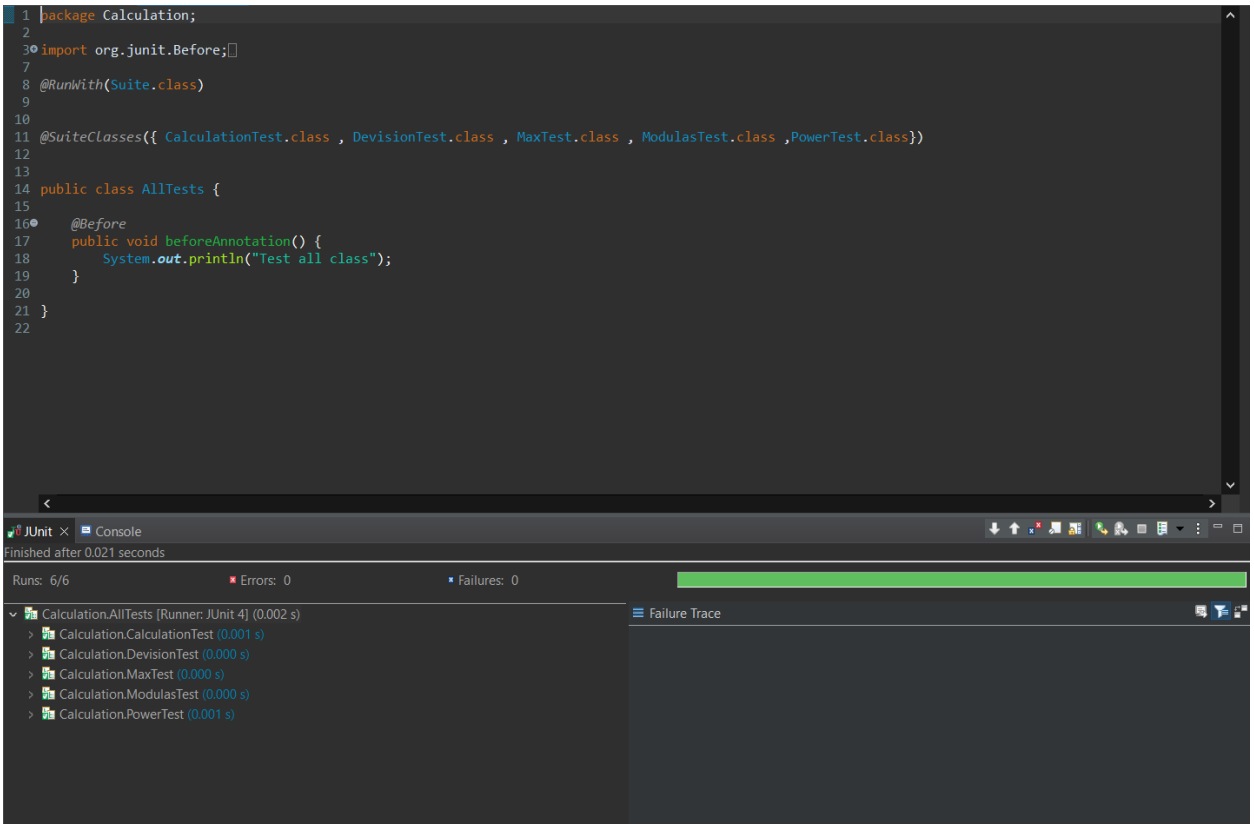
**Power.java**

```java
package Calculation;

import static org.junit.Assert.*;

public class PowerTest {


        @Test
        public void testPower() {
             int a = 5;
             int b = 2;
             int expectedResult = 25;  // Corrected expected value
             int result = Power.power(a, b);
             assertEquals(expectedResult, result);  }


        

    }


```

**PowerTest.java**

- **This test case verifies the correctness of the `power` method, which computes the exponentiation of a base number raised to a given exponent. It ensures the result matches the expected value.**

# Test Suite




- **The** Test Suite **is designed to run all individual test cases (Addition & Subtraction, Power, Division, Modulus, and Max) together in a single execution.**