Lab Assesment-7

1.Write a JUnit 5 test case for a simple Calculator class that has methods for addition, subtraction, multiplication, and division. Test each operation with multiple test cases.

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>Lab</groupId>
  <artifactId>JUnit</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <dependencies>
      <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-engine</artifactId>
            <version>5.5.2</version>
            <scope>test</scope>
      </dependency>
  </dependencies>
</project>
```

```
package com.employee;

public class BasicSalaryCalculator {
      private double basicSalary;

      public double getBasicSalary() {
            return basicSalary;
      }

      public void setBasicSalary(double basicSalary) {
            if (basicSalary < 0) {
                  throw new IllegalArgumentException("Negative salary is
invalid.");
            }
            this.basicSalary = basicSalary;
      }

      public double getGrossSalary() {
            return this.basicSalary + getSocialInsurance() +
getAdditionalBonus();
      }

      public double getSocialInsurance() {
            return this.basicSalary * 25 / 100;
      }
```

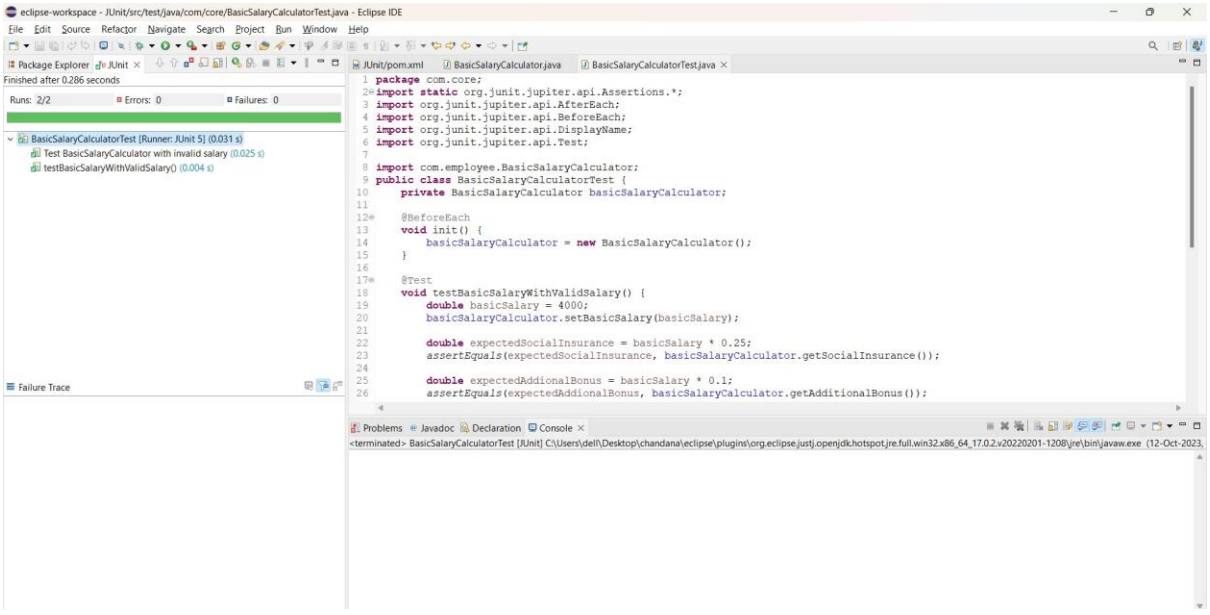
```
	public double getAdditionalBonus() {
		return this.basicSalary / 10;
	}
}
```

```
package com.core;

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

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.DisplayName;

import org.junit.jupiter.api.Test;


import com.employee.BasicSalaryCalculator;

public class BasicSalaryCalculatorTest {

	private BasicSalaryCalculator basicSalaryCalculator;


	@BeforeEach

	void init() {

		basicSalaryCalculator = new BasicSalaryCalculator();

	}


	@Test

	void testBasicSalaryWithValidSalary() {

		double basicSalary = 4000;

		basicSalaryCalculator.setBasicSalary(basicSalary);


		double expectedSocialInsurance = basicSalary * 0.25;

		assertEquals(expectedSocialInsurance,
basicSalaryCalculator.getSocialInsurance());
```

```
		double expectedAddionalBonus = basicSalary * 0.1;

		assertEquals(expectedAddionalBonus,
basicSalaryCalculator.getAdditionalBonus());


		double expectedGross = basicSalary + expectedSocialInsurance +
expectedAddionalBonus;

		assertEquals(expectedGross, basicSalaryCalculator.getGrossSalary());


	}


	@DisplayName("Test BasicSalaryCalculator with invalid salary")

	@Test

	void testBasicSalaryWithInValidSalary() {


		double basicSalary = -100;

		assertThrows(IllegalArgumentException.class, () -> {

			basicSalaryCalculator.setBasicSalary(basicSalary);

		});


	}


	@AfterEach

	void tearDown() {

		basicSalaryCalculator = null;

	}

}
```

Output:

```
       -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running com.core. [1mBasicSalaryCalculatorTest [m
[INFO]  [1;32mTests run:  [0;1;32m2 [m, Failures: 0, Errors:
0, Skipped: 0, Time elapsed: 0.117 s -- in
com.core. [1mBasicSalaryCalculatorTest [m
[INFO]
[INFO] Results:
[INFO]
[INFO]  [1;32mTests run: 2, Failures: 0, Errors: 0, Skipped:
0 [m
[INFO]
[INFO]  [1m-------------------------------------------------
---------------------- [m
[INFO]  [1;32mBUILD SUCCESS [m
[INFO]  [1m-------------------------------------------------
---------------------- [m
[INFO] Total time:  5.465 s
[INFO] Finished at: 2023-10-12T18:53:25+05:30
[INFO]  [1m-------------------------------------------------
---------------------- [m
```



2.Write a JUnit 5 test case to test a method that is expected to throw a specific exception for an invalid input.

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>JUnit</groupId>
  <artifactId>LabExercise</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <dependencies>
      <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-engine</artifactId>
            <version>5.5.2</version>
            <scope>test</scope>
      </dependency>
  </dependencies>
</project>
```

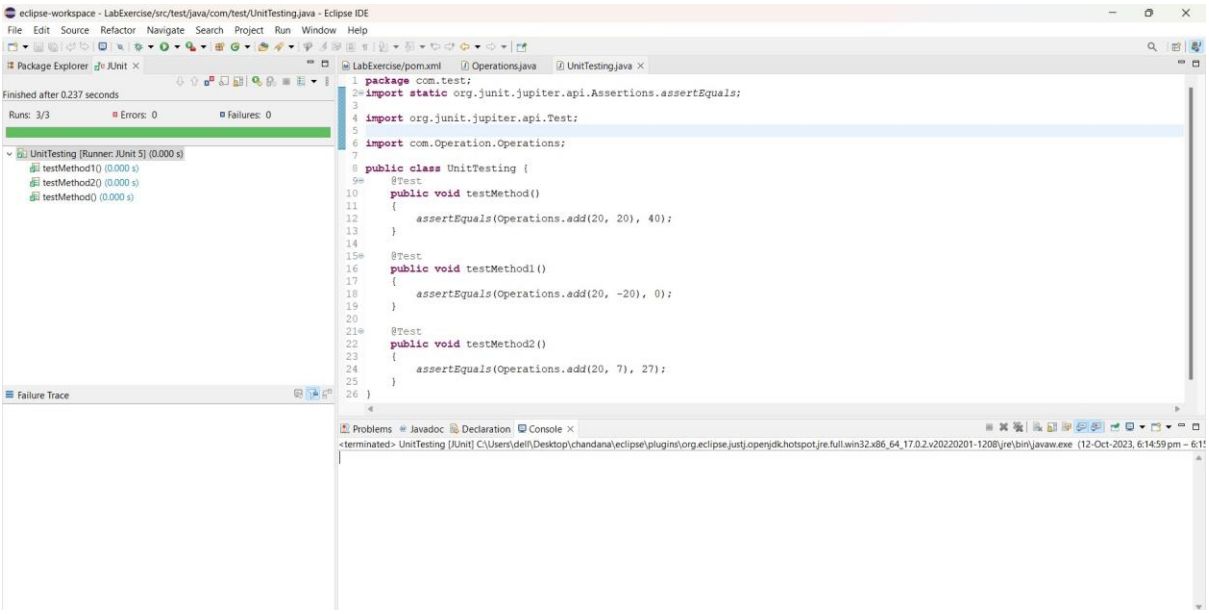
```
package com.Operation;

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

```
package com.test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;

import com.Operation.Operations;

public class UnitTesting {

      @Test

      public void testMethod()

      {

      assertEquals(Operations.add(20, 20), 40);
```

```
	}

	@Test
	public void testMethod1()
	{
		assertEquals(Operations.add(20, -20), 0);
	}

	@Test
	public void testMethod2()
	{
		assertEquals(Operations.add(20, 7), 27);
	}
}
```

Output:

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running UnitTesting
Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.021 s - in UnitTesting

Results:

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0
```



3.Write a JUnit 5 test case that uses the @RepeatedTest annotation to repeat a test a certain number of times.

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

import org.junit.jupiter.api.DisplayName;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class RepeatedTestExample {


    @RepeatedTest(3) // Repeat the test 3 times

    @DisplayName("Repeated Addition Test")

    void repeatedAdditionTest() {

        Calculator calculator = new Calculator();

        int result = calculator.add(2, 3);

        assertEquals(5, result, "Expected result is 5");

    }

}
```

Output:

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running RepeatedTestExample
Repeating Addition Test
Repeating Addition Test
Repeating Addition Test
Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.021 s - in RepeatedTestExample

Results:

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0
```