Faculty of Engineering
Ain Shams University

REPORT LAB (1)

NAME	ID	EMAIL
Mennat – Allah Ashraf Fouad Fetouh	**17p3051**	mennatallaashraf@gmail.com

Submitted to:

Dr. Islam El-Maddah
Eng. Adham Nour

Problem 1: *Watermelon*

- ***Problem Description:***

One hot summer day Pete and his friend Billy decided to buy a watermelon. They chose the biggest and the ripest one, in their opinion. After that the watermelon was weighed, and the scales showed *w* kilos. They rushed home, dying of thirst, and decided to divide the berry, however they faced a hard problem.

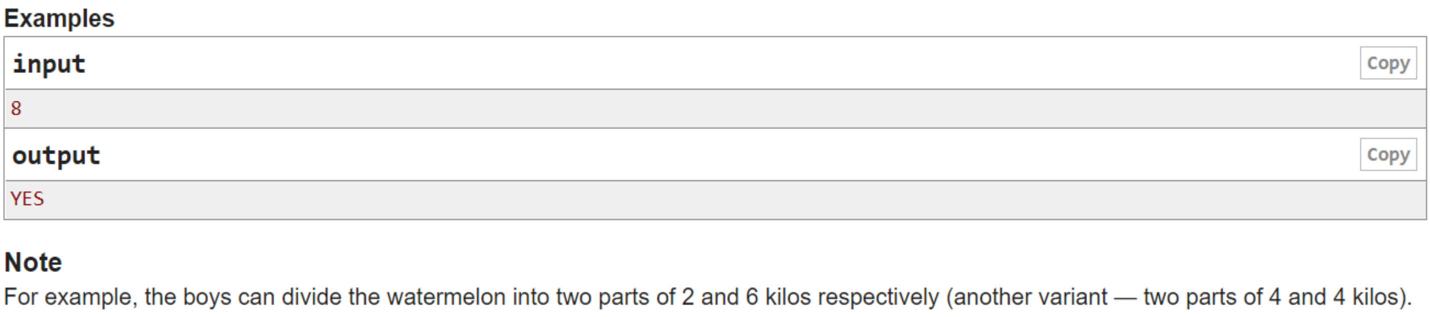
Pete and Billy are great fans of even numbers, that's why they want to divide the watermelon in such a way that each of the two parts weighs even number of kilos, at the same time it is not obligatory that the parts are equal. The boys are extremely tired and want to start their meal as soon as possible, that's why you should help them and find out, if they can divide the watermelon in the way they want. For sure, each of them should get a part of positive weight.

Input

The first (and the only) input line contains integer number w ($1 \leq w \leq 100$) — the weight of the watermelon bought by the boys.

Output

Print YES, if the boys can divide the watermelon into two parts, each of them weighing even number of kilos; and NO in the opposite case.



Examples

input

```
8
```

output

```
YES
```

Note

For example, the boys can divide the watermelon into two parts of 2 and 6 kilos respectively (another variant — two parts of 4 and 4 kilos).

Solution of Problem 1:

1- The main code:

Watermelon.java

```java
public class Watermelon {
    public int watermelon(int w)    // w is weight
    {
       if (w<1 || w> 100)   //1 ≤ w ≤ 100
       {
           System.out.println("WARNING:");
           System.out.println("**********WRONG Entered weight********");
           System.out.println("The weight RANGE:(1 ≤ w ≤ 100)");
           return -1;
       }

       else{
           if (w%2 ==0 && w!=2)
           {
               System.out.println("YES");
               return 1;
           }

           else
           {
               System.out.println("NO");
               return 0;
           }

       }


    }
}
```

2- Test Cases:

WatermelonTest.java

```java
import org.junit.Test;

import static org.junit.Assert.*;

public class WatermelonTest {


    @Test
    public void odd_weight() {
        Watermelon tester = new Watermelon();
        assertEquals("NO",0,tester.watermelon(5));
    }



    @Test
    public void even_weight() {

        Watermelon tester = new Watermelon();
        assertEquals("YES",1,tester.watermelon(8));

    }

    @Test
    public void weight_equal_2kilo() {
        Watermelon tester = new Watermelon();
        assertEquals("NO",0,tester.watermelon(2));
    }

    @Test
    public void weight_larger_than_100kilo() {
        Watermelon tester = new Watermelon();
        assertEquals(-1,tester.watermelon(150));

    }

    @Test
    public void ZERO_weight() {
        Watermelon tester = new Watermelon();
        assertEquals(-1,tester.watermelon(0));    }

}
```

Test Cases that fails:

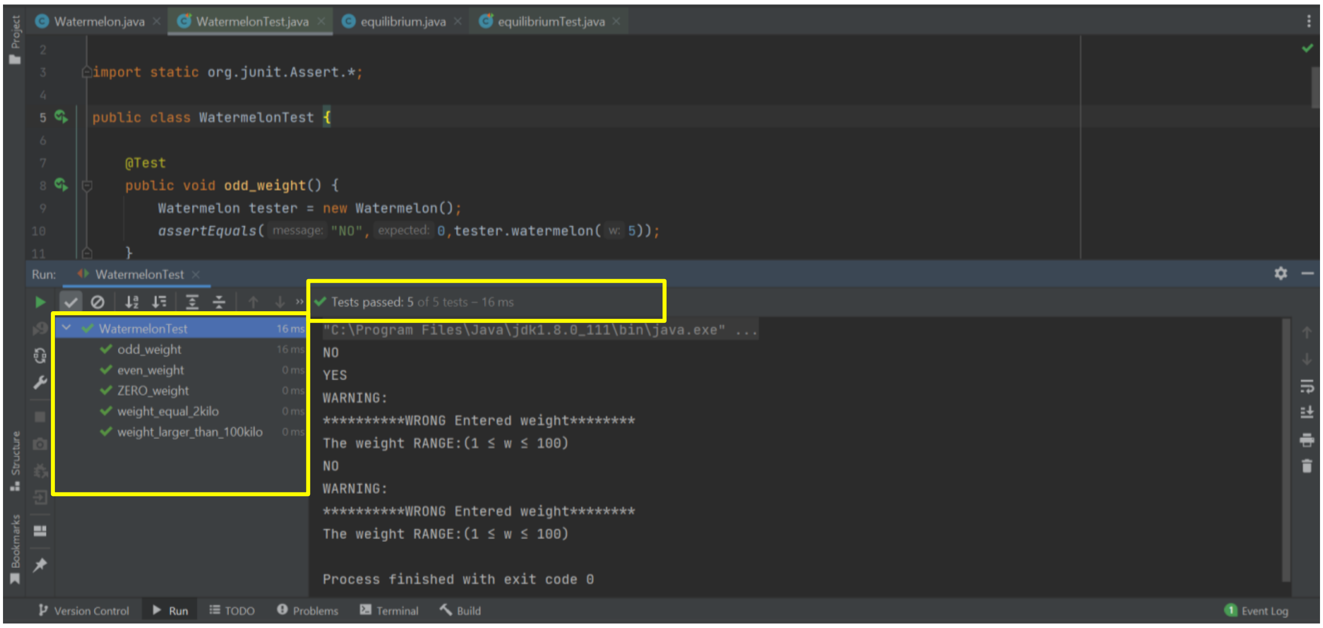
No test cases fails

Test Cases that succeeded:

All test cases are succeeded.

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

public class WatermelonTest {

    @Test
    public void odd_weight() {
        Watermelon tester = new Watermelon();
        assertEquals("NO",0,tester.watermelon(5));
    }

    @Test
    public void even_weight() {

        Watermelon tester = new Watermelon();
        assertEquals("YES",1,tester.watermelon(8));

    }
    @Test
    public void weight_equal_2kilo() {
        Watermelon tester = new Watermelon();
        assertEquals("NO",0,tester.watermelon(2));
    }

    @Test
    public void weight_larger_than_100kilo() {
        Watermelon tester = new Watermelon();
        assertEquals(-1,tester.watermelon(150));

    }

    @Test
    public void ZERO_weight() {
        Watermelon tester = new Watermelon();
        assertEquals(-1,tester.watermelon(0));     }

}
```

Test Cases "Run" window:



```
import static org.junit.Assert.*;

public class WatermelonTest {

    @Test
    public void odd_weight() {
        Watermelon tester = new Watermelon();
        assertEquals( message: "NO", expected: 0,tester.watermelon( w: 5));
    }
```

✓ Tests passed: 5 of 5 tests – 16 ms

- ✓ WatermelonTest 16 ms
  - ✓ odd_weight 16 ms
  - ✓ even_weight 0 ms
  - ✓ ZERO_weight 0 ms
  - ✓ weight_equal_2kilo 0 ms
  - ✓ weight_larger_than_100kilo 0 ms

```
"C:\Program Files\Java\jdk1.8.0_111\bin\java.exe" ...
NO
YES
WARNING:
**********WRONG Entered weight********
The weight RANGE:(1 ≤ w ≤ 100)
NO
WARNING:
**********WRONG Entered weight********
The weight RANGE:(1 ≤ w ≤ 100)

Process finished with exit code 0
```

Problem 2: *Young Physicist*

- ***Problem Description:***

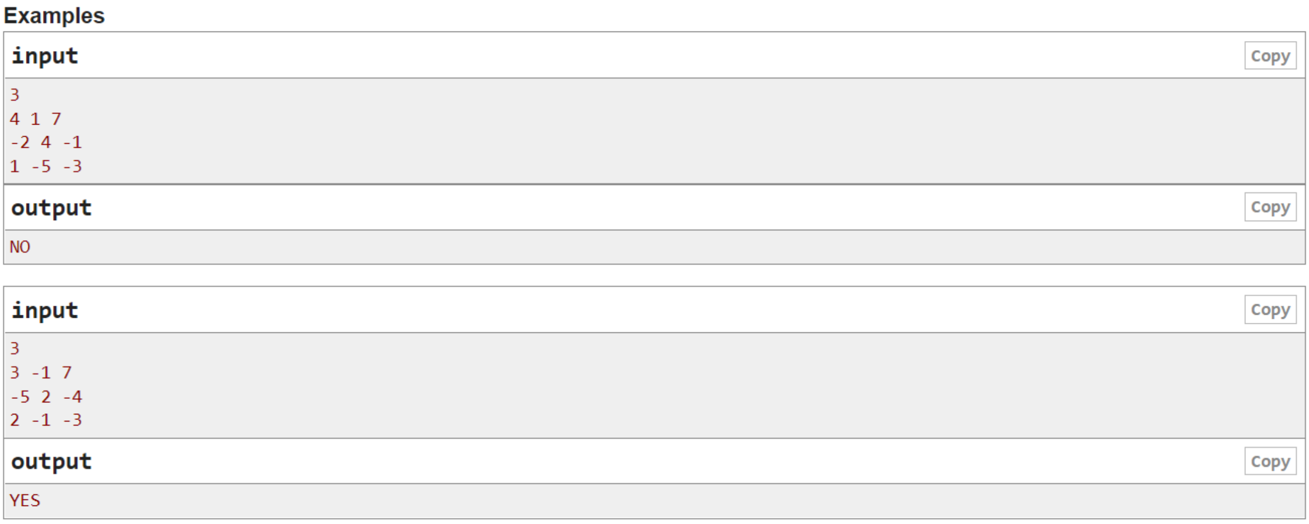
A guy named Vasya attends the final grade of a high school. One day Vasya decided to watch a match of his favorite hockey team. And, as the boy loves hockey very much, even more than physics, he forgot to do the homework. Specifically, he forgot to complete his physics tasks. Next day the teacher got very angry at Vasya and decided to teach him a lesson. He gave the lazy student a seemingly easy task: You are given an idle body in space and the forces that affect it. The body can be considered as a material point with coordinates (0; 0; 0). Vasya had only to answer whether it is in equilibrium. "Piece of cake" — thought Vasya, we need only to check if the sum of all vectors is equal to 0. So, Vasya began to solve the problem. But later it turned out that there can be lots and lots of these forces, and Vasya can not cope without your help. Help him. Write a program that determines whether a body is idle or is moving by the given vectors of forces.

Input

The first line contains a positive integer n ($1 \le n \le 100$), then follow n lines containing three integers each: the x_i coordinate, the y_i coordinate and the z_i coordinate of the force vector, applied to the body ($-100 \le x_i, y_i, z_i \le 100$).

Output

Print the word "YES" if the body is in equilibrium, or the word "NO" if it is not.



Examples

input

```
3
4 1 7
-2 4 -1
1 -5 -3
```

output

```
NO
```

input

```
3
3 -1 7
-5 2 -4
2 -1 -3
```

output

```
YES
```

Solution of Problem 2:

1- The main code:

equilibruim.java

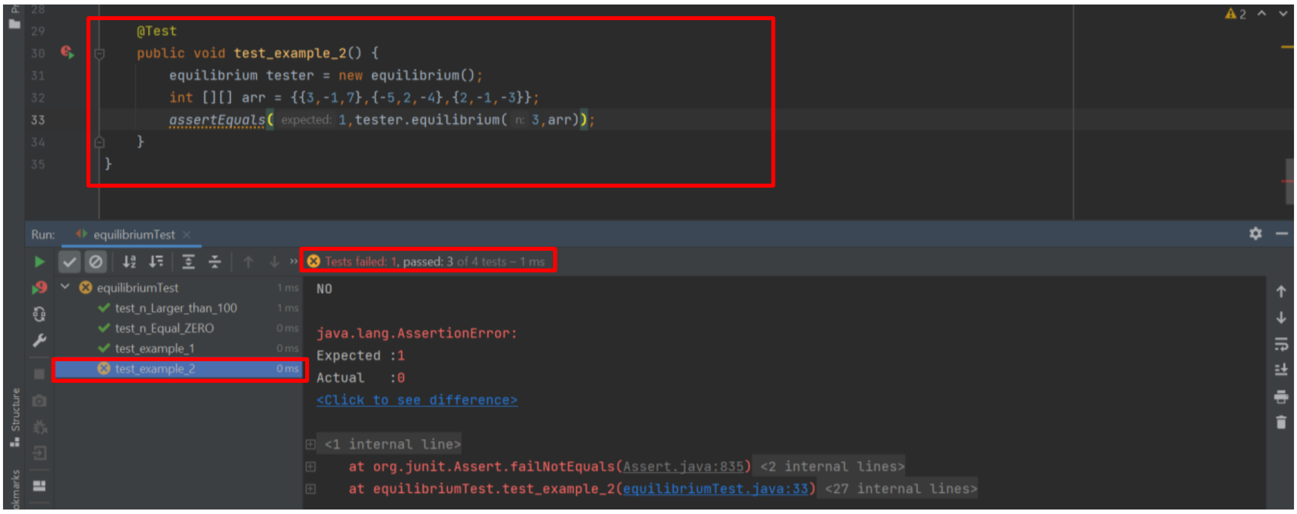
```
public class equilibrium {
    public int equilibrium (int n, int[][] arr)    //n is number of lines
    {
        if (n<1 || n> 100)     //1 ≤ n ≤ 100
        {
            System.out.println("WARNING:");
            System.out.println("******* WRONG NO. OF LINES ENTERED *******");
            System.out.println("NO. OF LINES (n) RANGE: (1 ≤ n ≤ 100)");
            return -1;
        }

        else
        {
            int [] sum ={0,0,0};

            for(int i=0; i < n ; i++)         //i = row
            {
                for (int j=0; j < 3 ; j++)   //j = column
                {
                    int x = arr[i][j];
                    sum[j] = +x;            // Sum of column items
                }
            }


            if (sum[0]==0 && sum[1]==0 && sum[2]==0){
                System.out.println("YES");
                return 1;
            }

            else
            {
                System.out.println("NO");
                return 0;
            }

        }
    }
}
```

2- Test Cases:

equilibruimTest.java

```
import org.junit.Test;

import static org.junit.Assert.*;

public class equilibriumTest {

    @Test
    public void test_n_Equal_ZERO() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals(-1,tester.equilibrium(0,arr));
    }

    @Test
    public void test_n_Larger_than_100() {

        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals(-1,tester.equilibrium(200,arr));
    }

    @Test
    public void test_example_1() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals("NO",0,tester.equilibrium(3,arr));
    }

    @Test
    public void test_example_2() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{3,-1,7},{-5,2,-4},{2,-1,-3}};
        assertEquals(1,tester.equilibrium(3,arr));
    }
}
```

Test Cases that fails:



```
    @Test
    public void test_example_2() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{3,-1,7},{-5,2,-4},{2,-1,-3}};
        assertEquals( expected: 1,tester.equilibrium( n: 3,arr));
    }
}
```

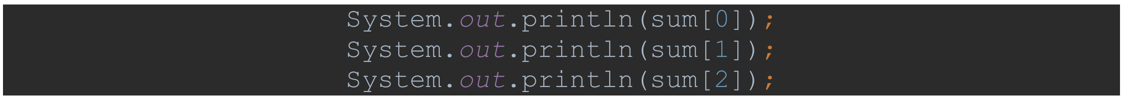
Tests failed: 1, passed: 3 of 4 tests – 1 ms

```
NO

java.lang.AssertionError:
Expected :1
Actual   :0
<Click to see difference>

<1 internal line>
    at org.junit.Assert.failNotEquals(Assert.java:835) <2 internal lines>
    at equilibriumTest.test_example_2(equilibriumTest.java:33) <27 internal lines>
```

All other test cases are **succeeded**.

How to fix the test case that failed?

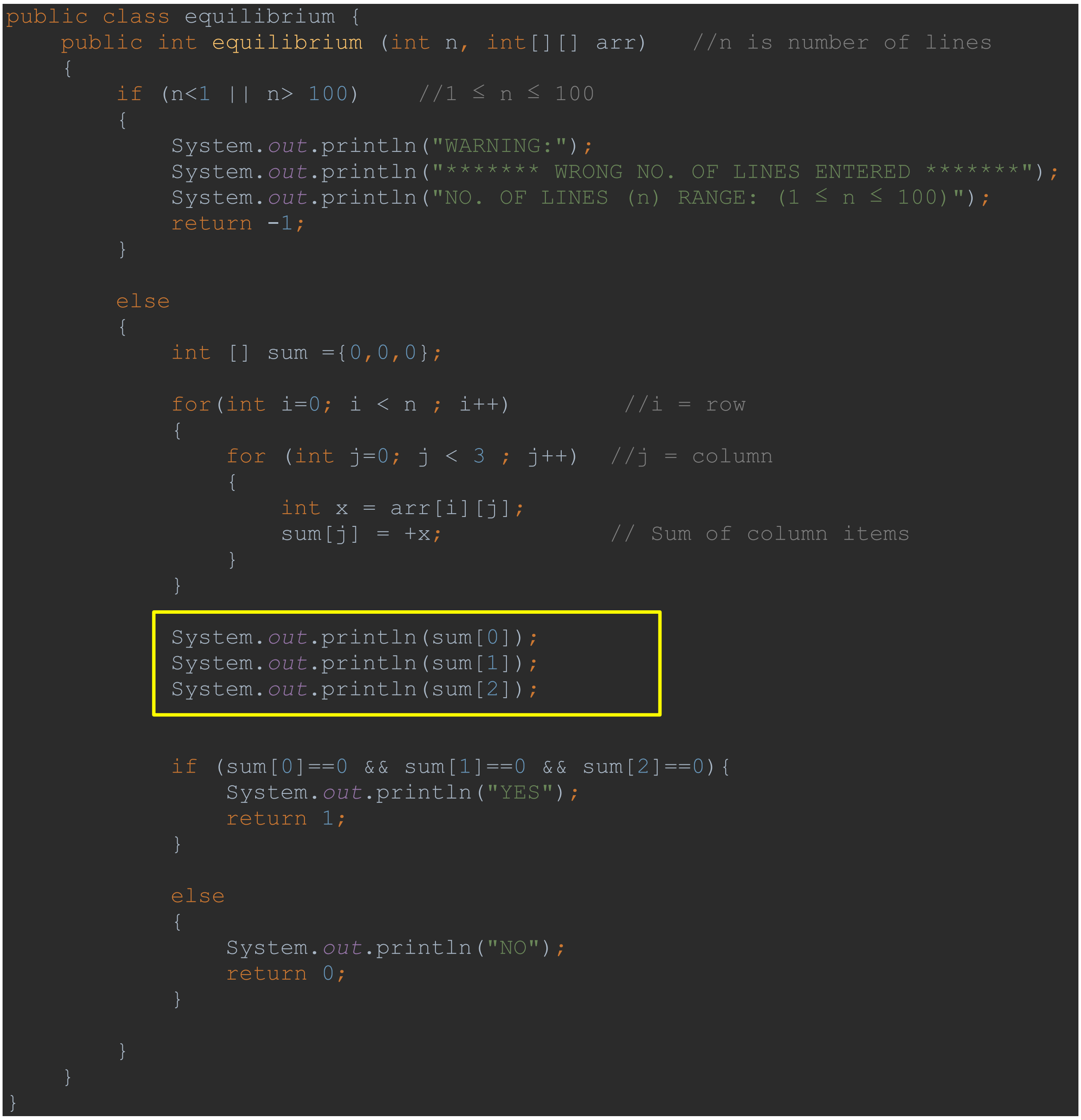
1- Add to the main code ***"equilibruim.java"*** the following code after the for loop and before the internal if-statement in the else part:



```
System.out.println(sum[0]);
System.out.println(sum[1]);
System.out.println(sum[2]);
```

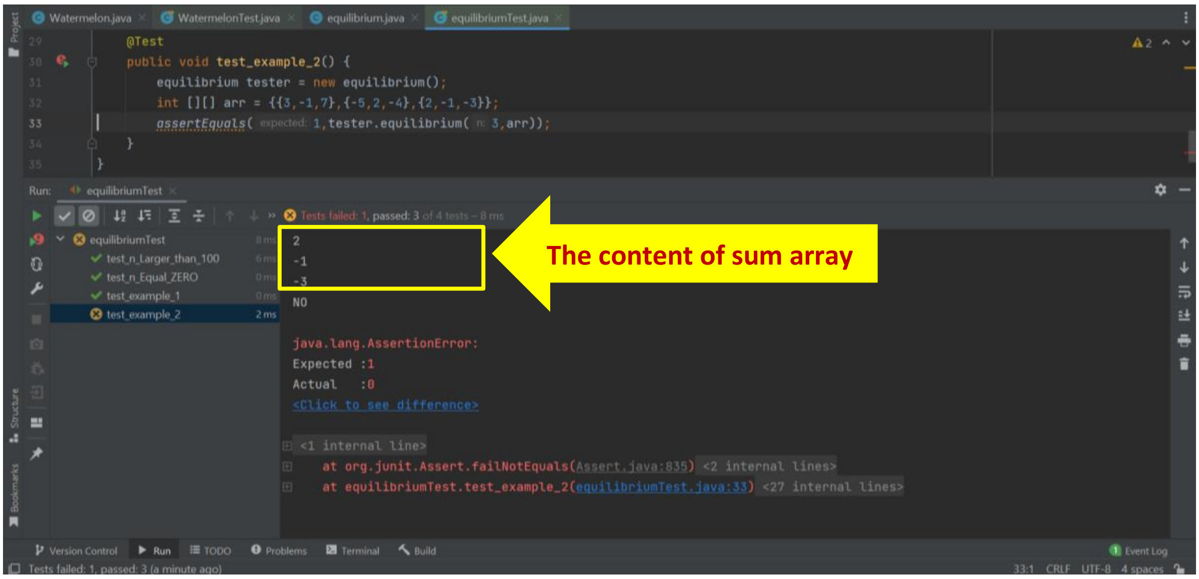
This will show the sum array content.

The main code ***"equilibruim.java"*** will be:



```
public class equilibrium {
    public int equilibrium (int n, int[][] arr)    //n is number of lines
    {
        if (n<1 || n> 100)     //1 ≤ n ≤ 100
        {
            System.out.println("WARNING:");
            System.out.println("******* WRONG NO. OF LINES ENTERED *******");
            System.out.println("NO. OF LINES (n) RANGE: (1 ≤ n ≤ 100)");
            return -1;
        }

        else
        {
            int [] sum ={0,0,0};

            for(int i=0; i < n ; i++)          //i = row
            {
                for (int j=0; j < 3 ; j++)   //j = column
                {
                    int x = arr[i][j];
                    sum[j] = +x;             // Sum of column items
                }
            }

            System.out.println(sum[0]);
            System.out.println(sum[1]);
            System.out.println(sum[2]);


            if (sum[0]==0 && sum[1]==0  && sum[2]==0){
                System.out.println("YES");
                return 1;
            }

            else
            {
                System.out.println("NO");
                return 0;
            }

        }
    }
}
```

Then, save changes and run again the test cases:



```
@Test
public void test_example_2() {
    equilibrium tester = new equilibrium();
    int [][] arr = {{3,-1,7},{-5,2,-4},{2,-1,-3}};
    assertEquals( expected: 1,tester.equilibrium( n: 3,arr));
}
}
```

Tests failed: 1, passed: 3 of 4 tests – 8 ms

```
2
-1
-3
NO

java.lang.AssertionError:
Expected :1
Actual   :0
<Click to see difference>

<1 internal line>
    at org.junit.Assert.failNotEquals(Assert.java:835) <2 internal lines>
    at equilibriumTest.test_example_2(equilibriumTest.java:33) <27 internal lines>
```

The content of sum array

Tests failed: 1, passed: 3 (a minute ago)

This content results from adding only the first 2 elements on column without adding the last element. Thus, the for loop doesn't work as expected.

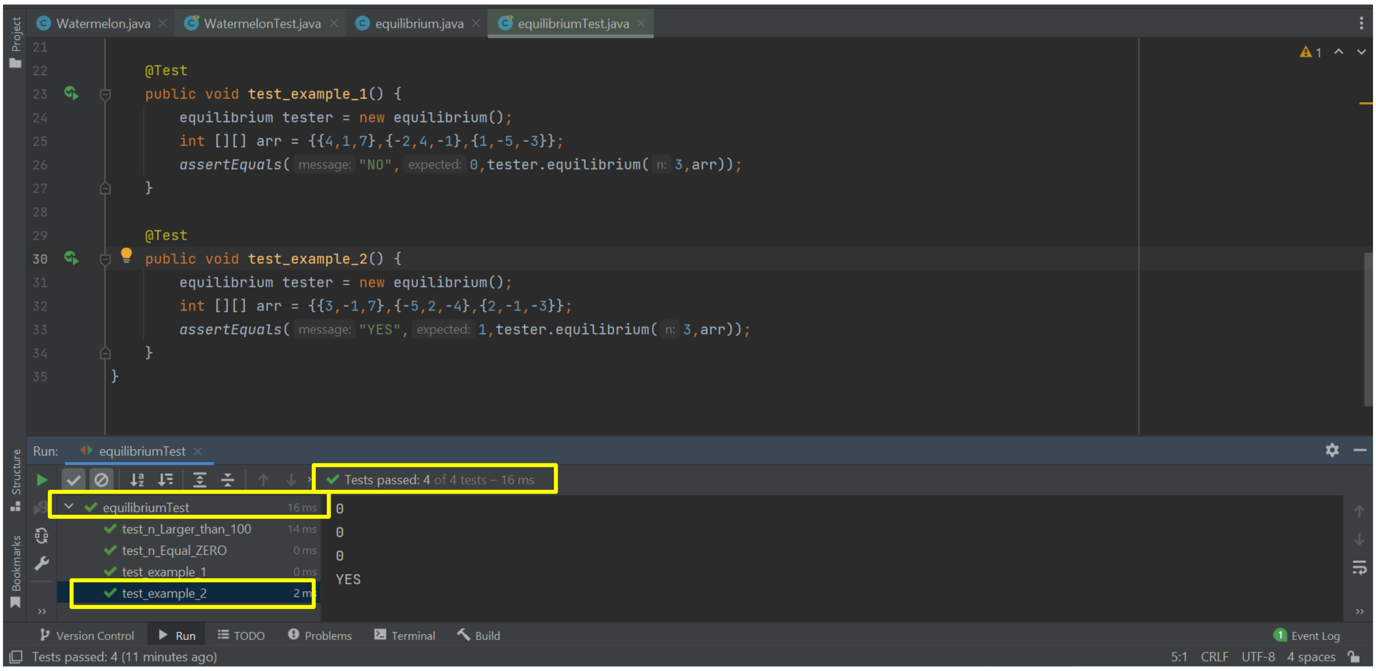
- After reviewing the for-loop code again, and change the code from :

```
sum[j] =+ x;          // Sum of column items
```

To:

```
sum[j] += x;          // Sum of column items
```

Then, it runs as expected and all test cases are succeeded.



```
@Test
public void test_example_1() {
    equilibrium tester = new equilibrium();
    int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
    assertEquals( message: "NO", expected: 0,tester.equilibrium( n: 3,arr));
}

@Test
public void test_example_2() {
    equilibrium tester = new equilibrium();
    int [][] arr = {{3,-1,7},{-5,2,-4},{2,-1,-3}};
    assertEquals( message: "YES", expected: 1,tester.equilibrium( n: 3,arr));
}
}
```

Tests passed: 4 of 4 tests – 16 ms

```
0
0
0
YES
```

Tests passed: 4 (11 minutes ago)

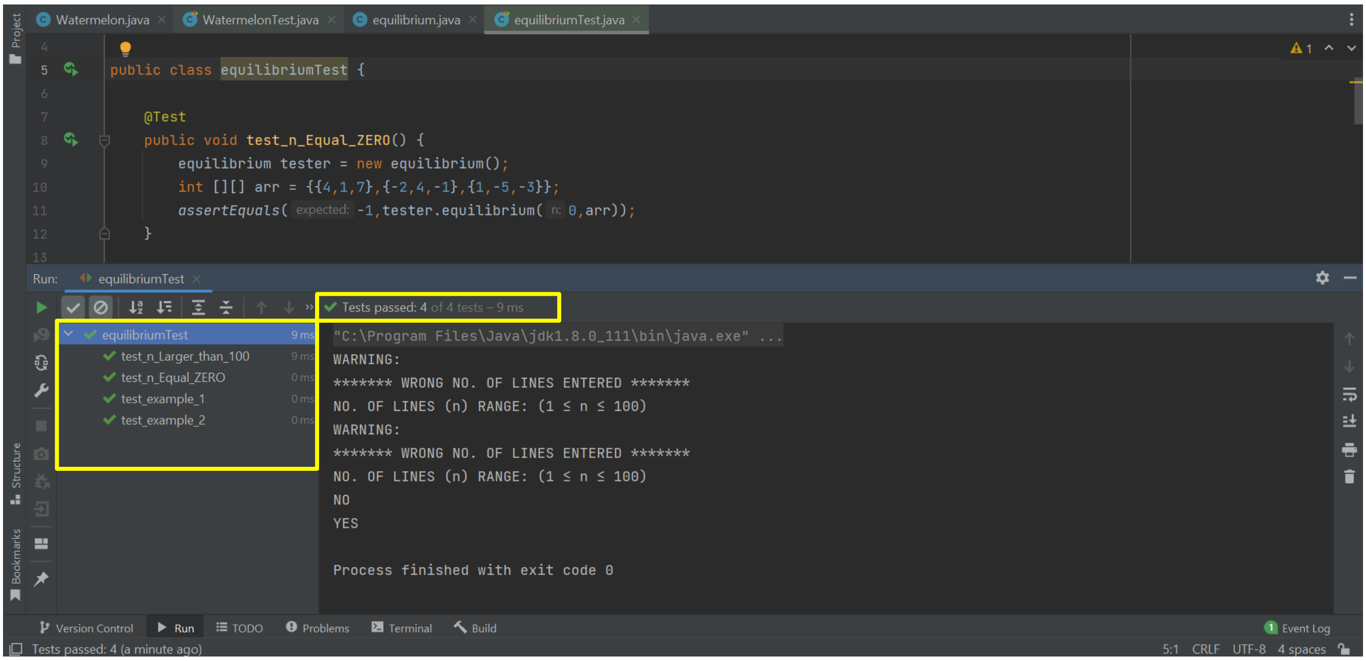
The main code ***"equilibruim.java"*** after editing is:

```
public class equilibrium {
    public int equilibrium (int n, int[][] arr)   //n is number of lines
    {
        if (n<1 || n> 100)     //1 ≤ n ≤ 100
        {
            System.out.println("WARNING:");
            System.out.println("******* WRONG NO. OF LINES ENTERED *******");
            System.out.println("NO. OF LINES (n) RANGE: (1 ≤ n ≤ 100)");
            return -1;
        }

        else
        {
            int [] sum ={0,0,0};

            for(int i=0; i < n ; i++)          //i = row
            {
                for (int j=0; j < 3 ; j++)   //j = column
                {
                    int x = arr[i][j];
                    sum[j] += x;               // Sum of column items
                }
            }


            if (sum[0]==0 && sum[1]==0 && sum[2]==0){
                System.out.println("YES");
                return 1;
            }

            else
            {
                System.out.println("NO");
                return 0;
            }

        }
    }
}
```

The code of test cases ***"equilibruimTest.java"*** is:

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

public class equilibriumTest {

    @Test
    public void test_n_Equal_ZERO() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals(-1,tester.equilibrium(0,arr));
    }

    @Test
    public void test_n_Larger_than_100() {

        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals(-1,tester.equilibrium(200,arr));
    }

    @Test
    public void test_example_1() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals("NO",0,tester.equilibrium(3,arr));
    }

    @Test
    public void test_example_2() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{3,-1,7},{-5,2,-4},{2,-1,-3}};
        assertEquals("YES",1,tester.equilibrium(3,arr));
    }
}
```

Test Cases "Run" window:



```
public class equilibriumTest {

    @Test
    public void test_n_Equal_ZERO() {
        equilibrium tester = new equilibrium();
        int [][] arr = {{4,1,7},{-2,4,-1},{1,-5,-3}};
        assertEquals( expected: -1,tester.equilibrium( n: 0,arr));
    }
```

Tests passed: 4 of 4 tests – 9 ms

```
equilibriumTest          9 ms
  test_n_Larger_than_100 9 ms
  test_n_Equal_ZERO      0 ms
  test_example_1         0 ms
  test_example_2         0 ms
```

```
"C:\Program Files\Java\jdk1.8.0_111\bin\java.exe" ...
WARNING:
******* WRONG NO. OF LINES ENTERED *******
NO. OF LINES (n) RANGE: (1 ≤ n ≤ 100)
WARNING:
******* WRONG NO. OF LINES ENTERED *******
NO. OF LINES (n) RANGE: (1 ≤ n ≤ 100)
NO
YES

Process finished with exit code 0
```

Tests passed: 4 (a minute ago)

Finally, all test cases are **succeeded**.

GitHub Repo:

My GitHub Repo contains solution & codes of this lab report problems.
Link is below:

https://github.com/Mennah-Ashraf/Testing-LabReport1.git

Important Note:

Recently, I have been transferred to be with Group 2 in this course labs. Thus, the deadline is been extended for me.