## Source code for a JUnit Test Case (with at least 2 tests)

We work on the lab 7 repo with an extra function that implements a solution to the two sum problem (given an array and a target value, find two elements from the array that sums up to the given value, or report that no such solutions exist).

### Source code to the implementation

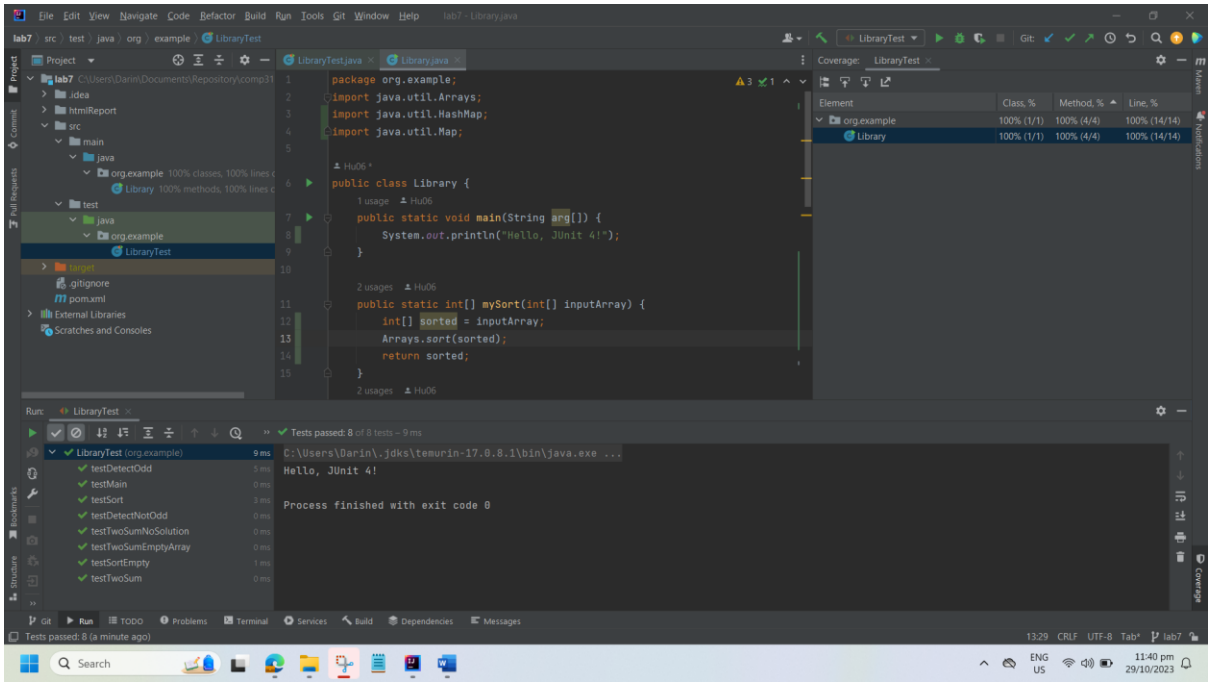
```
package org.example;
import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;

public class Library {
    public static void main(String arg[]) {
        System.out.println("Hello, JUnit 4!");
    }

    public static int[] mySort(int[] inputArray) {
        int[] sorted = inputArray;
        Arrays.sort(sorted);
        return sorted;
    }
    public static boolean detectOdd(int inputNumber) {
        if (inputNumber%2 == 1) {
            return true;
        }else {
            return false;
        }
    }

    public static int[] twoSum(int[] nums, int target) {
        Map<Integer, Integer> map = new HashMap<>();

        for (int i = 0; i < nums.length; i++) {
            int complement = target - nums[i];

            if (map.containsKey(complement)) {
                return new int[]{map.get(complement), i};
            }

            map.put(nums[i], i);
        }

        throw new IllegalArgumentException("No solution.");
    }
}
```

## Source code to the test cases

```
package org.example;

import static org.junit.Assert.*;
import org.junit.Test;
import java.util.Arrays;

public class LibraryTest {

    @Test
    public void testMain() {
        // Since the main method only prints a message, I don't know of any
way to test its correctness directly (unless its like intercepting stdout
but idk how)
        Library.main(new String[]{});
    }

    @Test
    public void testSort() {
        int[] inputArray = {5, 3, 1, 4, 2};
        int[] expected = {1, 2, 3, 4, 5};
        int[] sorted = Library.mySort(inputArray);
        assertArrayEquals(expected, sorted);
    }

    @Test
    public void testSortEmpty() {
        int[] inputArray = {};
        int[] expected = {};
        int[] sorted = Library.mySort(inputArray);
        assertArrayEquals(expected, sorted);
    }

    @Test
    public void testDetectOdd() {
        int inputNumber = 5;
        boolean expected = true;
        boolean isOdd = Library.detectOdd(inputNumber);
        assertEquals(expected, isOdd);
    }

    @Test
    public void testDetectNotOdd() {
        int inputNumber = 4;
        boolean expected = false;
        boolean isOdd = Library.detectOdd(inputNumber);
        assertEquals(expected, isOdd);
    }

    @Test
    public void testTwoSum() {
        int[] nums = {2, 7, 11, 15};
        int target = 9;
        int[] expected = {0, 1};
        int[] result = Library.twoSum(nums, target);
        assertArrayEquals(expected, result);
    }

    @Test
    public void testTwoSumNoSolution() {
        int[] nums = {2, 7, 11, 15};
```

```
        int target = 8;
        try {
            Library.twoSum(nums, target);
            fail("Expect IllegalArgumentException");
        } catch (IllegalArgumentException e) {
            assertEquals("No solution.", e.getMessage());
        }
    }

    @Test
    public void testTwoSumEmptyArray() {
        int[] nums = {};
        int target = 9;
        try {
            Library.twoSum(nums, target);
            fail("Expect IllegalArgumentException");
        } catch (IllegalArgumentException e) {
            assertEquals("No solution.", e.getMessage());
        }
    }
}
```

## Test report showing all the tests have been passed (100% successful)



# Coverage report showing the coverage (10% or higher)