

Code Challenge #1:

Coded in Java [Eclipse]

Main App:

```

1 package MaxSubArraySum;
2
3 import java.util.ArrayList;
4
5
6
7 public class App {
8
9     public static void main(String[] args) {
10
11         System.out.println("JDK 17 here...");
12
13         List<Integer> arrList = List.of(-2, -3, 4, -1, -2, 1, 5, -3);
14         System.out.println(displayList(arrList));
15         System.out.println(getMaxSumList(arrList));
16
17     }
18
19     public static List<Integer> displayList(List<Integer> nums) {
20         if (nums.size() == 0) {
21             return Collections.emptyList();
22         }
23
24         final int maxSum = getMaxSumList(nums);
25         int curSum = nums.get(0);
26
27         int start = 0;
28         int end = 0;
29
30     }
31 }

```

Problems | Javadoc | Declaration | Console | Debug

<terminated> App (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.0.2.jdk/Contents/Home/bin/java (Sep. 8, 2022, 10:48:07 p.m. – 10:48:09 p.m.) [pid: 8879]

JDK 17 here...

[4, -1, -2, 1, 5]

7

Test Cases:

```

1 List<Integer> list2 = Arrays.asList(-2, -3, 4, -1, -2, 1, 5, -3);
2 List<Integer> list3 = Arrays.asList(1, 5, 0, 1, 2, 2, 0, 3);
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18 @Before
19 public void setup() {
20     testApp = new AppTest();
21 }
22
23
24 @Test
25 public void testList1() {
26     int maxSum = a.getMaxSumList(list1);
27     int maxSumExpected = 6;
28     Assert.assertEquals(maxSumExpected, maxSum);
29 }
30
31 @Test
32 public void testList2() {
33     int maxSum = a.getMaxSumList(list2);
34     int expectedSum = 7;
35     Assert.assertEquals(expectedSum, maxSum);
36 }
37
38 @Test
39 public void testList3() {
40     int maxSum = a.getMaxSumList(list3);
41     int expectedSum = 14;
42     Assert.assertEquals(expectedSum, maxSum);
43 }
44
45 @Test
46 public void testList3NotEqual() {
47     int maxSum = a.getMaxSumList(list3);
48     int expectedSum = 104;
49     Assert.assertNotEquals(expectedSum, maxSum);
50 }
51
52
53
54
55
56
57
58
59
60

```

Package Explorer | JUnit

Runs: 4/4 Errors: 0 Failures: 0

Finished after 0.317 seconds

AppTest [Runner: JUnit 5] (0.052 s)

testList1() (0.001 s)

testList2() (0.002 s)

testList3() (0.002 s)

testList3NotEqual() (0.041 s)

Failure Trace

```

package MaxSubArraySum;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class App {

    public static void main(String[] args) {

        System.out.println("JDK 17 here...");

        List<Integer> arrList = List.of(-2, -3, 4, -1, -2, 1, 5, -3);
        System.out.println(displayList(arrList));
        System.out.println(getMaxSumList(arrList));

    }

    public static List<Integer> displayList(List<Integer> nums) {
        if (nums.size() == 0) {
            return Collections.emptyList();
        }

        final int maxSum = getMaxSumList(nums);
        int curSum = nums.get(0);

        int start = 0;
        int end = 0;

        for (int i = 1; i < nums.size(); i++) {
            if (curSum == maxSum) {
                end = i;
                break; // maximus sub-array was found
            }

            if (nums.get(i) > curSum + nums.get(i)) {
                start = i;
                curSum = nums.get(i);
            } else {
                curSum += nums.get(i);
            }
        }
        return new ArrayList<>(nums.subList(start, end));
    }

    // using Kadane's algorithm

```

```

    public static int getMaxSumList(List<Integer> nums) {
        int max = Integer.MIN_VALUE;
        int sum = nums.get(0);
        for (int i = 1; i < nums.size(); i++) {
            sum = Math.max(nums.get(i), sum + nums.get(i));
            max = Math.max(max, sum);
        }
        return max;
    }
}

```

Test.java

```

package MaxSubArraySum;

import java.util.Arrays;
import java.util.List;
import org.junit.Assert;
import org.junit.Before;
import org.junit.jupiter.api.Test;

class AppTest {

    private AppTest testApp;

    App a = new App();
    List<Integer> list1 = Arrays.asList(-2, 1, -3, 4, -1, 2, 1, -5, 4);
    List<Integer> list2 = Arrays.asList(-2, -3, 4, -1, -2, 1, 5, -3);
    List<Integer> list3 = Arrays.asList(1, 5, 0, 1, 2, 2, 0, 3);

    @Before
    public void setup() {
        testApp = new AppTest();
    }

    @Test
    public void testList1() {
        int maxSum = a.getMaxSumList(list1);
        int maxSumExpected = 6;
        Assert.assertEquals(maxSumExpected, maxSum);
    }
}

```

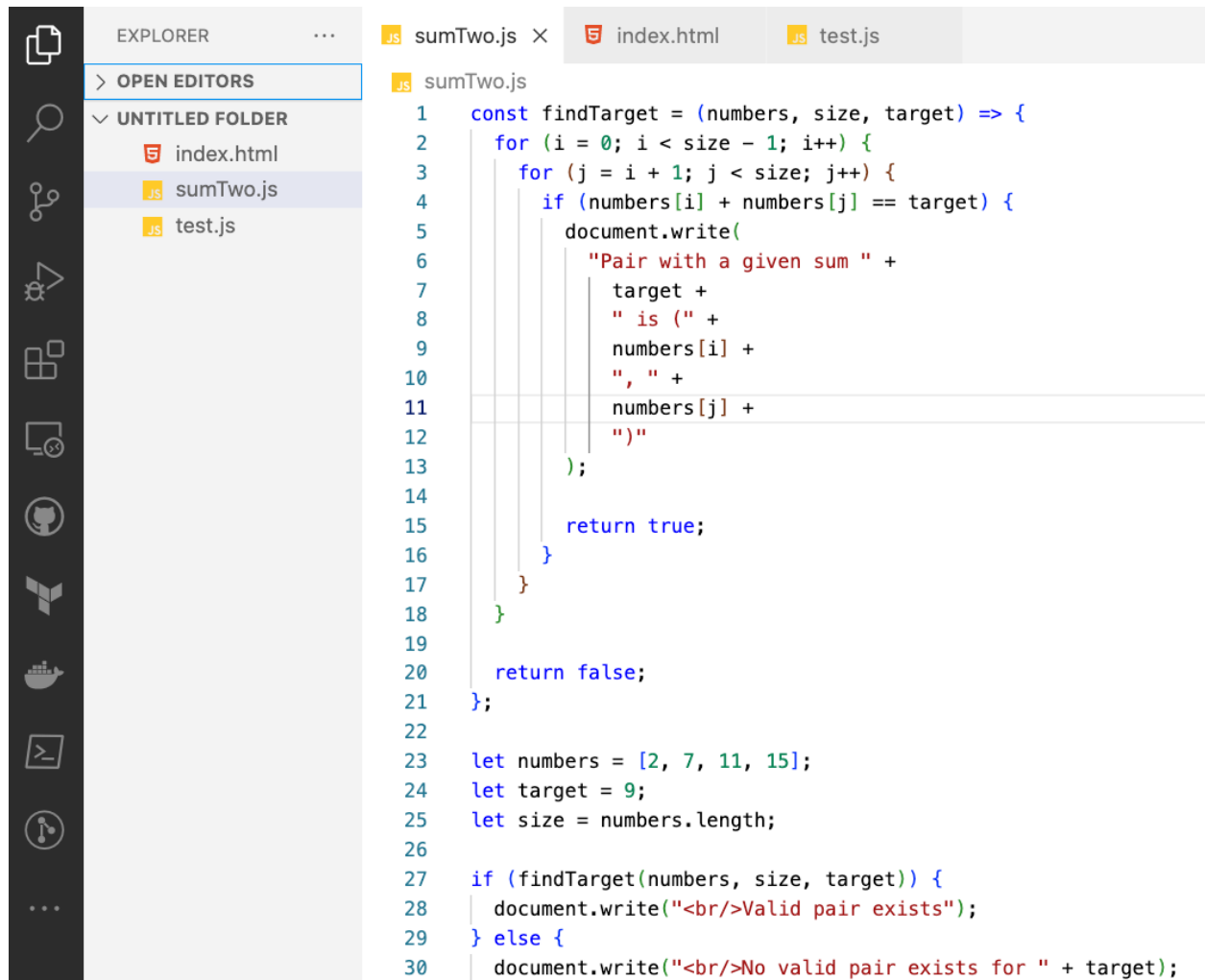
```
@Test
public void testList2() {
    int maxSum = a.getMaxSumList(list2);
    int expectedSum = 7;
    Assert.assertEquals(expectedSum, maxSum);
}

@Test
public void testList3() {
    int maxSum = a.getMaxSumList(list3);
    int expectedSum = 14;
    Assert.assertEquals(expectedSum, maxSum);
}

@Test
public void testList3NotEqual() {
    int maxSum = a.getMaxSumList(list3);
    int expectedSum = 104;
    Assert.assertNotEquals(expectedSum, maxSum);
}
}
```

Code Challenge #2:

JavaScript [Visual Studio]



```
1  const findTarget = (numbers, size, target) => {
2    for (i = 0; i < size - 1; i++) {
3      for (j = i + 1; j < size; j++) {
4        if (numbers[i] + numbers[j] == target) {
5          document.write(
6            "Pair with a given sum " +
7            target +
8            " is (" +
9            numbers[i] +
10           ", " +
11           numbers[j] +
12           ")"
13         );
14       }
15       return true;
16     }
17   }
18 }
19
20 return false;
21 };
22
23 let numbers = [2, 7, 11, 15];
24 let target = 9;
25 let size = numbers.length;
26
27 if (findTarget(numbers, size, target)) {
28   document.write("<br/>Valid pair exists");
29 } else {
30   document.write("<br/>No valid pair exists for " + target);
```

OutPut

Pair with a given sum 9 is (2, 7)Pair with a given sum -2 is (-3, 1)Pair with a given sum 16 is (6, 10)

Index.html

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Mocha Tests</title>
  <link href="https://cdn.rawgit.com/mochajs/mocha/2.2.5/mocha.css" rel="stylesheet"/>
</head>
<body>
  <div id="mocha"></div>
  <script src="https://cdn.rawgit.com/Automattic/expect.js/0.3.1/index.js"></script>
  <script src="https://cdn.rawgit.com/chaijs/chai/3.5.0/chai.js"></script>
  <script src="https://cdn.rawgit.com/mochajs/mocha/2.2.5/mocha.js"></script>

  <script src="./sumTwo.js"></script>
  <script>
    const mocha = window.mocha;
    mocha.setup('bdd');
  </script>
  <script src="./test.js"></script>
  <script>
    mocha.checkLeaks();
    mocha.run();
  </script>
</body>
</html>

```

SumTwo.js

```

const findTarget = (numbers, size, target) => {
  for (i = 0; i < size - 1; i++) {
    for (j = i + 1; j < size; j++) {
      if (numbers[i] + numbers[j] == target) {
        document.write(
          "Pair with a given sum " +
            target +
            " is (" +
              numbers[i] +
              ", " +
              numbers[j] +
            ")"
        );
      }
    }
  }
  return true;
}

```

```

    return false;
};

let numbers = [2, 7, 11, 15];
let target = 9;
let size = numbers.length;

if (findTarget(numbers, size, target)) {
    document.write("<br/>Valid pair exists");
} else {
    document.write("<br/>No valid pair exists for " + target);
}

```

Test.js

```

const chai = window.chai;
const expect = chai.expect;

describe("SumTwoFound", () => {
    it("Should be true, if target found", function () {
        let numbers = [2, 7, 11, 15];
        let target = 9;
        let size = numbers.length;
        expect(findTarget(numbers, size, target)).to.be.true;
    });
    it("Should be true, if target found", function () {
        let numbers = [0, -1, 2, -3, 1];
        let target = -2;
        let size = numbers.length;
        expect(findTarget(numbers, size, target)).to.be.true;
    });
    it("Should be true, if target found", function () {
        let numbers = [1, 4, 45, 6, 10, -8];
        let target = 16;
        let size = numbers.length;
        expect(findTarget(numbers, size, target)).to.be.true;
    });
    it("Should be false, if target not found", function () {
        let numbers = [1, 4, 45, 6, 10, -8];
        let target = 106;
        let size = numbers.length;
        expect(findTarget(numbers, size, target)).to.be.false;
    });
});

```