

**Program: Mainstream**

***Course Name: Algorithms***

***Name: Yousef Amer Awadallah***

***ID: 1901524***

***Department: CSE***

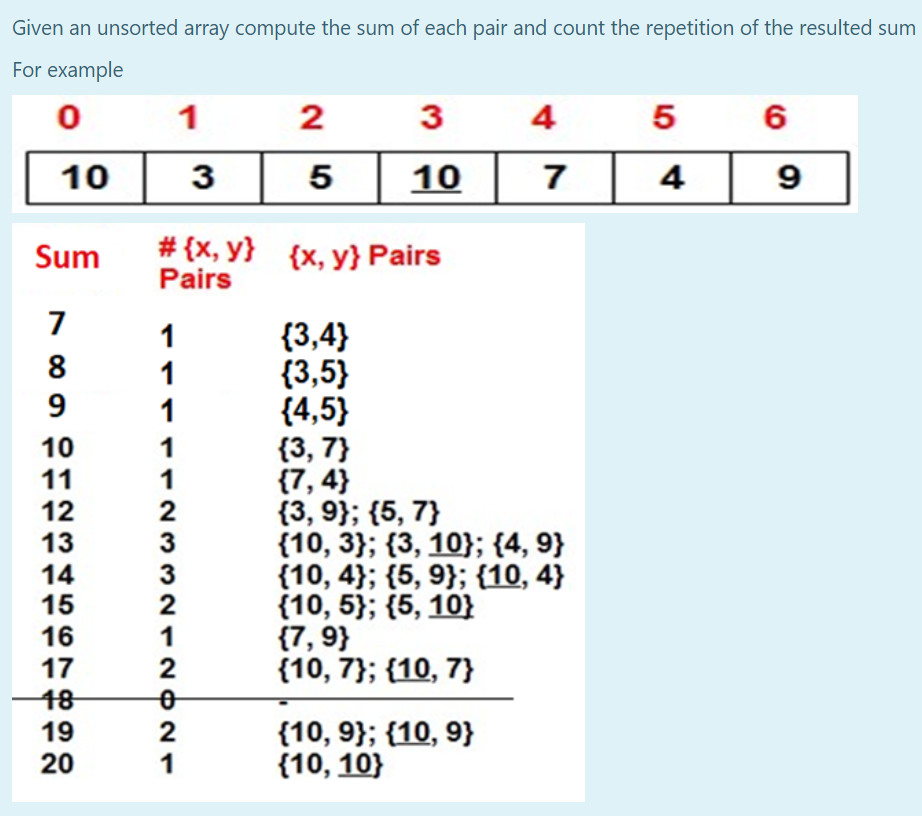
**Ain Shams University**

**Faculty of Engineering**

**Spring Semester – 2022**

Lab Test

**Question :**



**Java Code:**

import java.util.Scanner;

public class PairSumCount {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of elements: ");

        int n = scanner.nextInt();

        int[] array = new int[n];

        System.out.println("Enter the elements:");

        for (int i = 0; i < n; i++) {

            array[i] = scanner.nextInt();

        }

        computePairSumCount(array);

    }

    public static void computePairSumCount(int[] array) {

        int[] sumCountArray = new int[2 \* getMaxValue(array) + 1];

        for (int i = 0; i < array.length; i++) {

            for (int j = i + 1; j < array.length; j++) {

                int sum = array[i] + array[j];

                sumCountArray[sum]++;

            }

        }

        System.out.println("Sum\tCount\tPairs");

        for (int sum = 0; sum < sumCountArray.length; sum++) {

            int count = sumCountArray[sum];

            if (count > 0) {

                System.out.print(sum + "\t" + count + "\t");

                printPairs(array, sum);

                System.out.println();

            }

        }

    }

    public static void printPairs(int[] array, int targetSum) {

        for (int i = 0; i < array.length; i++) {

            for (int j = i + 1; j < array.length; j++) {

                if (array[i] + array[j] == targetSum) {

                    System.out.print("(" + array[i] + "," + array[j] + ") ");

                }

            }

        }

    }

    public static int getMaxValue(int[] array) {

        int max = Integer.MIN\_VALUE;

        for (int value : array) {

            if (value > max) {

                max = value;

            }

        }

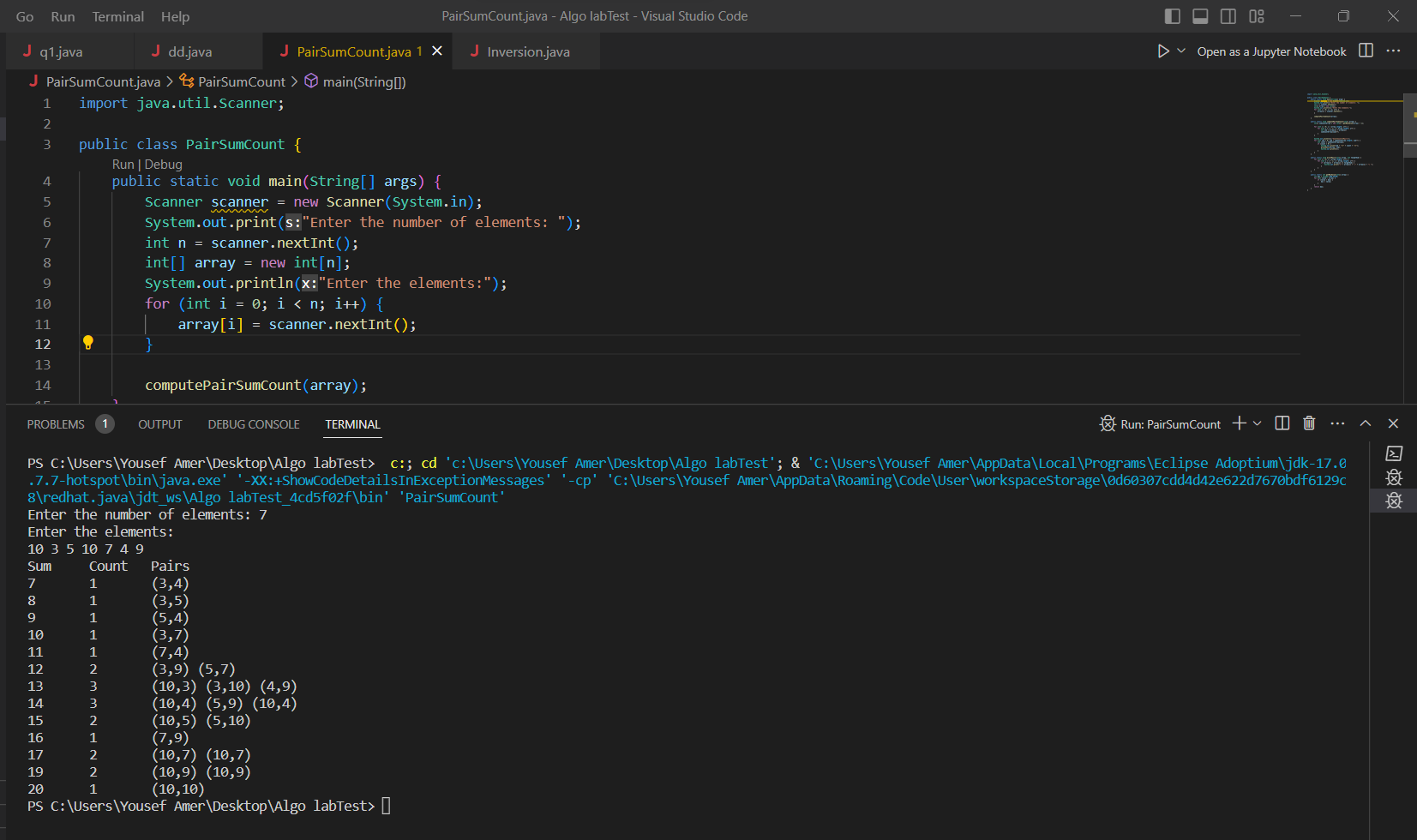
        return max;

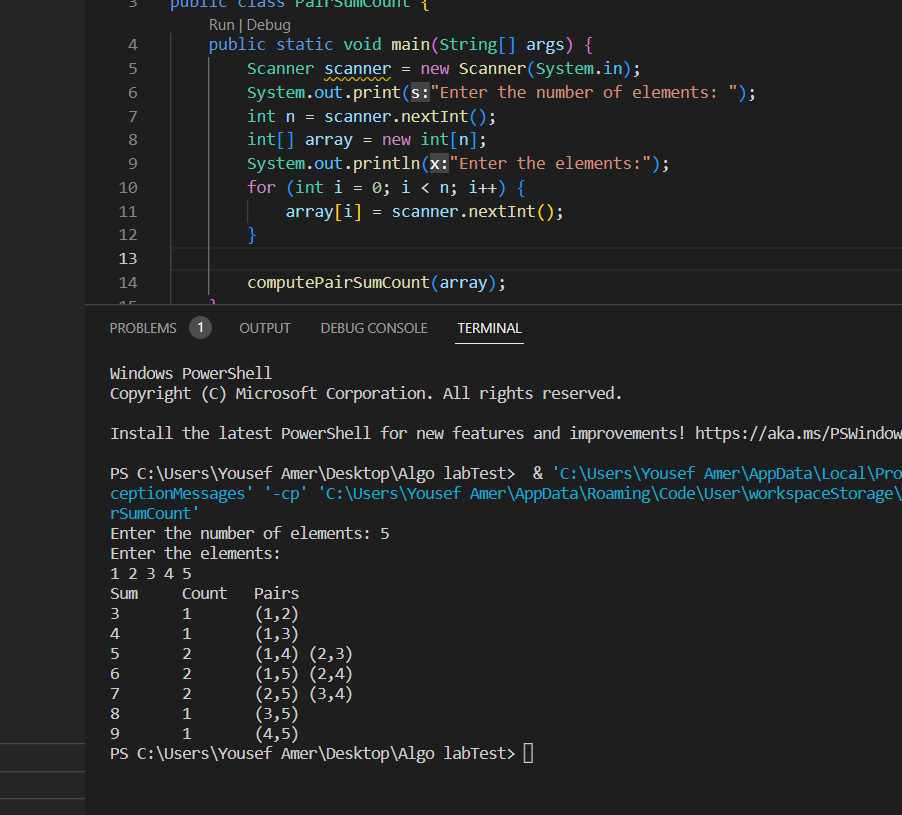
    }

}

**Output screenshots:**

Test1: The same example given in the question



Test2:

**Algorithm of the code:**

1. Take input from the user for the number of elements, n (number of elements).

2. Create an array of size n to store the elements.

3. Take input from the user for each element of the array.

4. Initialize an integer array, `sumCountArray`, of size 2 \* getMaxValue(array) + 1 with all elements set to 0.

5. Iterate over each element of the array using two nested loops:

- The outer loop iterates from 0 to n-1.

- The inner loop iterates from i+1 to n-1.

6. For each pair of elements at indices i and j, calculate the sum as array[i] + array[j].

7. Increment the count of the corresponding sum in `sumCountArray` by 1.

8. After the nested loops, iterate over `sumCountArray`:

- For each index sum, retrieve the count of pairs with the sum.

- If the count is greater than 0, print the sum, count, and the pairs by calling the `printPairs` function.

9. In the `printPairs` function, iterate over each pair of elements using two nested loops:

- The outer loop iterates from 0 to n-1.

- The inner loop iterates from i+1 to n-1.

- Check if the sum of array[i] and array[j] equals the targetSum.

- If it matches, print the pair (array[i], array[j]).

10. Finally, implement the `getMaxValue` function to iterate over the elements of the array and find the maximum value.

**Complexity**

1. Taking input: O(n), where n is the number of elements in the array.

2. Initializing `sumCountArray`: O(n), as it depends on the size of the input array.

3. Nested Loops: O(n^2), as the nested loops iterate over all possible pairs.

4. Counting Sums: O(n^2), since each pair of elements contributes to the sum count.

5. Printing Output: O(maxValue), where maxValue is the maximum value in the array.

6. `printPairs` Function: O(n^2), as it iterates over all possible pairs.

7. `getMaxValue` Function: O(n), as it iterates over the elements of the array to find the maximum value.

Overall, the time complexity is **O(n^2)** due to the nested loops, and the space complexity is **O(n)** for the input array and O(maxValue) for the `sumCountArray`.