# LAB # 04

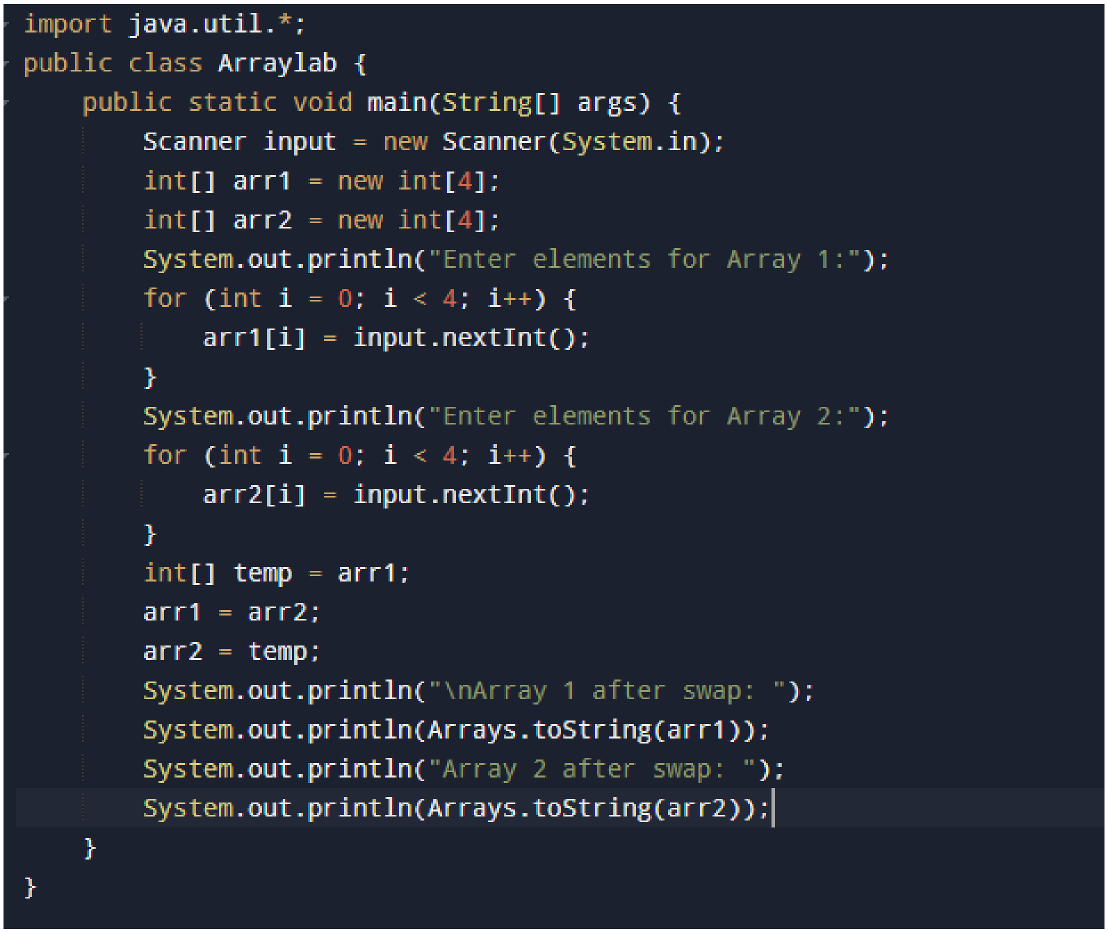
# ARRAYS IN JAVA

**OBJECTIVE: To understand arrays and its memory allocation.**

## LAB TASKS

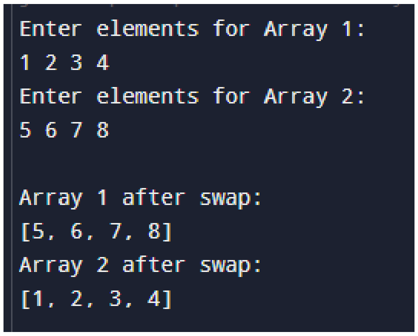
1. Write a program that takes two arrays of size 4 and swap the elements of those arrays

Lab task#1:



```java
import java.util.*;
public class Arraylab {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int[] arr1 = new int[4];
        int[] arr2 = new int[4];
        System.out.println("Enter elements for Array 1:");
        for (int i = 0; i < 4; i++) {
            arr1[i] = input.nextInt();
        }
        System.out.println("Enter elements for Array 2:");
        for (int i = 0; i < 4; i++) {
            arr2[i] = input.nextInt();
        }
        int[] temp = arr1;
        arr1 = arr2;
        arr2 = temp;
        System.out.println("\nArray 1 after swap: ");
        System.out.println(Arrays.toString(arr1));
        System.out.println("Array 2 after swap: ");
        System.out.println(Arrays.toString(arr2));
    }
}
```

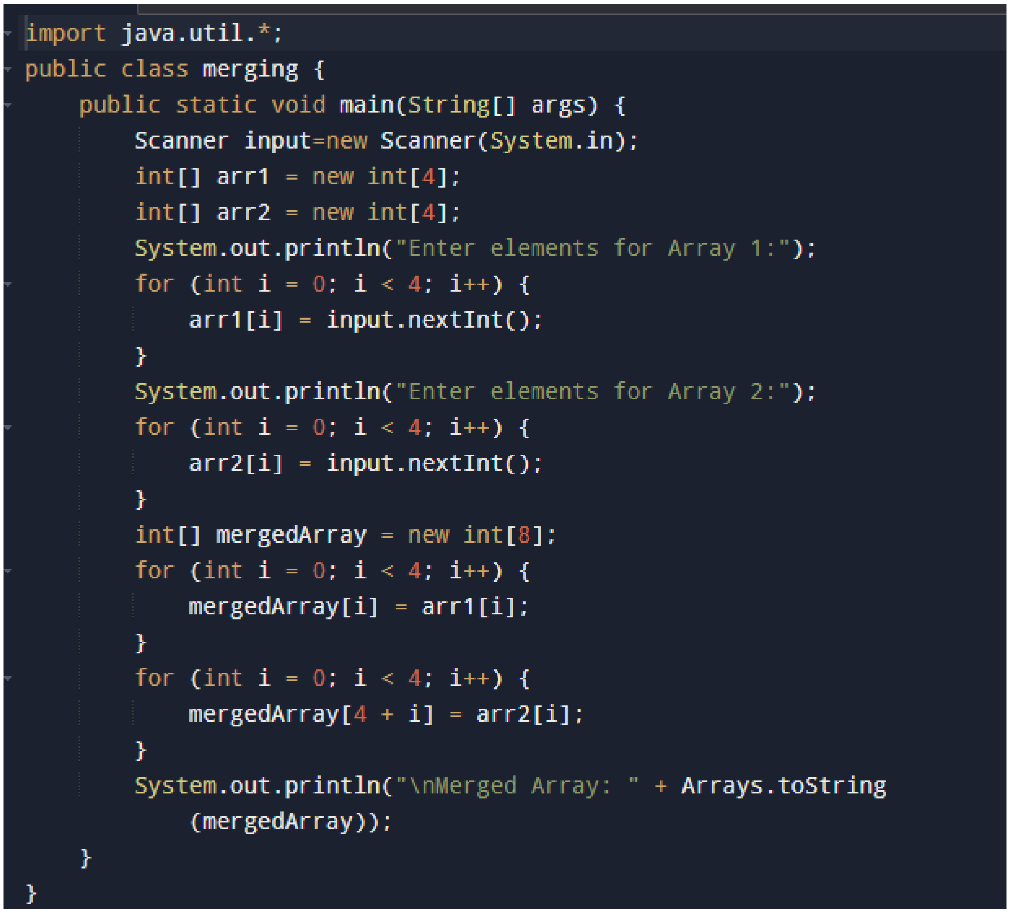
Output:



```
Enter elements for Array 1:
1 2 3 4
Enter elements for Array 2:
5 6 7 8

Array 1 after swap:
[5, 6, 7, 8]
Array 2 after swap:
[1, 2, 3, 4]
```

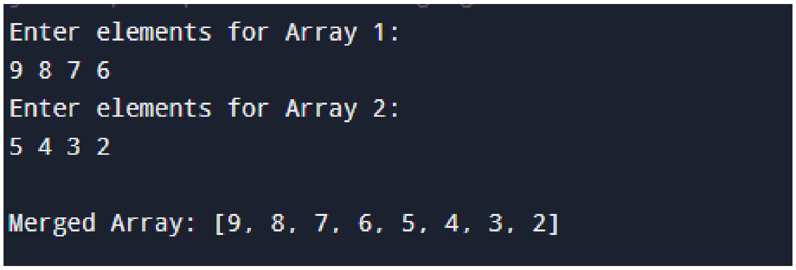
task#2:

2. Add a method in the class that takes array and merge it with the existing one.



```java
import java.util.*;
public class merging {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        int[] arr1 = new int[4];
        int[] arr2 = new int[4];
        System.out.println("Enter elements for Array 1:");
        for (int i = 0; i < 4; i++) {
            arr1[i] = input.nextInt();
        }
        System.out.println("Enter elements for Array 2:");
        for (int i = 0; i < 4; i++) {
            arr2[i] = input.nextInt();
        }
        int[] mergedArray = new int[8];
        for (int i = 0; i < 4; i++) {
            mergedArray[i] = arr1[i];
        }
        for (int i = 0; i < 4; i++) {
            mergedArray[4 + i] = arr2[i];
        }
        System.out.println("\nMerged Array: " + Arrays.toString
            (mergedArray));
    }
}
```

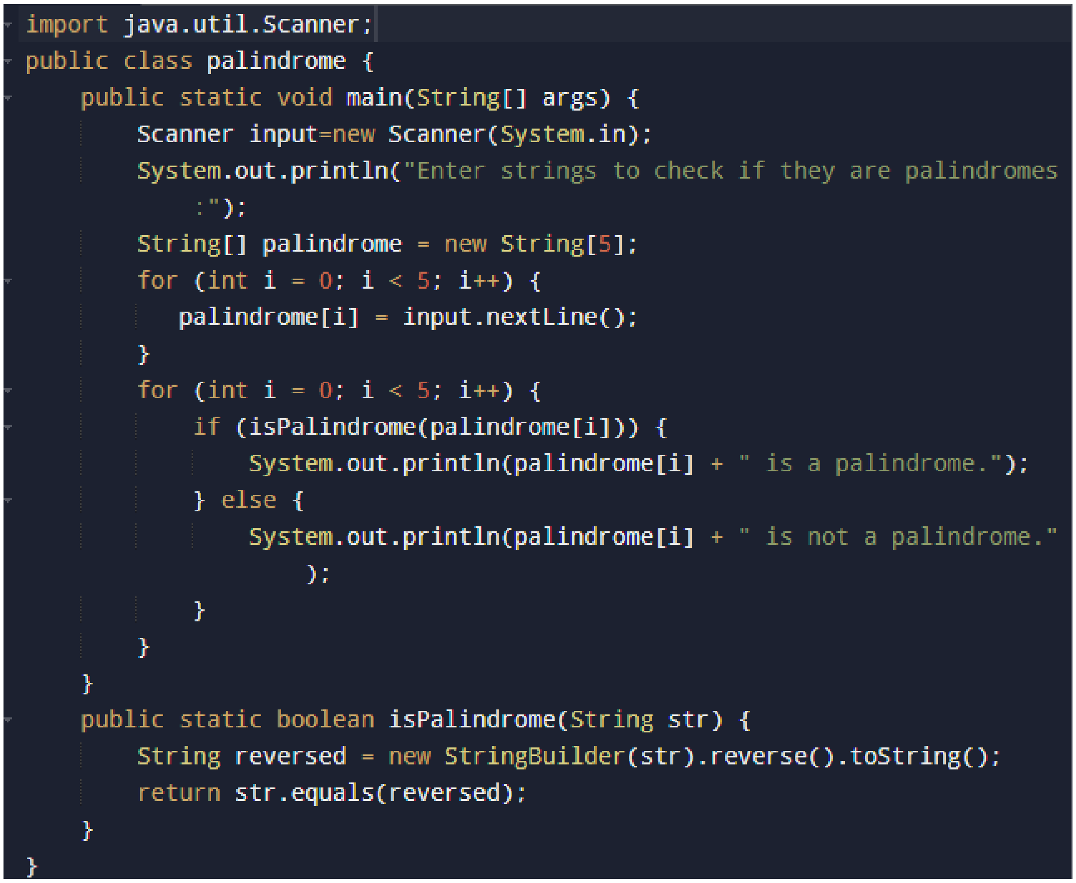
Output:



```
Enter elements for Array 1:
9 8 7 6
Enter elements for Array 2:
5 4 3 2

Merged Array: [9, 8, 7, 6, 5, 4, 3, 2]
```

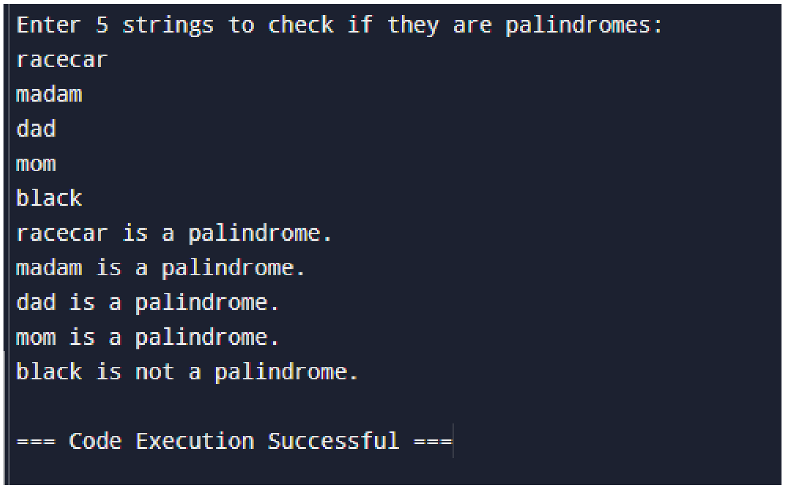
Lab task#3:

3. In a JAVA program, take an array of type string and then check whether the strings are palindrome or not.



```
import java.util.Scanner;
public class palindrome {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        System.out.println("Enter strings to check if they are palindromes
            :");
        String[] palindrome = new String[5];
        for (int i = 0; i < 5; i++) {
           palindrome[i] = input.nextLine();
        }
        for (int i = 0; i < 5; i++) {
            if (isPalindrome(palindrome[i])) {
                System.out.println(palindrome[i] + " is a palindrome.");
            } else {
                System.out.println(palindrome[i] + " is not a palindrome."
                    );
            }
        }
    }
    public static boolean isPalindrome(String str) {
        String reversed = new StringBuilder(str).reverse().toString();
        return str.equals(reversed);
    }
}
```

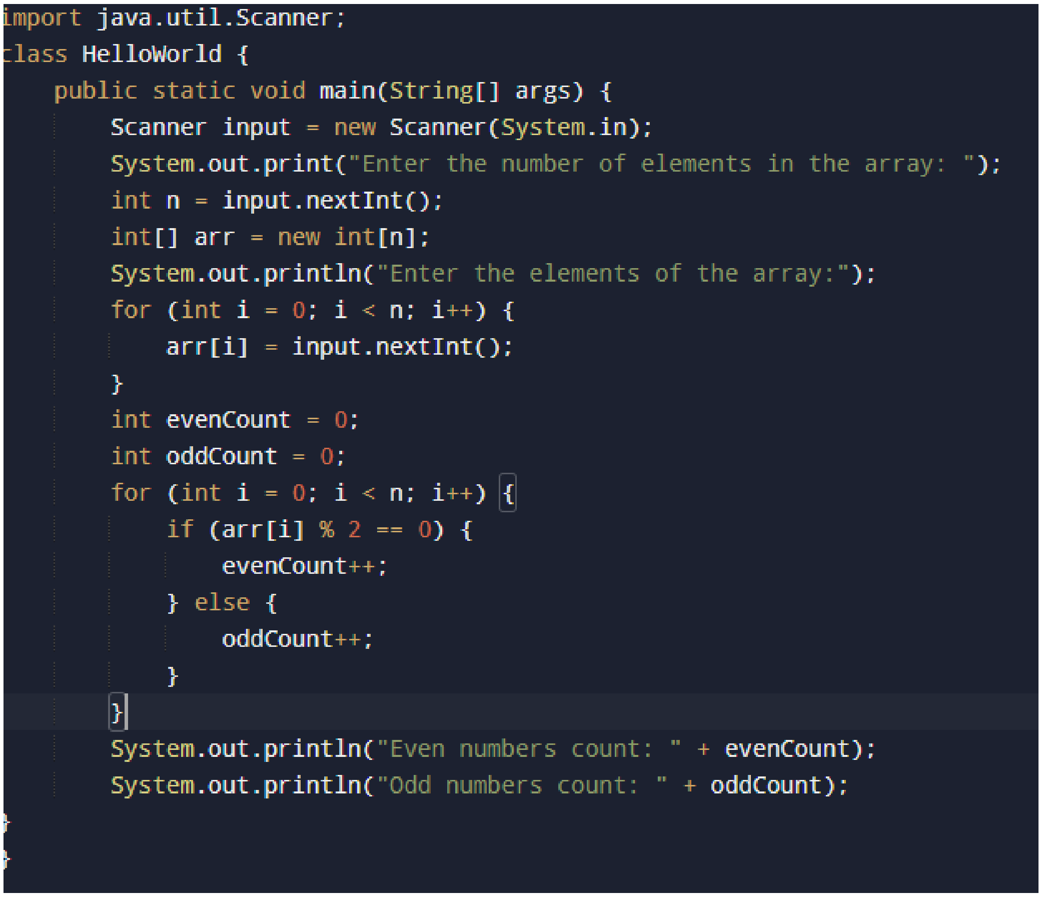
Output:



```
Enter 5 strings to check if they are palindromes:
racecar
madam
dad
mom
black
racecar is a palindrome.
madam is a palindrome.
dad is a palindrome.
mom is a palindrome.
black is not a palindrome.

=== Code Execution Successful ===
```

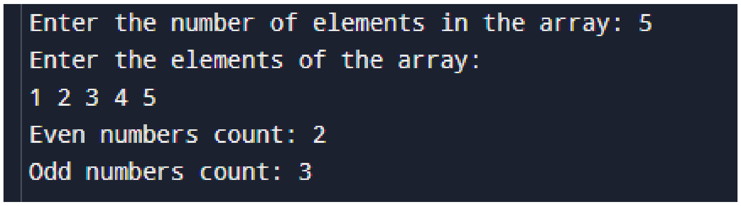
Lab task#4:

4. Given an array of integers, count how many numbers are even and how many are odd.



```
import java.util.Scanner;
class HelloWorld {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = input.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            arr[i] = input.nextInt();
        }
        int evenCount = 0;
        int oddCount = 0;
        for (int i = 0; i < n; i++) {
            if (arr[i] % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }
        System.out.println("Even numbers count: " + evenCount);
        System.out.println("Odd numbers count: " + oddCount);
}
}
```

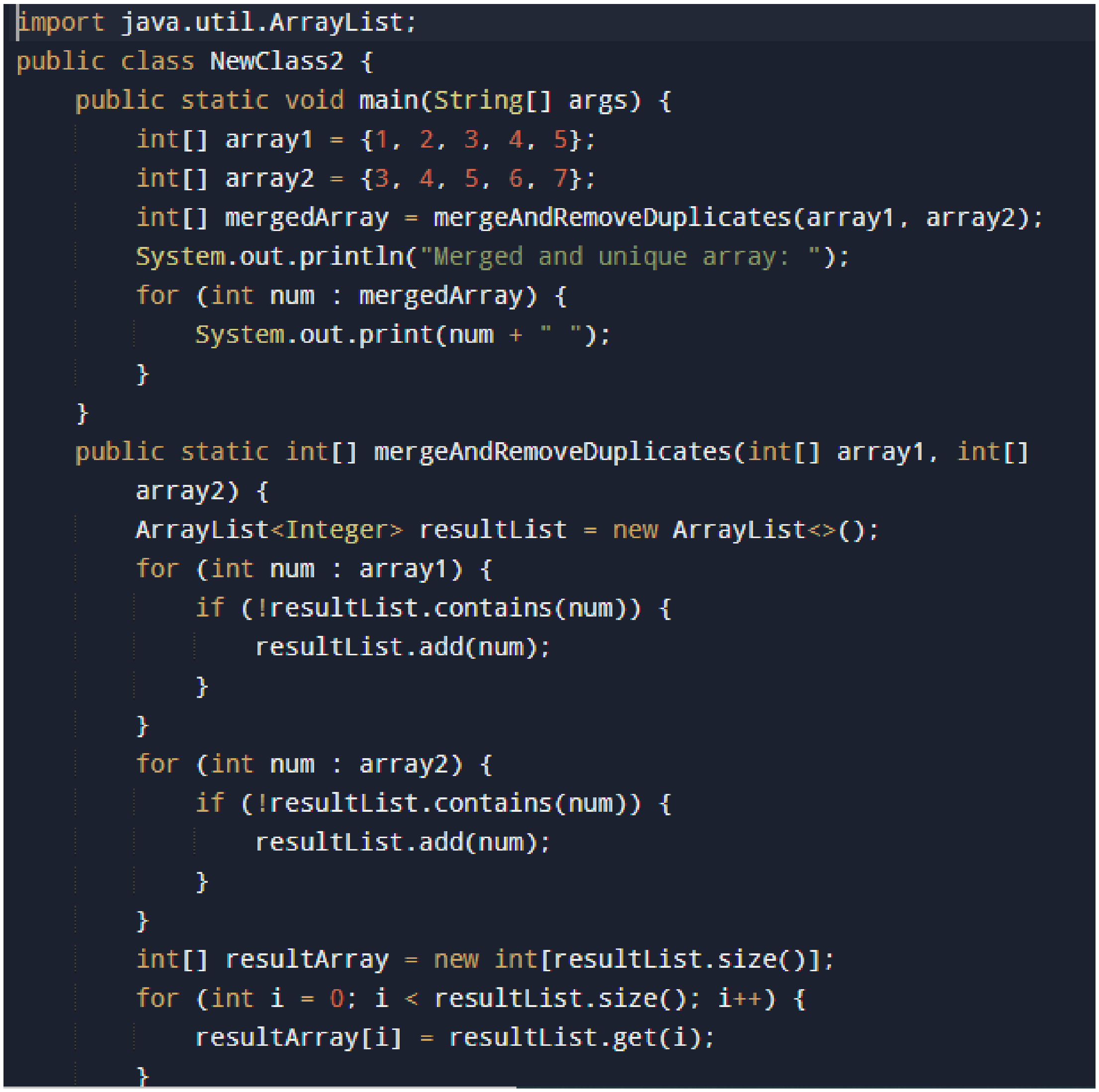
Output:



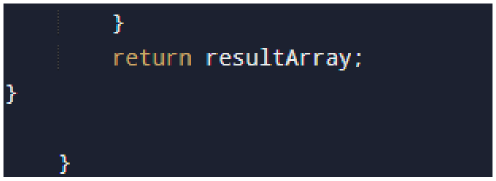
```
Enter the number of elements in the array: 5
Enter the elements of the array:
1 2 3 4 5
Even numbers count: 2
Odd numbers count: 3
```

Lab task#5:

5. Given two integer arrays, merge them and remove any duplicate values from the resulting array.

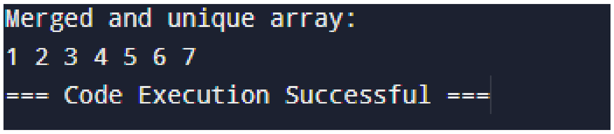


```
import java.util.ArrayList;
public class NewClass2 {
    public static void main(String[] args) {
        int[] array1 = {1, 2, 3, 4, 5};
        int[] array2 = {3, 4, 5, 6, 7};
        int[] mergedArray = mergeAndRemoveDuplicates(array1, array2);
        System.out.println("Merged and unique array: ");
        for (int num : mergedArray) {
            System.out.print(num + " ");
        }
    }
    public static int[] mergeAndRemoveDuplicates(int[] array1, int[]
        array2) {
        ArrayList<Integer> resultList = new ArrayList<>();
        for (int num : array1) {
            if (!resultList.contains(num)) {
                resultList.add(num);
            }
        }
        for (int num : array2) {
            if (!resultList.contains(num)) {
                resultList.add(num);
            }
        }
        int[] resultArray = new int[resultList.size()];
        for (int i = 0; i < resultList.size(); i++) {
            resultArray[i] = resultList.get(i);
        }
```



```
        }
        return resultArray;
}

    }
```

Output:

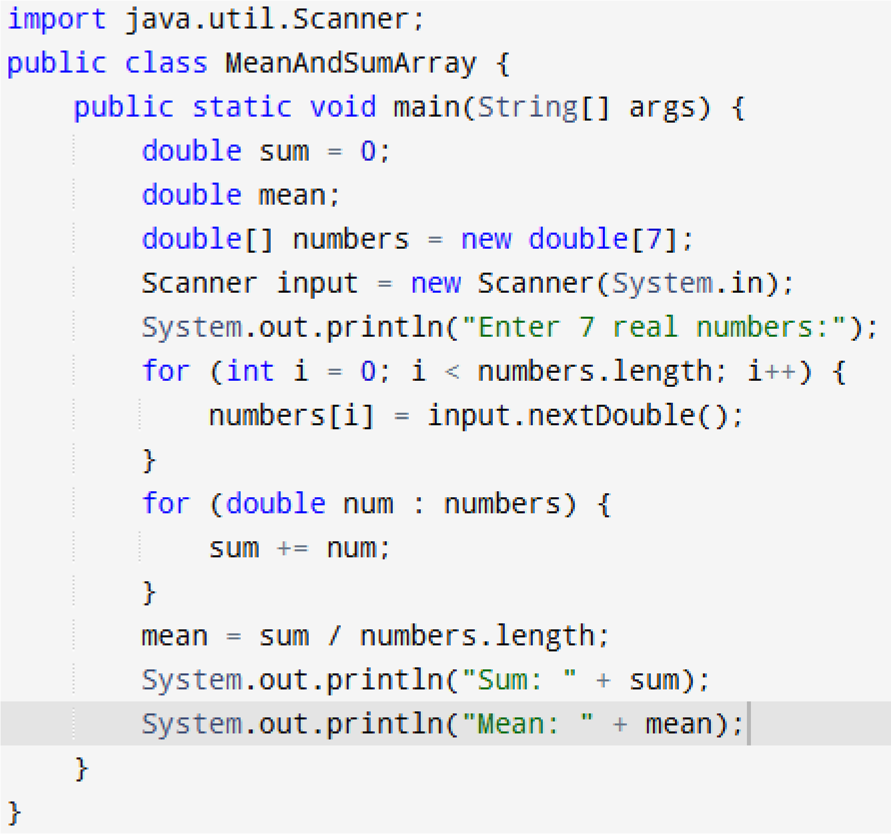


```
Merged and unique array:
1 2 3 4 5 6 7
=== Code Execution Successful ===
```

# Home tasks:

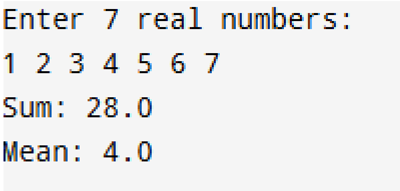
Task#1:

1. Write a program that takes an array of Real numbers having size 7 and calculate the sum and mean of all the elements. Also depict the memory management of this task.



```java
import java.util.Scanner;
public class MeanAndSumArray {
    public static void main(String[] args) {
        double sum = 0;
        double mean;
        double[] numbers = new double[7];
        Scanner input = new Scanner(System.in);
        System.out.println("Enter 7 real numbers:");
        for (int i = 0; i < numbers.length; i++) {
            numbers[i] = input.nextDouble();
        }
        for (double num : numbers) {
            sum += num;
        }
        mean = sum / numbers.length;
        System.out.println("Sum: " + sum);
        System.out.println("Mean: " + mean);
    }
}
```

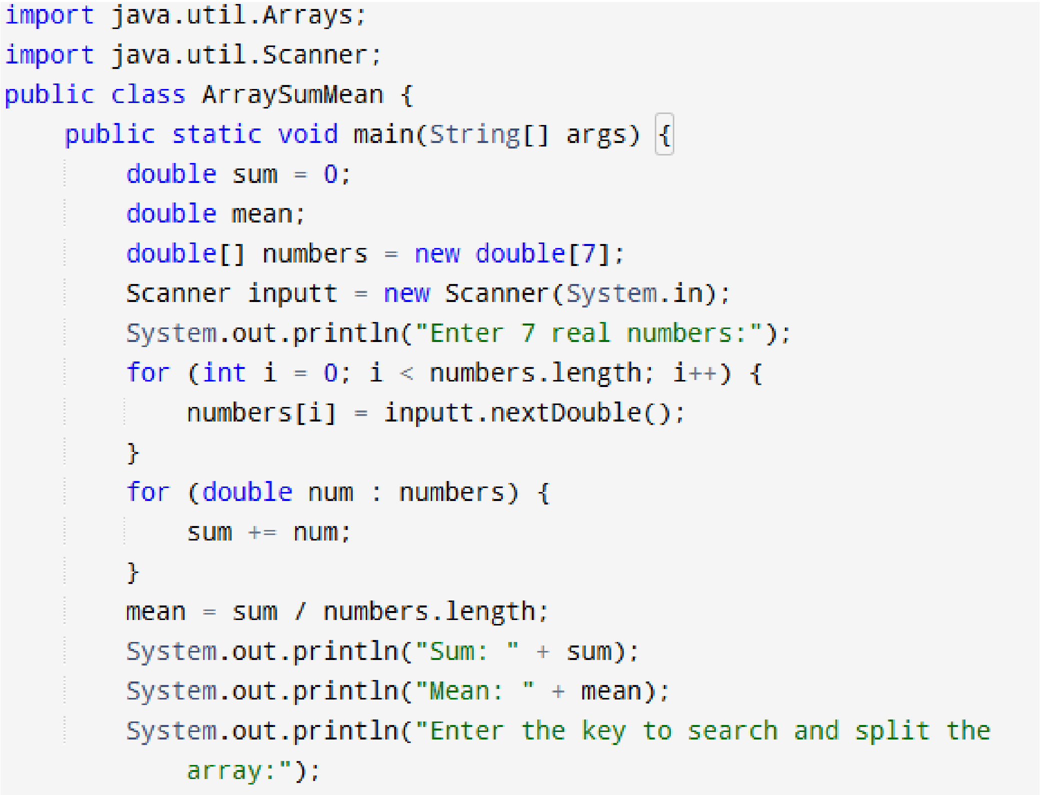
Output:



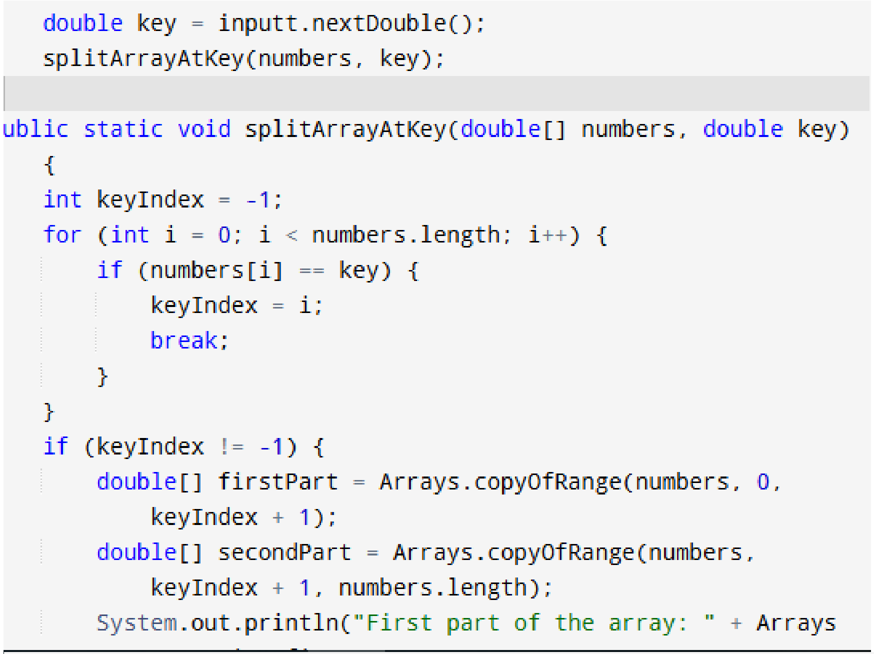
```
Enter 7 real numbers:
1 2 3 4 5 6 7
Sum: 28.0
Mean: 4.0
```

Task#2:

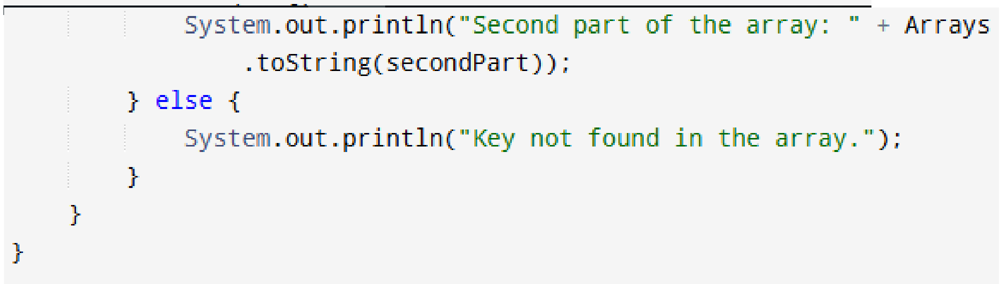
2. Add a method in the same class that splits the existing array into two. The method should search a key in array and if found splits the array from that index of the key



```
import java.util.Arrays;
import java.util.Scanner;
public class ArraySumMean {
    public static void main(String[] args) {
        double sum = 0;
        double mean;
        double[] numbers = new double[7];
        Scanner inputt = new Scanner(System.in);
        System.out.println("Enter 7 real numbers:");
        for (int i = 0; i < numbers.length; i++) {
            numbers[i] = inputt.nextDouble();
        }
        for (double num : numbers) {
            sum += num;
        }
        mean = sum / numbers.length;
        System.out.println("Sum: " + sum);
        System.out.println("Mean: " + mean);
        System.out.println("Enter the key to search and split the
            array:");
```

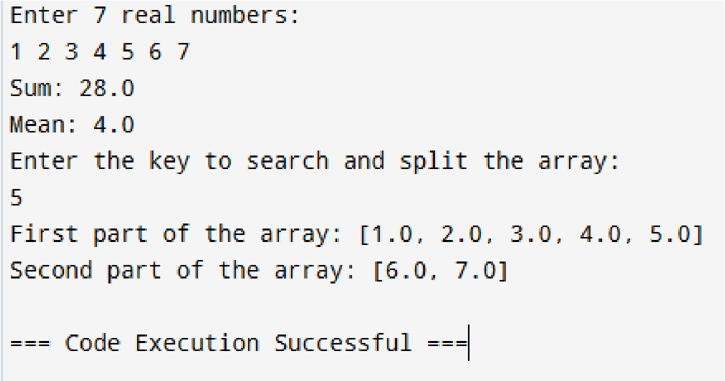


```
   double key = inputt.nextDouble();
   splitArrayAtKey(numbers, key);

ublic static void splitArrayAtKey(double[] numbers, double key)
   {
   int keyIndex = -1;
   for (int i = 0; i < numbers.length; i++) {
       if (numbers[i] == key) {
           keyIndex = i;
           break;
       }
   }
   if (keyIndex != -1) {
       double[] firstPart = Arrays.copyOfRange(numbers, 0,
           keyIndex + 1);
       double[] secondPart = Arrays.copyOfRange(numbers,
           keyIndex + 1, numbers.length);
       System.out.println("First part of the array: " + Arrays
```



```
            System.out.println("Second part of the array: " + Arrays
                .toString(secondPart));
        } else {
            System.out.println("Key not found in the array.");
        }
    }
}
```

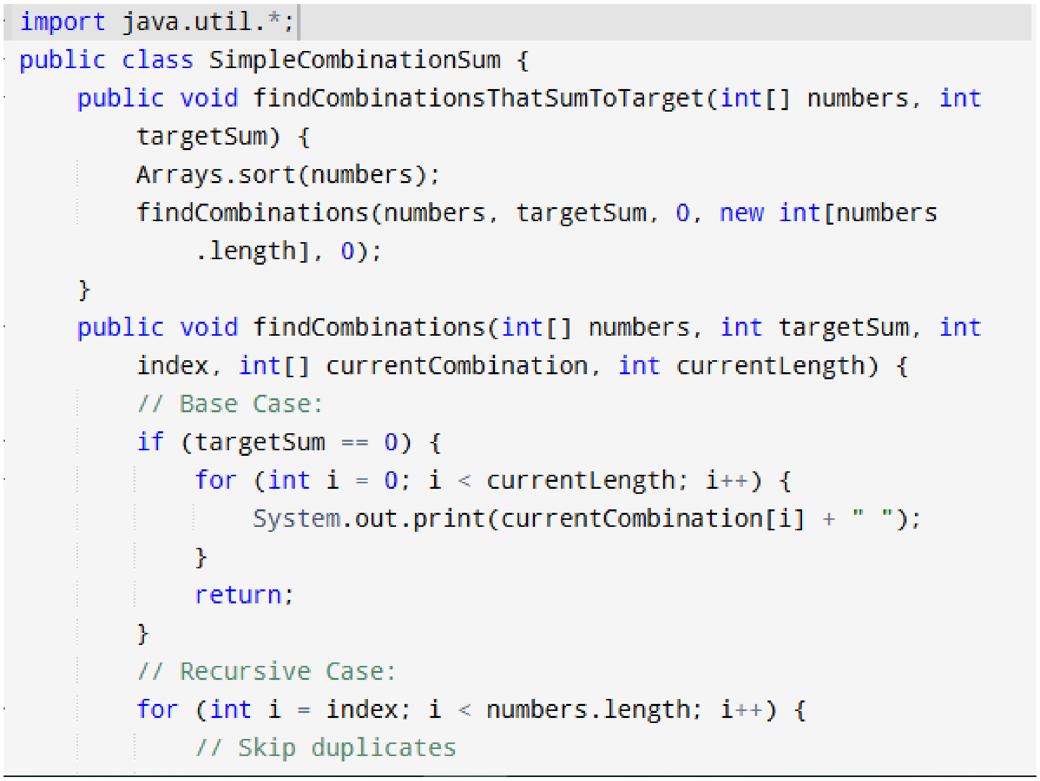
Output:



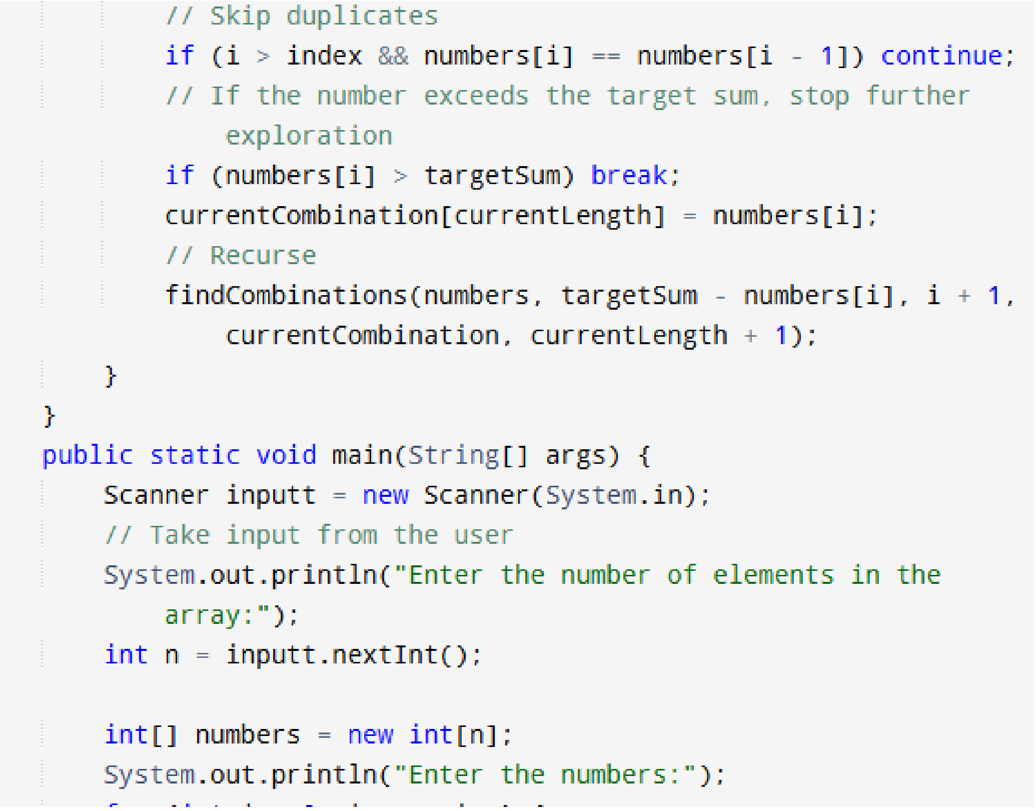
```
Enter 7 real numbers:
1 2 3 4 5 6 7
Sum: 28.0
Mean: 4.0
Enter the key to search and split the array:
5
First part of the array: [1.0, 2.0, 3.0, 4.0, 5.0]
Second part of the array: [6.0, 7.0]

=== Code Execution Successful ===
```

Task#3:

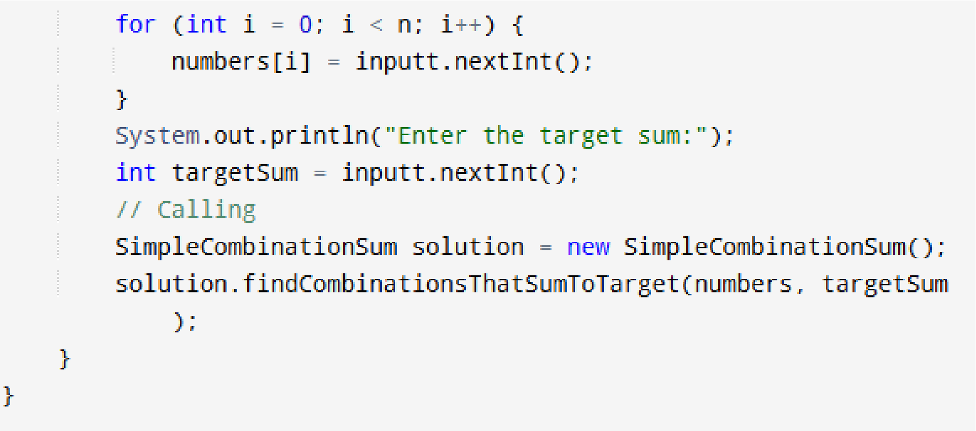
3. Given an array of distinct integers and a target integer, return all unique combinations of numbers that add up to the target. Each number can be used only once in the combination.



```
import java.util.*;
public class SimpleCombinationSum {
    public void findCombinationsThatSumToTarget(int[] numbers, int
        targetSum) {
        Arrays.sort(numbers);
        findCombinations(numbers, targetSum, 0, new int[numbers
            .length], 0);
    }
    public void findCombinations(int[] numbers, int targetSum, int
        index, int[] currentCombination, int currentLength) {
        // Base Case:
        if (targetSum == 0) {
            for (int i = 0; i < currentLength; i++) {
                System.out.print(currentCombination[i] + " ");
            }
            return;
        }
        // Recursive Case:
        for (int i = index; i < numbers.length; i++) {
            // Skip duplicates
```

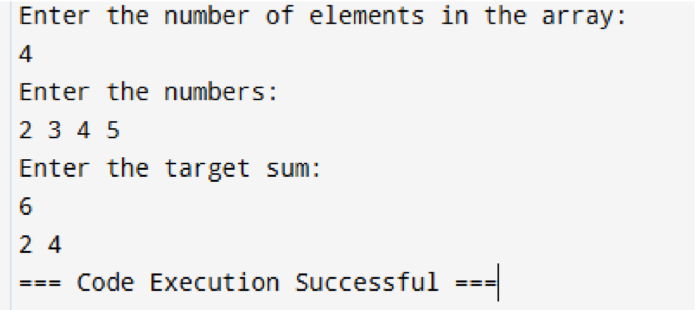


```
            // Skip duplicates
            if (i > index && numbers[i] == numbers[i - 1]) continue;
            // If the number exceeds the target sum, stop further
                exploration
            if (numbers[i] > targetSum) break;
            currentCombination[currentLength] = numbers[i];
            // Recurse
            findCombinations(numbers, targetSum - numbers[i], i + 1,
                currentCombination, currentLength + 1);
        }
    }
    public static void main(String[] args) {
        Scanner inputt = new Scanner(System.in);
        // Take input from the user
        System.out.println("Enter the number of elements in the
            array:");
        int n = inputt.nextInt();

        int[] numbers = new int[n];
        System.out.println("Enter the numbers:");
```



```
        for (int i = 0; i < n; i++) {
            numbers[i] = inputt.nextInt();
        }
        System.out.println("Enter the target sum:");
        int targetSum = inputt.nextInt();
        // Calling
        SimpleCombinationSum solution = new SimpleCombinationSum();
        solution.findCombinationsThatSumToTarget(numbers, targetSum
            );
    }
}
```

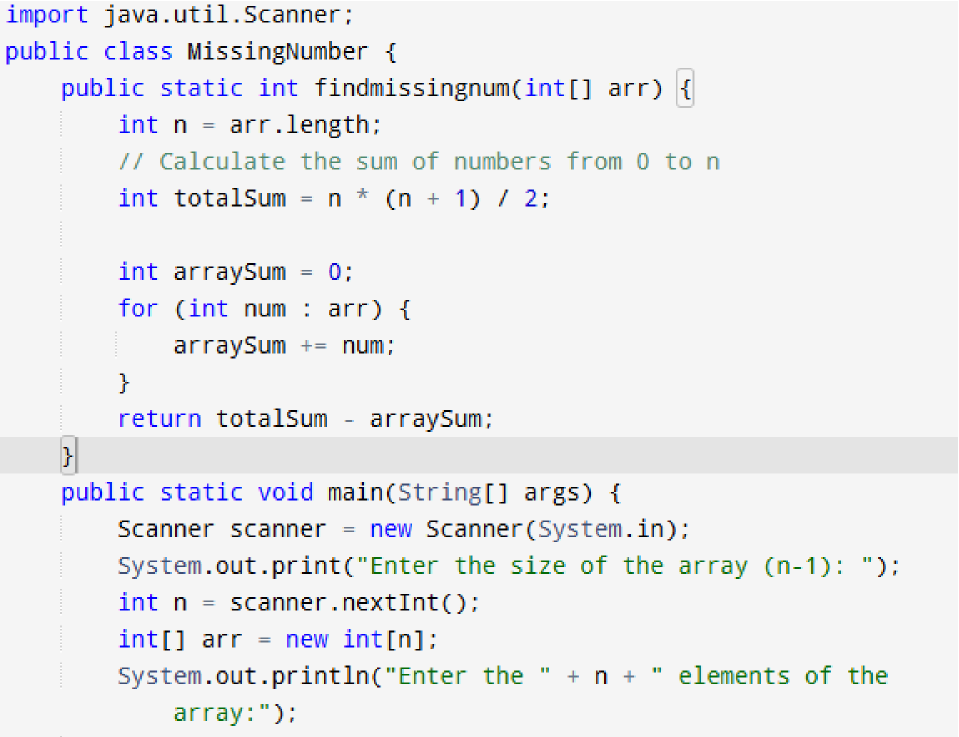
Output:



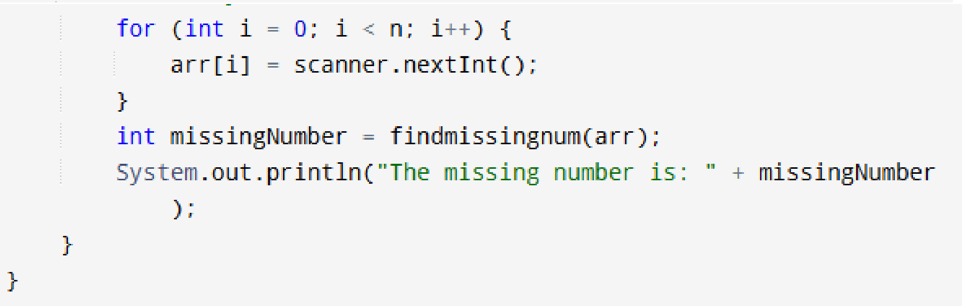
```
Enter the number of elements in the array:
4
Enter the numbers:
2 3 4 5
Enter the target sum:
6
2 4
=== Code Execution Successful ===
```

Task#4:

4. You are given an array containing n distinct numbers taken from 0, 1, 2, ..., n. Write a program to find the one number that is missing from the array

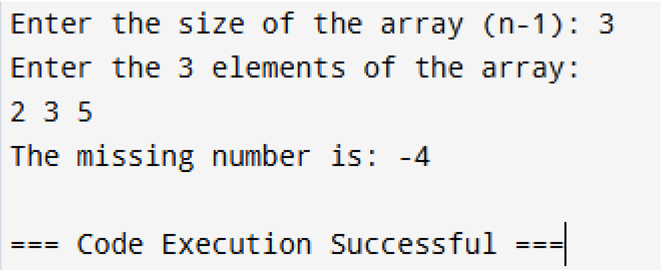


```java
import java.util.Scanner;
public class MissingNumber {
    public static int findmissingnum(int[] arr) {
        int n = arr.length;
        // Calculate the sum of numbers from 0 to n
        int totalSum = n * (n + 1) / 2;

        int arraySum = 0;
        for (int num : arr) {
            arraySum += num;
        }
        return totalSum - arraySum;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the size of the array (n-1): ");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the " + n + " elements of the
            array:");
```



```java
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        int missingNumber = findmissingnum(arr);
        System.out.println("The missing number is: " + missingNumber
            );
    }
}
```

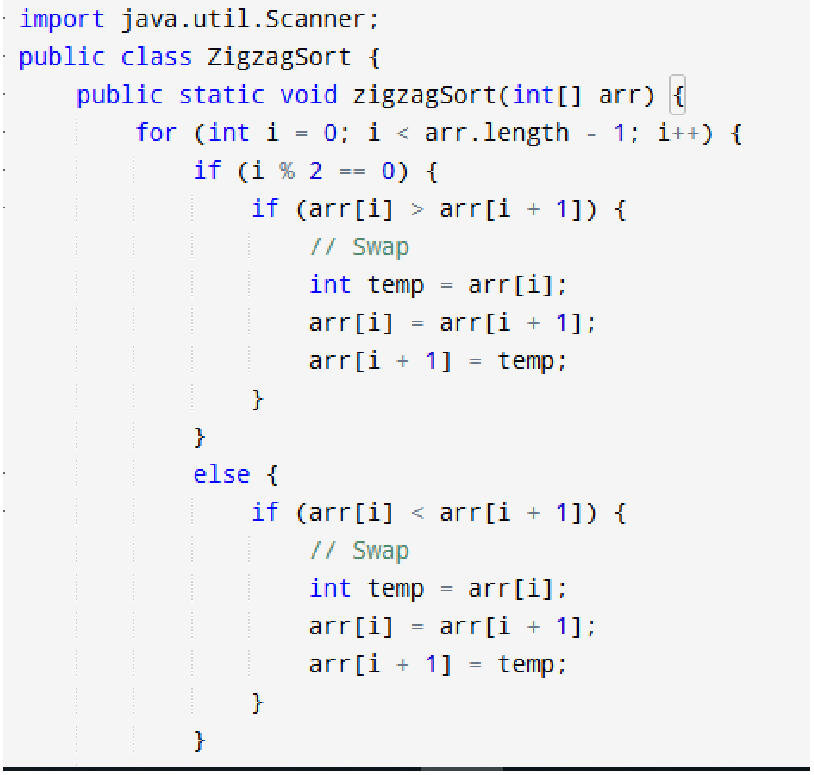
output:



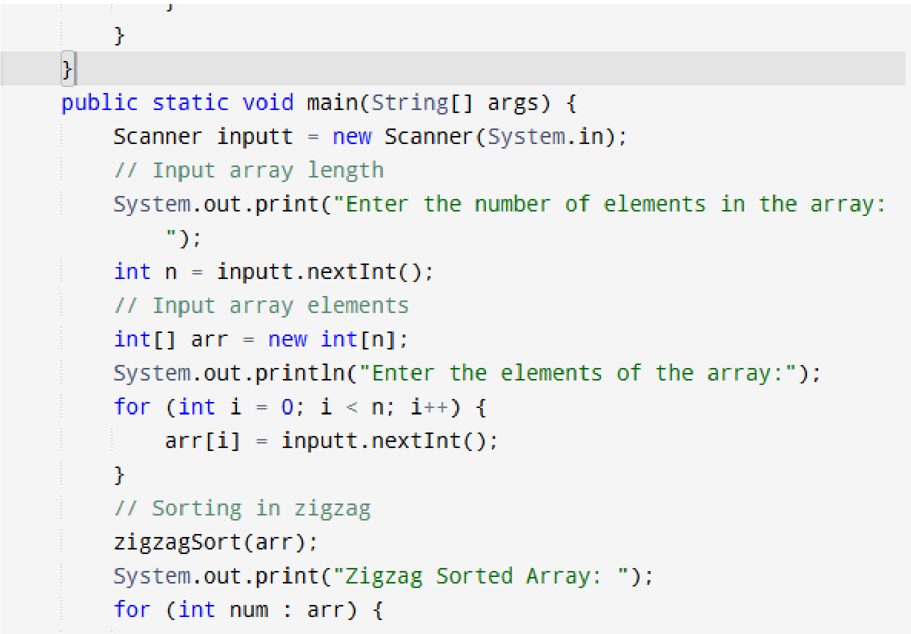
```
Enter the size of the array (n-1): 3
Enter the 3 elements of the array:
2 3 5
The missing number is: -4

=== Code Execution Successful ===
```

Task#5:

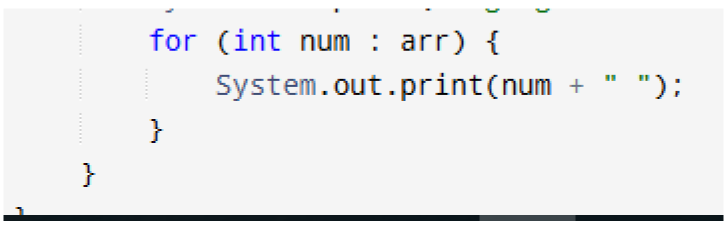
5. You are given an array of integers. Write a program to sort the array such that it follows a zigzag pattern: the first element is less than the second, the second is greater than the third, and so on.



```java
import java.util.Scanner;
public class ZigzagSort {
    public static void zigzagSort(int[] arr) {
        for (int i = 0; i < arr.length - 1; i++) {
            if (i % 2 == 0) {
                if (arr[i] > arr[i + 1]) {
                    // Swap
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                }
            }
            else {
                if (arr[i] < arr[i + 1]) {
                    // Swap
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                }
            }
```

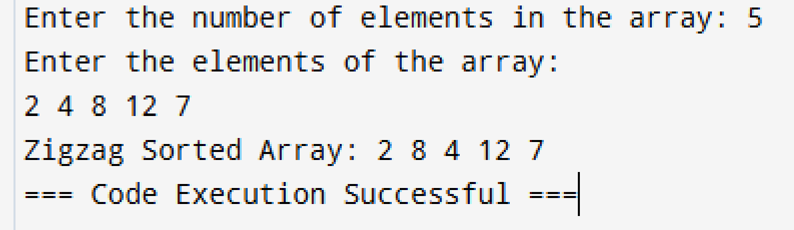


```java
        }
    }
    public static void main(String[] args) {
        Scanner inputt = new Scanner(System.in);
        // Input array length
        System.out.print("Enter the number of elements in the array:
            ");
        int n = inputt.nextInt();
        // Input array elements
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            arr[i] = inputt.nextInt();
        }
        // Sorting in zigzag
        zigzagSort(arr);
        System.out.print("Zigzag Sorted Array: ");
        for (int num : arr) {
```



```
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }
```

Output:



```
Enter the number of elements in the array: 5
Enter the elements of the array:
2 4 8 12 7
Zigzag Sorted Array: 2 8 4 12 7
=== Code Execution Successful ===
```