NAME: Shaheer Javaid ROLLNO:2023F-BSE-238 SECTION: E

LAB NO 4

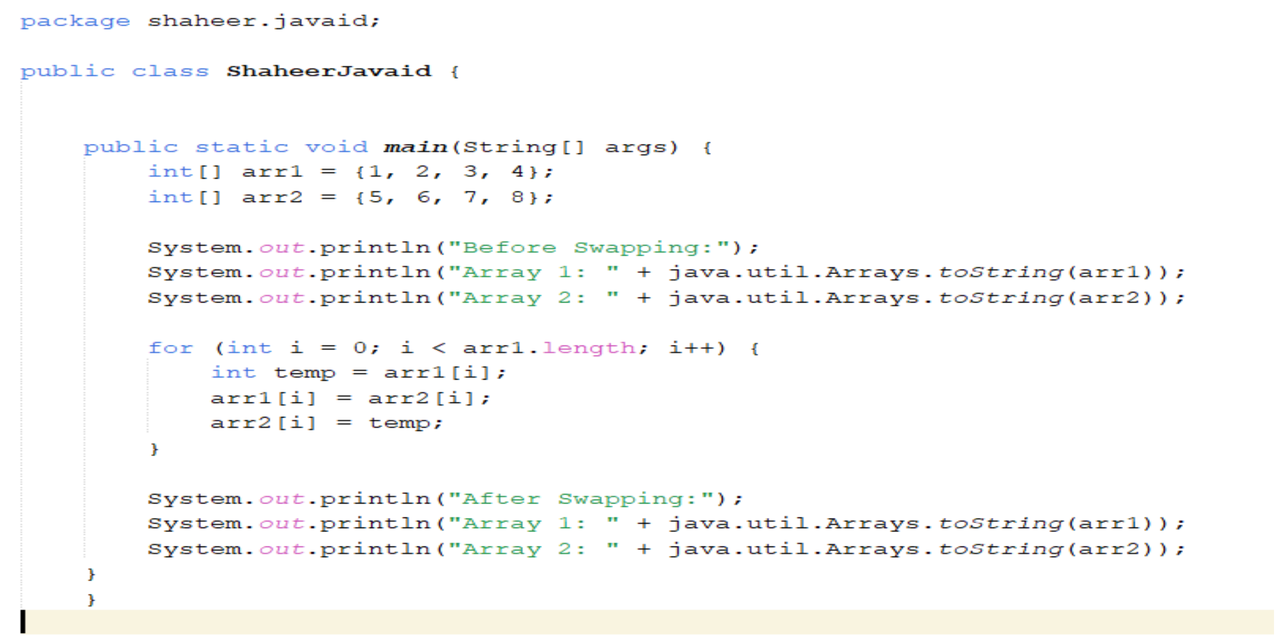
DATA STRUCTURES AND ALGORITHMS

OBJECTIVE: To understand arrays and its memory allocation.

TASK NO 1:

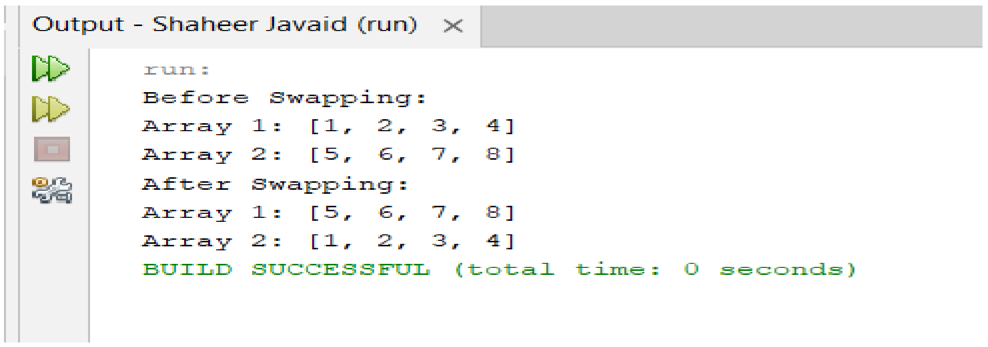
Write a program that takes two arrays of size 4 and swap the elements of those arrays.

INPUT:



```java
package shaheer.javaid;

public class ShaheerJavaid {


    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3, 4};
        int[] arr2 = {5, 6, 7, 8};

        System.out.println("Before Swapping:");
        System.out.println("Array 1: " + java.util.Arrays.toString(arr1));
        System.out.println("Array 2: " + java.util.Arrays.toString(arr2));

        for (int i = 0; i < arr1.length; i++) {
            int temp = arr1[i];
            arr1[i] = arr2[i];
            arr2[i] = temp;
        }

        System.out.println("After Swapping:");
        System.out.println("Array 1: " + java.util.Arrays.toString(arr1));
        System.out.println("Array 2: " + java.util.Arrays.toString(arr2));
    }
    }
```

OUTPUT:



Output - Shaheer Javaid (run)

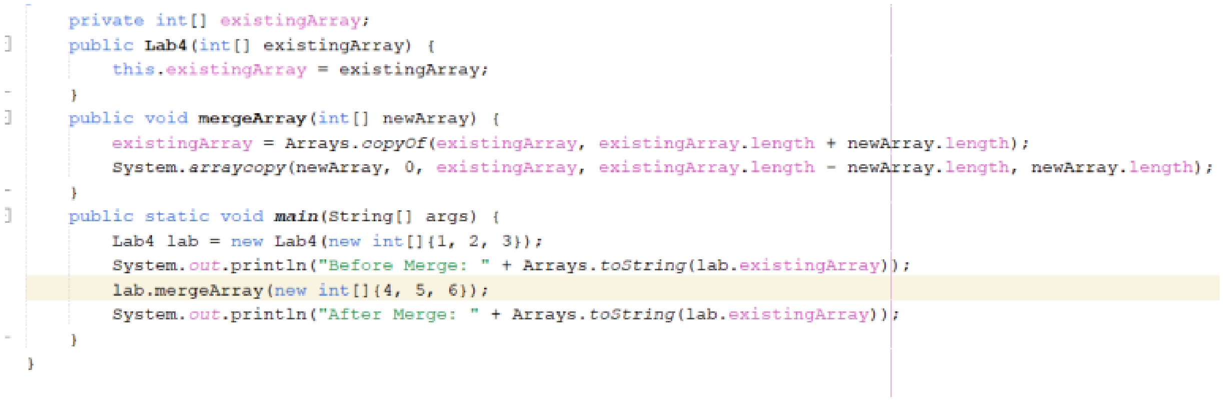
```
run:
Before Swapping:
Array 1: [1, 2, 3, 4]
Array 2: [5, 6, 7, 8]
After Swapping:
Array 1: [5, 6, 7, 8]
Array 2: [1, 2, 3, 4]
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 2:

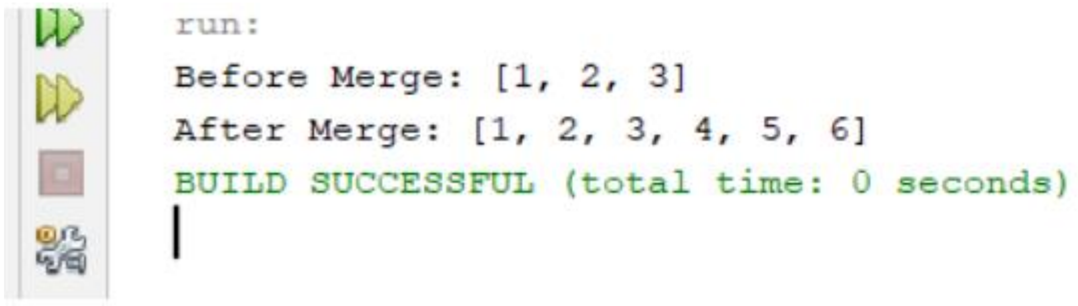
Add a method in the class that takes array and merge it with the existing one.

INPUT:



```java
    private int[] existingArray;
    public Lab4(int[] existingArray) {
        this.existingArray = existingArray;
    }
    public void mergeArray(int[] newArray) {
        existingArray = Arrays.copyOf(existingArray, existingArray.length + newArray.length);
        System.arraycopy(newArray, 0, existingArray, existingArray.length - newArray.length, newArray.length);
    }
    public static void main(String[] args) {
        Lab4 lab = new Lab4(new int[]{1, 2, 3});
        System.out.println("Before Merge: " + Arrays.toString(lab.existingArray));
        lab.mergeArray(new int[]{4, 5, 6});
        System.out.println("After Merge: " + Arrays.toString(lab.existingArray));
    }
}
```

OUTPUT:



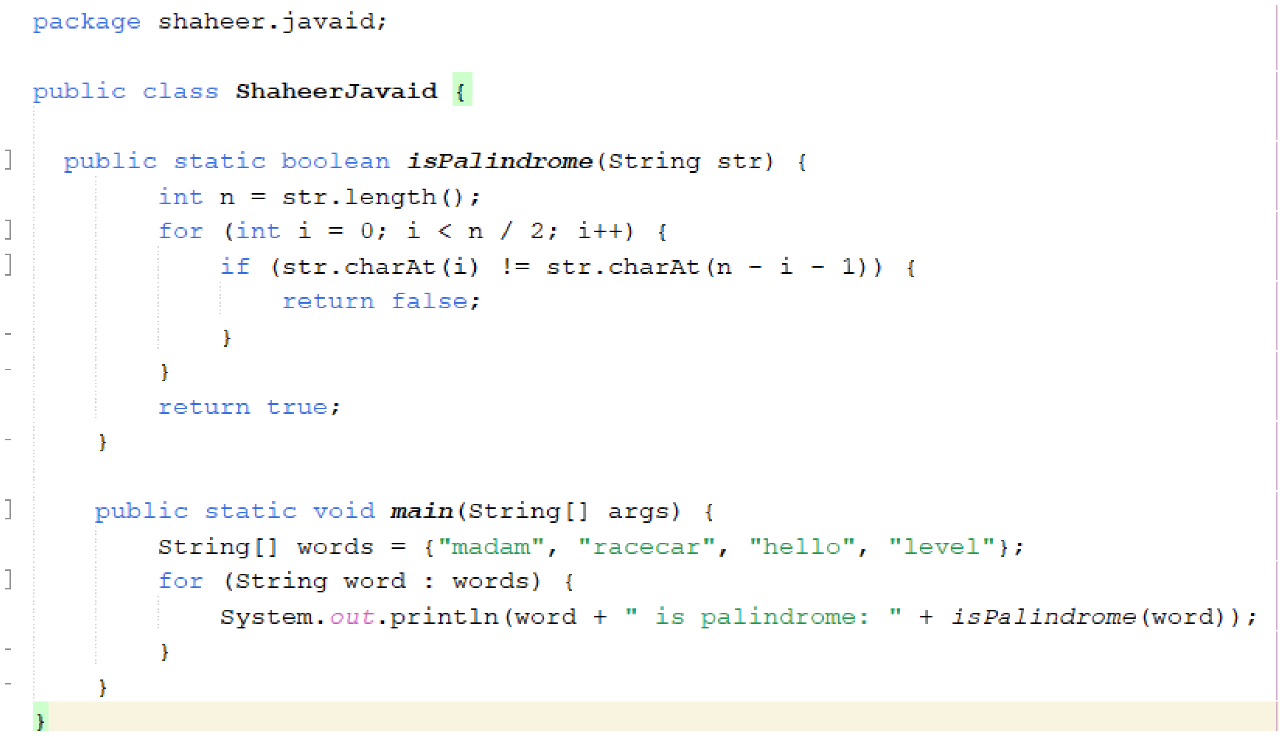
```
run:
Before Merge: [1, 2, 3]
After Merge: [1, 2, 3, 4, 5, 6]
BUILD SUCCESSFUL (total time: 0 seconds)
```

NAME: Shaheer Javaid **ROLLNO:2023F-BSE-238** **SECTION: E**

TASK NO 3:

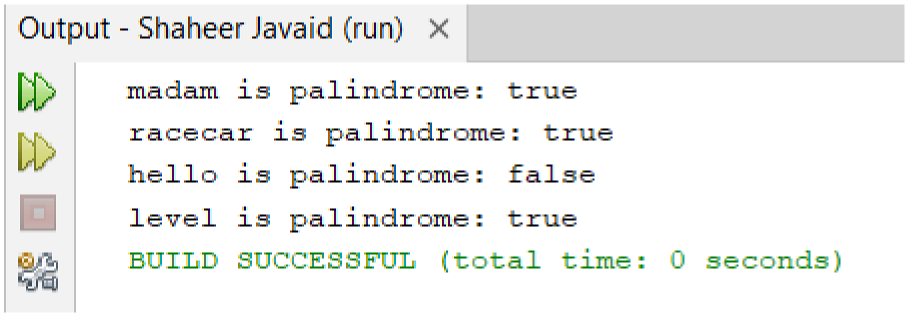
In a JAVA program, take an array of type string and then check whether the strings are palindrome or not.

INPUT:



```java
package shaheer.javaid;

public class ShaheerJavaid {

    public static boolean isPalindrome(String str) {
        int n = str.length();
        for (int i = 0; i < n / 2; i++) {
            if (str.charAt(i) != str.charAt(n - i - 1)) {
                return false;
            }
        }
        return true;
    }

    public static void main(String[] args) {
        String[] words = {"madam", "racecar", "hello", "level"};
        for (String word : words) {
            System.out.println(word + " is palindrome: " + isPalindrome(word));
        }
    }
}
```

OUTPUT:



```
madam is palindrome: true
racecar is palindrome: true
hello is palindrome: false
level is palindrome: true
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

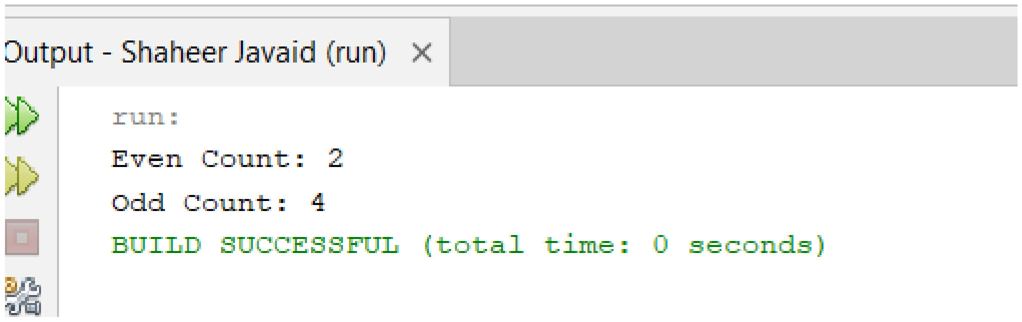
TASK NO 4:

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

INPUT:

```java
package shaheer.javaid;

public class ShaheerJavaid {

    public static void main(String[] args) {
        int[] numbers = {1, 25, 43, 34, 25, 16};
        int evenCount = 0, oddCount = 0;

        for (int num : numbers) {
            if (num % 2 == 0) {
                evenCount++;
            } else {
                oddCount++;
            }
        }

        System.out.println("Even Count: " + evenCount);
        System.out.println("Odd Count: " + oddCount);
    }
}
```

OUTPUT:



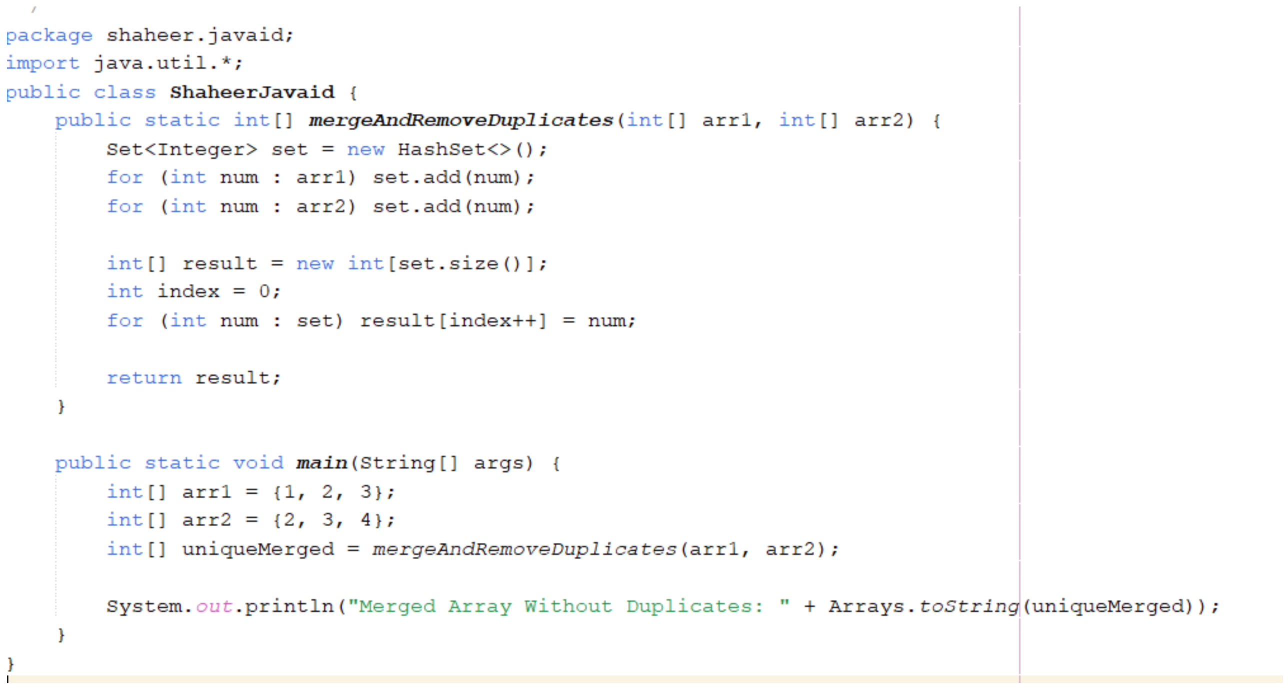
```
run:
Even Count: 2
Odd Count: 4
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 5

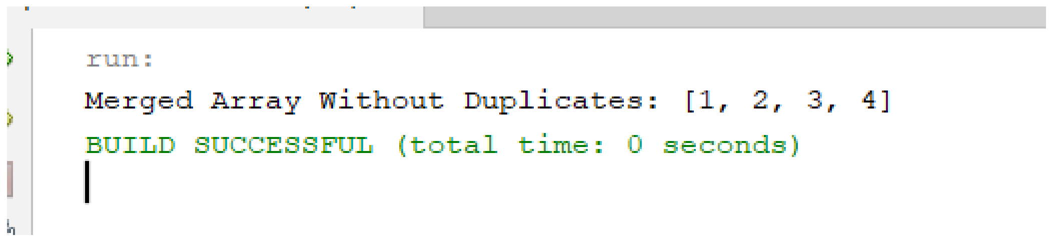
Given two integer arrays, merge them and remove any duplicate values from the resulting array.

INPUT:



```java
package shaheer.javaid;
import java.util.*;
public class ShaheerJavaid {
    public static int[] mergeAndRemoveDuplicates(int[] arr1, int[] arr2) {
        Set<Integer> set = new HashSet<>();
        for (int num : arr1) set.add(num);
        for (int num : arr2) set.add(num);

        int[] result = new int[set.size()];
        int index = 0;
        for (int num : set) result[index++] = num;

        return result;
    }

    public static void main(String[] args) {
        int[] arr1 = {1, 2, 3};
        int[] arr2 = {2, 3, 4};
        int[] uniqueMerged = mergeAndRemoveDuplicates(arr1, arr2);

        System.out.println("Merged Array Without Duplicates: " + Arrays.toString(uniqueMerged));
    }
}
```

OUTPUT:



```
run:
Merged Array Without Duplicates: [1, 2, 3, 4]
BUILD SUCCESSFUL (total time: 0 seconds)
```

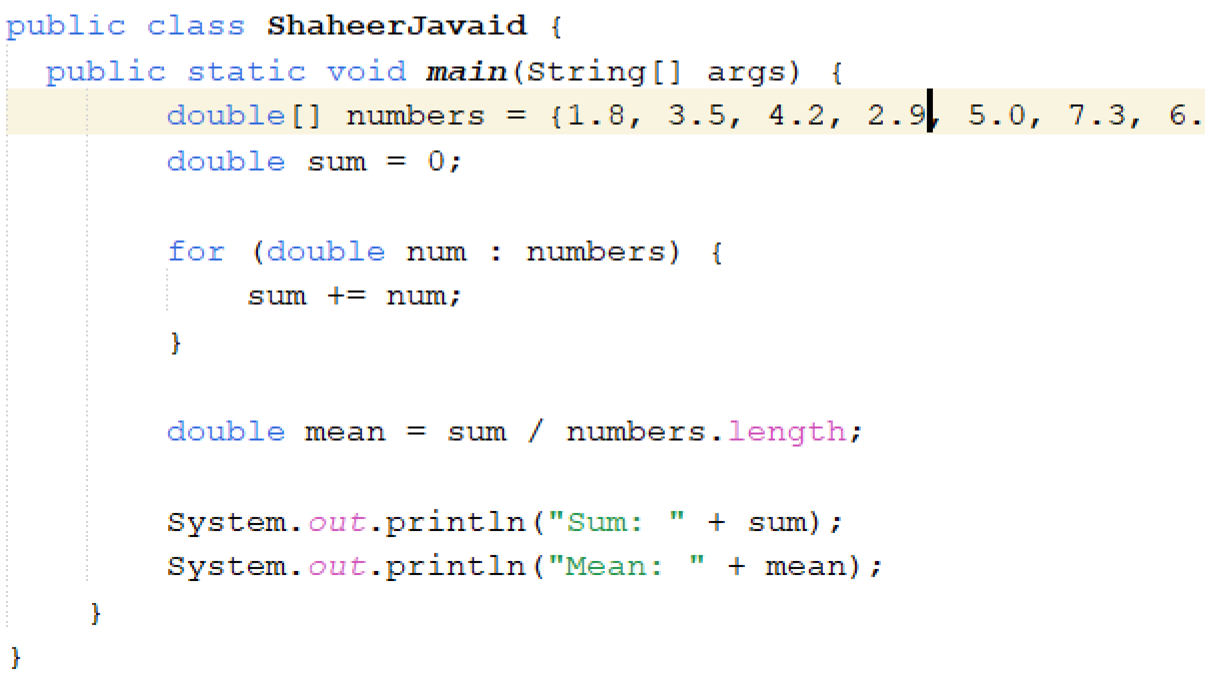
Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

HOME TASKS

TASK NO 1:

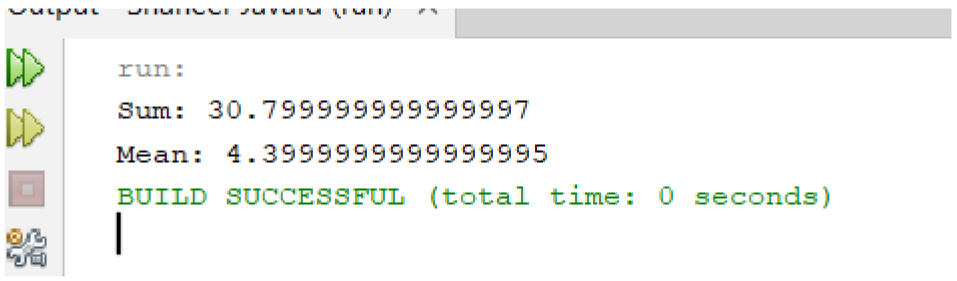
Write a program that takes an array of Real numbers having size 7 and calculate the sum and mean of all the elements. Also depict the memory management of this task.

INPUT:



```java
public class ShaheerJavaid {
  public static void main(String[] args) {
        double[] numbers = {1.8, 3.5, 4.2, 2.9, 5.0, 7.3, 6.
        double sum = 0;

        for (double num : numbers) {
            sum += num;
        }

        double mean = sum / numbers.length;

        System.out.println("Sum:  " + sum);
        System.out.println("Mean:  " + mean);
    }
}
```

OUTPUT:



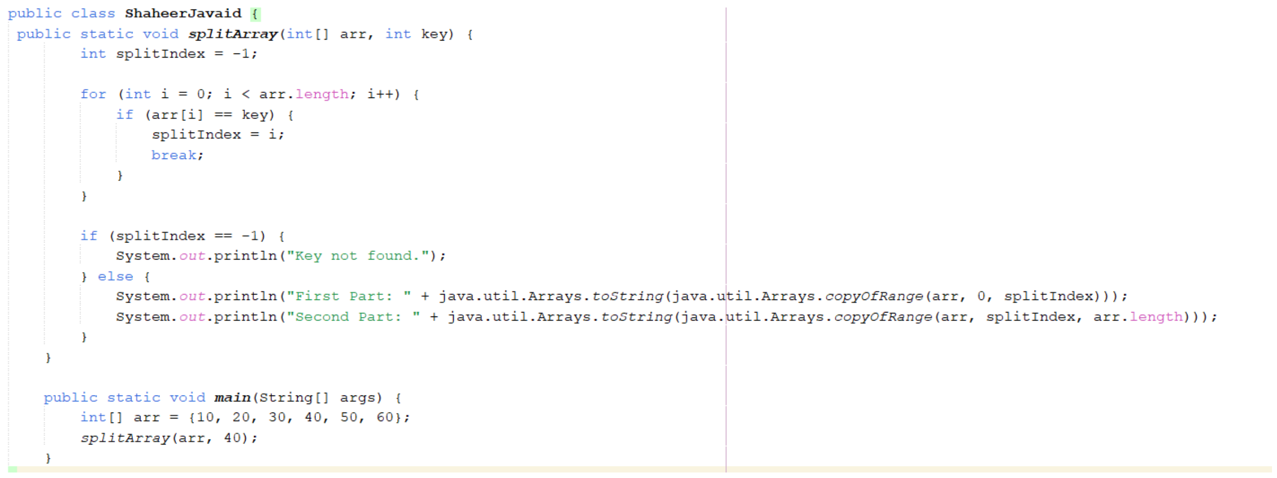
```
run:
Sum: 30.799999999999997
Mean: 4.3999999999999995
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 2

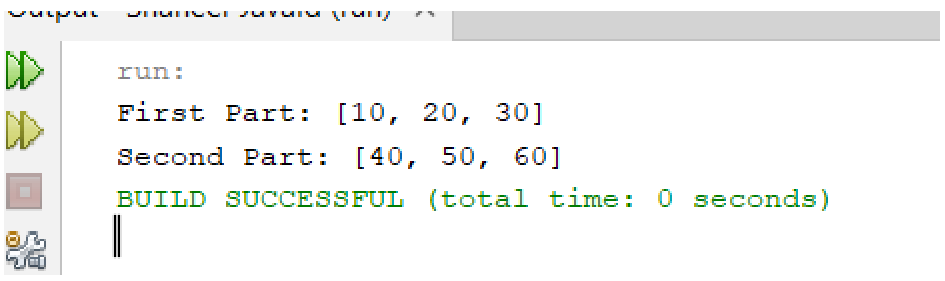
Add a method in the same class that splits the existing array into two. The method should search a key in array and if found splits the array from that index of the key.

INPUT:



```
public class ShaheerJavaid {
 public static void splitArray(int[] arr, int key) {
        int splitIndex = -1;

        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == key) {
                splitIndex = i;
                break;
            }
        }

        if (splitIndex == -1) {
            System.out.println("Key not found.");
        } else {
            System.out.println("First Part: " + java.util.Arrays.toString(java.util.Arrays.copyOfRange(arr, 0, splitIndex)));
            System.out.println("Second Part: " + java.util.Arrays.toString(java.util.Arrays.copyOfRange(arr, splitIndex, arr.length)));
        }
    }

    public static void main(String[] args) {
        int[] arr = {10, 20, 30, 40, 50, 60};
        splitArray(arr, 40);
    }
```

OUTPUT:



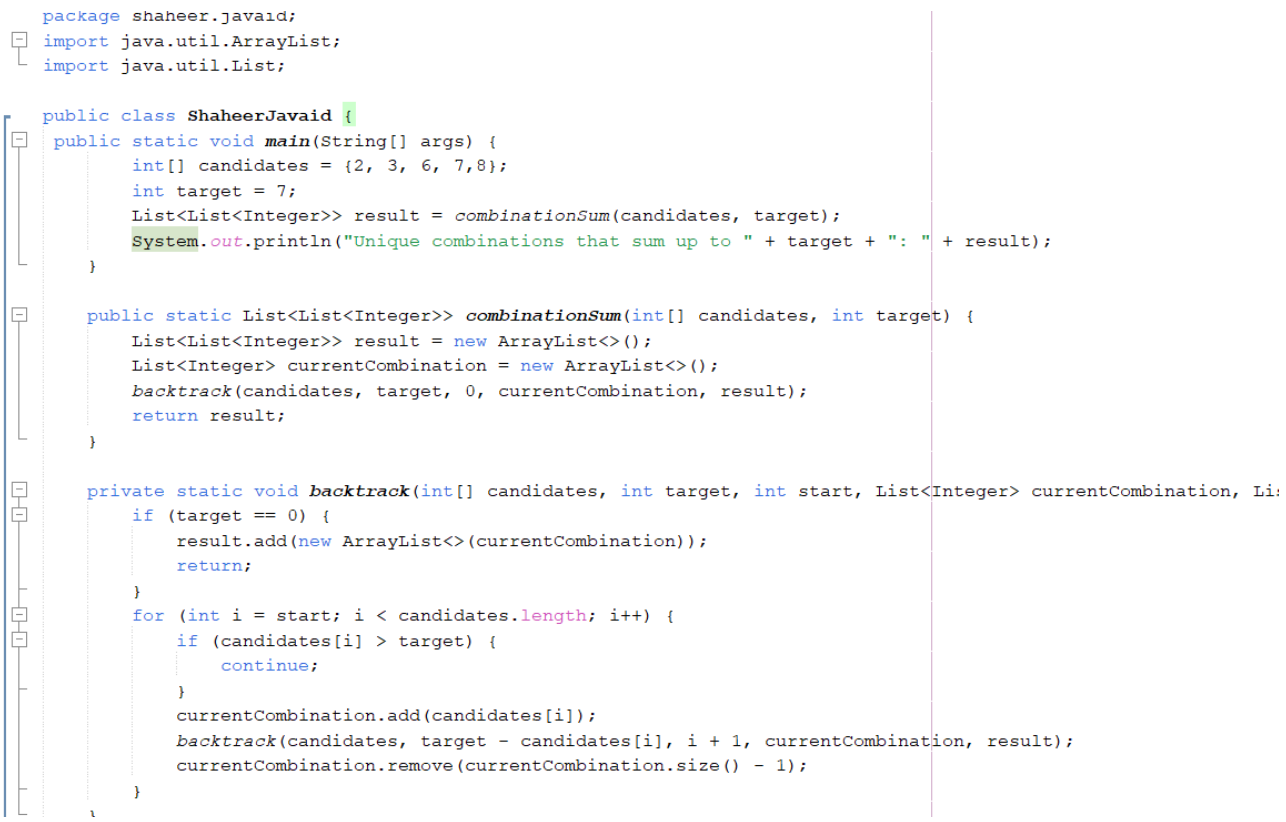
```
run:
First Part: [10, 20, 30]
Second Part: [40, 50, 60]
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 3:

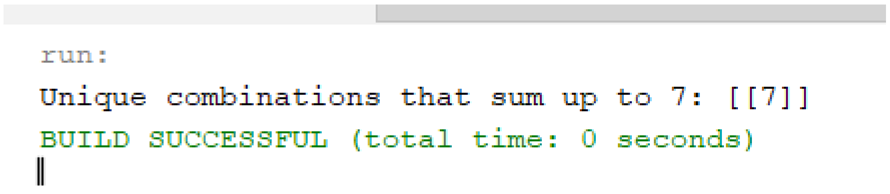
Given an array of distinct integers and a target integer, return all unique combinations of numbers that add up to the target. Each number can be used only once in the combination..

INPUT:



```java
package shaheer.javaid;
import java.util.ArrayList;
import java.util.List;

public class ShaheerJavaid {
 public static void main(String[] args) {
        int[] candidates = {2, 3, 6, 7,8};
        int target = 7;
        List<List<Integer>> result = combinationSum(candidates, target);
        System.out.println("Unique combinations that sum up to " + target + ": " + result);
    }

    public static List<List<Integer>> combinationSum(int[] candidates, int target) {
        List<List<Integer>> result = new ArrayList<>();
        List<Integer> currentCombination = new ArrayList<>();
        backtrack(candidates, target, 0, currentCombination, result);
        return result;
    }

    private static void backtrack(int[] candidates, int target, int start, List<Integer> currentCombination, Li
        if (target == 0) {
            result.add(new ArrayList<>(currentCombination));
            return;
        }
        for (int i = start; i < candidates.length; i++) {
            if (candidates[i] > target) {
                continue;
            }
            currentCombination.add(candidates[i]);
            backtrack(candidates, target - candidates[i], i + 1, currentCombination, result);
            currentCombination.remove(currentCombination.size() - 1);
        }
    }
```

OUTPUT:



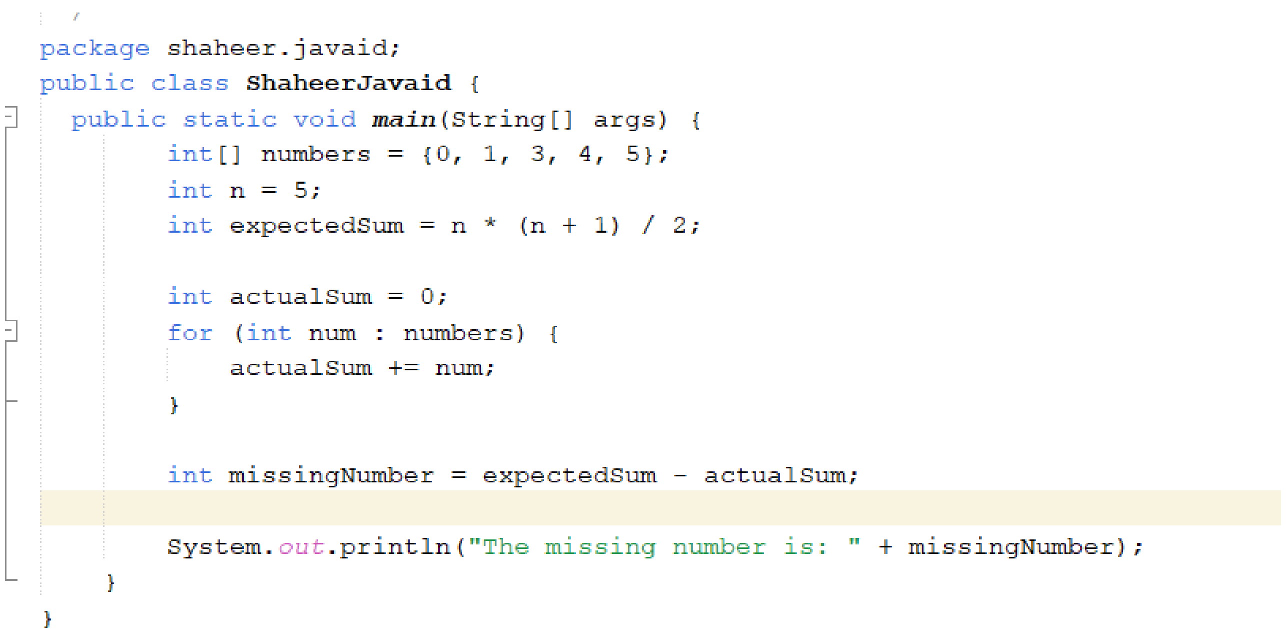
```
run:
Unique combinations that sum up to 7: [[7]]
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 4:

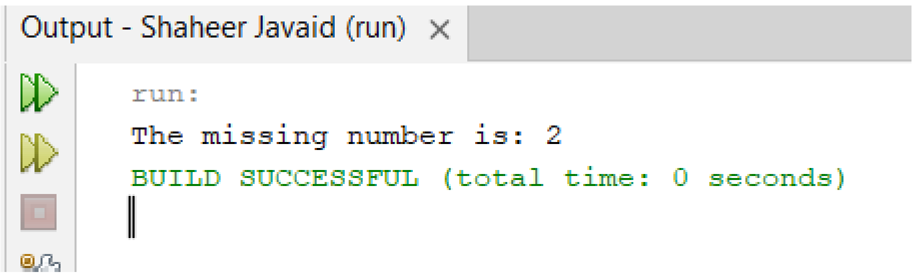
You are given an array containing n distinct numbers taken from 0, 1, 2, ..., n. Write a program to find the one number that is missing from the array.

INPUT:



```java
package shaheer.javaid;
public class ShaheerJavaid {
    public static void main(String[] args) {
            int[] numbers = {0, 1, 3, 4, 5};
            int n = 5;
            int expectedSum = n * (n + 1) / 2;

            int actualSum = 0;
            for (int num : numbers) {
                actualSum += num;
            }

            int missingNumber = expectedSum - actualSum;

            System.out.println("The missing number is: " + missingNumber);
        }
}
```

OUTPUT:



Output - Shaheer Javaid (run)

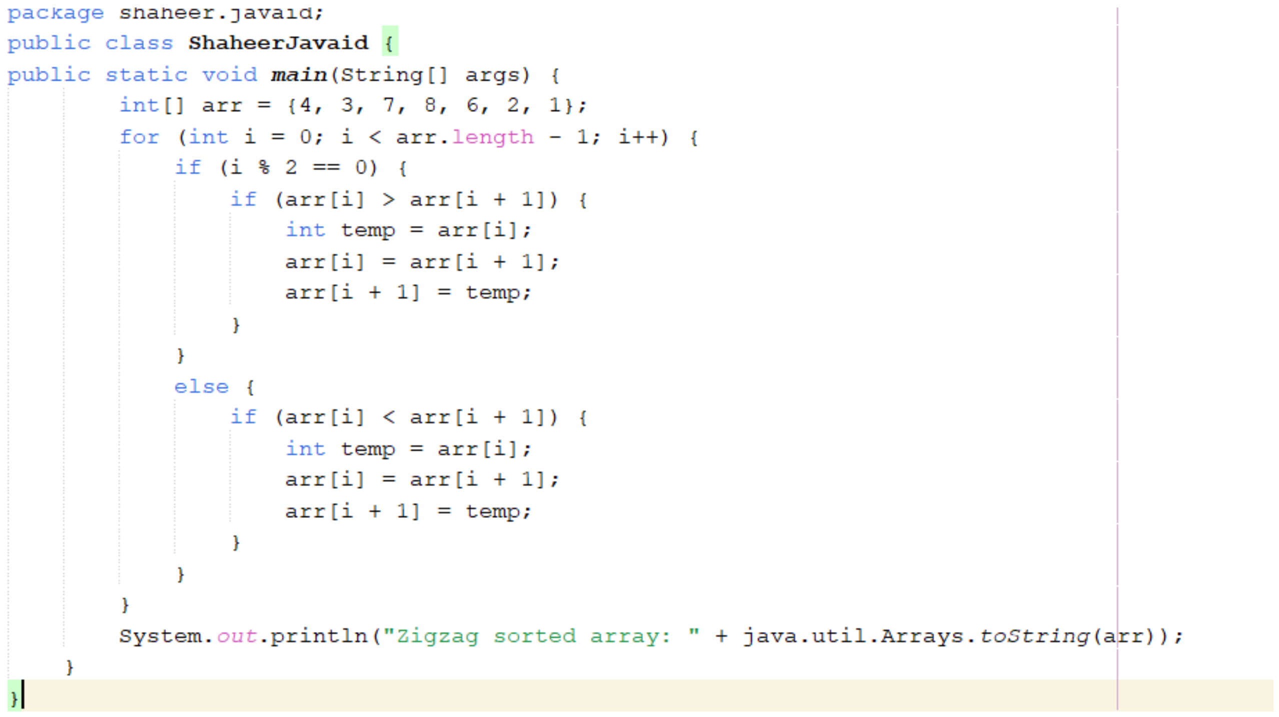
```
run:
The missing number is: 2
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 5:

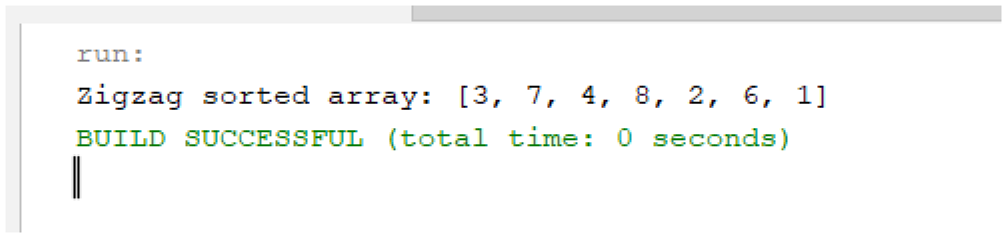
You are given an array of integers. Write a program to sort the array such that it follows a zigzag pattern: the first element is less than the second, the second is greater than the third, and so on.

INPUT:



```
package shaheer.javaid;
public class ShaheerJavaid {
public static void main(String[] args) {
        int[] arr = {4, 3, 7, 8, 6, 2, 1};
        for (int i = 0; i < arr.length - 1; i++) {
            if (i % 2 == 0) {
                if (arr[i] > arr[i + 1]) {
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                }
            }
            else {
                if (arr[i] < arr[i + 1]) {
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                }
            }
        }
        System.out.println("Zigzag sorted array: " + java.util.Arrays.toString(arr));
    }
}
```

OUTPUT:



```
run:
Zigzag sorted array: [3, 7, 4, 8, 2, 6, 1]
BUILD SUCCESSFUL (total time: 0 seconds)
```