L21

Sunday, January 23, 2022 2:36 AM

Arrays - Problem Solving - 1

Saurav Hathi

https://www.youtube.com/channel/UCp6MFWao5vWRnyRCxBsKnfw

Q1. Write a class SumOfElements with a **public** method sum that takes one parameter arr of type int[] and returns the sum of all elements in arr. The return type of sum should be long.

Assumptions:

1. arr is never null
2. return **0** in case arr is empty

Here is an example:

Cmd Args : 3 5 3 2 0
Sum of all elements in the given array is : 13
Note how the return type of the function is long and not int. The reason for this is to overcome errors due to data overflow while adding multiple int values.

q11046/SumOfElements.ja | q11046/SumOfElementsM

```java
package q11046;
public class SumOfElements {

    /**
     * Computes the sum of all the elements in the given array
     *
     *
     * @return the sum
     */
    public long sum(int[] arr) {
        // Write the code

        long sum=0;

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

            sum += arr[i];
        }
        return sum;
    }
}
```

Q2. Write a class ReversePrint with a **public** method reversePrint that takes one parameter arr of type int[] and returns the elements of arr in reverse order. The return type of ReversePrint should be int.

Assumptions:

1. arr is never null

Here is an example:

Cmd Args : 32 56 85 1
Array in reverse order is :
1
85
56
32

q11047/ReversePrint.java | q11047/ReversePrintMain

```java
package q11047;
public class ReversePrint {
    /**
     * write a logic to find the array of elements in reverse order.
     *
     *
     * @return the reverseArray
     */

     public int[] reversePrint(int[] arr) {
        //Wrtie code here

        int[] ReversePrint = new int[arr.length];

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

            ReversePrint[i] = arr[arr.length-1-i];
        }
        return ReversePrint;
     }
}
```

Q3.

Write a class ElementCheck with a **public** method checkFirstOrLast that takes two parameters one is arr of type int[] and second one is arg of type int and returns true if the arg is first or last element in the arr else returns false. The return type of checkFirstOrLast is boolean.

Assumption:

1. arr is never null

Here is an example:

Enter no of elements in the array: 4
Enter elements in the array seperated by space:1 2 3 6
Enter the search element:6
true

q11048/ElementCheck.jav | q11048/ElementCheckMa

```java
package q11048;

public class ElementCheck {

/** write a logic to check whether the given element is present in the first or last in the array
 *
 *
 *
 *@return true if the element present if not return false
 *
 */
    public boolean checkFirstOrLast(int[] arr, int arg) {
        //Write your code

        boolean checkFirstOrLast = false;

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

            if(arr[0] == arg || arr[arr.length-1] == arg){

                checkFirstOrLast = true;

            } else{

                return false;
            }
        }

        return checkFirstOrLast;
    }
}
```

Q4. write a class ElementCount with a **main** method which passes an arr of type int[] and an element of type int. Print number of times the element is present in the arr.

Here is an example:

int[] arr = {1, 12, 9, 3, 5, 3, 78, 4, 3, 9, 18, 56, 1, 5}
Cmd Args : 1
2

q11049/ElementCount.jav

```java
package q11049;
public class ElementCount {
    public static void main(String[] args) {

        int[] arr = {1, 12, 9, 3, 5, 3, 78, 4, 3, 9, 18, 56, 1, 5};
        int element = Integer.parseInt(args[0]);
        int count = 0;

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

            if(arr[i] == element){
                count++;
            }
        }

        System.out.println(count);
    }
}
```

Q5. Write a class ElementCheck with a **public** method checkFirstOrLast that takes two parameters arr1 and arr2 are of type int[] and returns true if first or last elements of arr1 and arr2 are same. The return type of checkFirstOrLast should be boolean.

Assumptions:

1. arr1 and arr2 will never null
2. arr1 and arr2 lengths are not equal

Here is an example:
Enter no of elements in the array1:
5
Enter elements in the array1 seperated by space:
3 6 8 7 4
Enter no of elements in the array2:
3
Enter elements in the array2 seperated by space:
6 5 4
true

Sample Test Cases

Test Case 1:
Expected Output:
Enter·no·of·elements·in·the·array1:
4
Enter·elements·in·the·array1·seperated·by·space:
6 3 2 1
Enter·no·of·elements·in·the·array2:
4
Enter·elements·in·the·array2·seperated·by·space:
3 2 1 5
false

Test Case 2:
Expected Output:
Enter·no·of·elements·in·the·array1:
5
Enter·elements·in·the·array1·seperated·by·space:
3 6 8 7 4
Enter·no·of·elements·in·the·array2:
3
Enter·elements·in·the·array2·seperated·by·space:
6 5 4
true

q11050/ElementCheck.jav | q11050/ElementCheckMa

```java
package q11050;
public class ElementCheck {
    /**
     * check if first or last elements of the arr1 and arr2 are same or not
     *
     *
     * @return result
     */

    public boolean checkFirstOrLast(int[] arr1, int[] arr2) {
        //Write your code here

        boolean checkFirstOrLast = false;

        if(arr1[0] == arr2[0] || arr1[arr1.length-1] == arr2[arr2.length-1]){

            checkFirstOrLast = true;
        } else {

            return false;
        }

        return checkFirstOrLast;

    }
}
```

Q6. Write a class ElementCheck with a **public** method elementFinder that takes two parameters one is arr of type int[] second one is element of type int that returns true if the element present in the arr only one time.

Assumptions:

1. arr is never null

These are examples for your understanding:

Enter no of elements in the array:
5
Enter elements in the array seperated by space:
9 5 12 35 6
Enter the search element:
5
true
Enter no of elements in the array:
4
Enter elements in the array seperated by space:
1 2 2 3
Enter the search element:
2
false

q11054/ElementCheck.jav | q11054/ElementCheckMa

```java
package q11054;

public class ElementCheck {
    /**
     * Compute if the given elemetn is present in the array only one time
     *
     *
     * @return true if it is present else return false
     */

    public boolean elementFinder(int[] arr, int element) {

        //Write your code here
        boolean foundEl = false;
        int count = 0;

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

            if(arr[i] == element){

                count++;
            }
        }

        if(count > 1){

            foundEl = false;
            return foundEl;

        } else{

            return true;
        }

    }
}
```

Q7. Write a class ElementCheck with a **public** method elementFinder that takes one parameter arr of type int[] and return true if the first and last elements of the arr are same else return false. The return type of elementFinder should be boolean.

Assumptions:

1. arr is never null

Here are examples for your understanding:
Cmd Args : 33 25 12 5 33
true
Cmd Args : 1 2 3 4
false

q11055/ElementCheck.jav | q11055/ElementCheckMa

```java
package q11055;

public class ElementCheck {

/**
 * Find first and last elements of the array are same are not
 *
 * @return true if both are same else return false
 */


    public boolean elementFinder(int[] arr) {

        //Write your code here
        if(arr[0] == arr[arr.length - 1]){

            boolean yes = true;
            return yes;

        } else{

            return false;
        }

    }
}
```

Q8. Write a class ElementCheck with a **public** method elementFinder that takes one parameter arr of type int[] and returns true if the first four elements in the arr contains number 4 else returns false.

Assumptions:

arr is never null

Length of arr may be less than four

These are examples for understanding:

Cmd Args : 36 51 42 4
true
Cmd Args : 1 2
false

q11056/ElementCheck.jav | q11056/ElementCheckMa

```java
package q11056;

public class ElementCheck {

    /**
     * Find if the first four elements in the array contains number 4 or not
     *
     *
     * @return result
     */

    public boolean elementFinder(int[] arr) {

        //Write your code here
        boolean result = false;
        int high = arr.length;
        int num = 4;

        for(int i = 0; i < high; i++){

            if(arr[i] != num){
                result = false;

            } else if(high > num){

                result = false;
            }
            else{

                return true;
            }
        }

        return result;

    }
}
```

Q9. Write a class SumOfElements with a **public** method sum that takes one parameter arr of type int[] and returns sum of all positive elements in the arr. The return type of sum should be int.

Assumptions:

arr is never null

arr may contain -ve numbers

These are examples:

Cmd Args : -35 -52 -12 -99
Sum of all positive elements in the array is : 0
Cmd Args : 36 12 -11 10
Sum of all positive elements in the array is : 58

q11057/SumOfElements.j q11057/SumOfElementsM

```java
package q11057;

public class SumOfElements {
    /**
     * Compute sum of all +ve elements in the array excluding -ve numbers
     *
     *
     * @return sum
     */

        public int sum(int[] arr) {

            //Write your code here
            int sum = 0;

            for(int i = 0; i < arr.length; i++){
                if(arr[i] > 0){
                    sum += arr[i];
                }
            }
            return sum;

    }
}
```

Q10.

Write a class SwapFirstAndLast with a **public** method swap that takes one parameter arr of type int[]. Write a code to swap the first and last elements of the array and print all the elements of the array.

For example:
Cmd Args : 1 5 6 7 8
8
5
6
7
1

q11058/SwapFirstAndLast q11058/SwapFirstAndLast

```java
package q11058;

public class SwapFirstAndLast{
    public static void swap(int[] arr){

        int swap=0;
        int size = arr.length;

            swap = arr[size-1];
            arr[size-1] = arr[0];
            arr[0] = swap;

        for(int i = 0; i < size; i++){

            System.out.println(arr[i]);
        }
    }
}
```

Q11.

Write a class SequenceCheck with a **public** method sequenceCheck that takes one parameter arr of type int[] and returns true if 6, 9, 12 present consecutively in the arr. The return type of sequenceCheck should be boolean.

Assumptions:
1. arr is never null
2. Elements 6, 9, 12 are appear consecutiviely

Here are examples:
Cmd Args : 62 32 6 9 12
true
Cmd Args : 99 36 6 12 56 9
false

q11059/SequenceCheck.ja | q11059/SequenceCheckM

```java
package q11059;

public class SequenceCheck {
    /**
     * Find the given elements present consecutively in the array or not
     *
     *
     * @return result
     */

    public boolean sequenceCheck(int[] arr) {

        //Write your code here

        int size = arr.length;
        int count = 1;
        boolean result = false;

        for(int i = 0; i < size-1; i++){

            if(arr[i] == 6){

                if(arr[i+1] == 9 && arr[i+2] == 12){

                    result = true;
                } else{

                    return false;
                }
            }
        }
        return result;

    }
}
```

Q12. Write a class SequenceCheck with a **public** method sequenceCheck that takes one parameter arr of type int[] and returns true if the elements 1,2,3 are present in the arr.The returen type of sequenceCheck should be boolean.

Assumptions:

1. arr is never null
2. The elements need not be in consecutive order

Here are examples:

Cmd Args : 1 6 3 2
true
Cmd Args : 3 6 4 7 8
false

q11060/SequenceCheck.ja | q11060/SequenceCheckM

```java
package q11060;
import java.util.Arrays;


public class SequenceCheck {
    /**
     * check if the given array contains the elements 1,2,3
     *
     *
     *
     * @return true if contain else return false
     */




    public boolean sequenceCheck(int[] arr) {

        //Write your code here
        int size = arr.length;
        boolean sequenceCheck = false;
        int a=1;

        for(int i = 0; i < size; i++){

            if(arr[i] == 1 && arr[i] == 2){

                sequenceCheck = true;
            } else if(arr[i] == 2){

                sequenceCheck = true;
            } else if(arr[i] == 3 && arr[i] == 1){

                sequenceCheck = true;
            } else if(arr[i] < 0){

                sequenceCheck = false;
                break;
            }
        }

        if(sequenceCheck == true){

        return sequenceCheck;
        } else{

            return false;
        }
    }
}
```

Q13. Write a class FindMiddle with a **public** method findMiddle that takes one parameter arr of type int[] and print the middle element in the arr

Assumptions:

1. arr is never null
2. arr length is even it should print the middle two numbers
3. arr length is odd it prints the middle element

Here are examples for your understanding:

Cmd Args : 1 6 3 5 4
3
Cmd Args : 3 2 1 6 5 4
1
6

q11061/FindMiddle.java | q11061/FindMiddleMain.j

```java
package q11061;

public class FindMiddle {
    /**
     * Find the middle element in the given array
     *
     *
     *
     * @ return element
     */
    public void findMiddle(int[] arr) {

        //Write your code here
        int size = arr.length;
        int big = arr[0];
        int end = size-1;
        // float mid = arr[0] + (size-arr[0])/2;
        int mid,mid1;

        if(size%2 == 0){

            mid = arr[size/2-1];

            mid1 = arr[size/2];

            System.out.println(mid+"\n"+mid1);
        } else{

            System.out.println(arr[size/2]);

        }


    }

}
```

Q14. Write a class SequenceCount with a **public** method sequenceCount that takes one parameter arr of type int[] and returns the sequence count 1,1 in the arr. The return type of sequenceCount should be int.

Assumptions:

1. arr is never null
2. Overlapping of counting is allowed

Here is an example:

Enter no of elements in the array:
7
Enter elements in the array seperated by space:
1 -1 1 1 1 2 3 1
2

q11062/SequenceCount.ja | q11062/SequenceCountM

```java
package q11062;

public class SequenceCount {
    /**
     * Find the sequence count 1,1 int given array
     *
     *
     * @return count
     */

    public int sequenceCount(int[] arr) {

        //Write your code here
        int size = arr.length;
        int count=0;


        for(int i = 0; i < size-1; i++){

            if(arr != null  && arr[i] == 1 && arr[i+1] == 1){
                    count++;

              }
            // System.out.println(arr[i]);
        }

        return count;


    }
}
```

Q15. Write a class InitializeArray with a **public** method initialize that takes two parameters len and ele are of type int returns an array of length len and set all the elements in the array to ele. Assumptions:

1. arr is never null

Here is an example:

Enter length of array:
4
Enter element in the array:
3
The output array is:
3
3
3
3

q11063/InitializeArray.jav | q11063/InitializeArrayMai

```java
package q11063;

public class InitializeArray {
    /**
     * Set all the elemets in the array to given element and set to length of array to given length
     *
     *
     *
     * @return arry
     */

    public int[] initialize(int len, int ele) {
        //Write  your code here

        int arr[] = new int[len];

        for(int i = 0; i < len; i++){

            arr[i]=ele;
        }
        return arr;

    }
}
```

Q16. Write a class SequenceCount with a **public** method sequenceCount that takes one parameter arr of type int[] and returns the count of sequences present in the arr. The return type of sequenceCount should be int.

Assumptions:

1. arr is never null
2. arr may contain zero or more sequences

A sequence is defined as a combination of three numbers in continuous sequence which are of values: 1x, 2x, 3x.

Here are some examples for your understanding:
Cmd Args : 9 7 3 6 9
1
Cmd Args : 2 6 8 2 1
0
Cmd Args : 10 20 30 1 2 3
2

q11064/SequenceCount.ja | q11064/SequenceCountM

```java
package q11064;

public class SequenceCount {
    /**
     * Find the count of sequences present in the given arry.
     *
     *
     *
     *
     * @return count
     */



    public int sequenceCount(int[] arr) {

        int count=0,i=0;

        //Write your code here

        while( arr != null && i<arr.length-2){
            if(arr[i+1] == 2*arr[i] && (arr[i+2] == 3*arr[i])){
                count++;
            }
            i++;
        }
        return count;

    }
}
```

Q17. Write a class SumOfArrays with a **public** method sum that takes two parameters arr1 and arr2 are of type int[] and returns the sum of arr1 and arr2 elements in to the third array.

Assumptions:

1. arr1 and arr2 will never null
2. arr1 and arr2 are of same length

Here is an example:
Enter no of elements in the arr1:
3
Enter elements in the arr1 seperated by space:
1 2 3
Enter no of elements in the arr2:
3
Enter elements in the arr2 seperated by space:
4 5 6
5
7
9

q11065/SumOfArrays.java | q11065/SumOfArraysMain

```java
package q11065;
public class SumOfArrays {
    /**
     * Find sum of two array elements and store the elements in the third array
     *
     *
     *
     *
     * @return resultant arry
     */

    public int[] sum(int[] arr1, int[] arr2) {
        //Write your code here

        int[] result = new int[arr1.length];

            for(int i = 0, j = 0, k = 0; i < result.length; i++, j++, k++){

                if(arr1 != null && arr2 != null && arr1.length == arr2.length){

                    result[k] =arr1[i]+arr2[j];

                } else{

                    result[k] = arr1[i]+arr2[j];

                }
        }

        return result;

    }
}
```

Q18. Write a class CompareArrays with a **public** method compareArrays that takes two parameters arr1 and arr2 are of type int[] and returns true if the lengths of arr1 and arr2 are equal.

Here is an example:
Enter lenght of the arr1:
5
Enter lenght of the arr2:
5
true

q11066/CompareArrays.ja | q11066/CompareArraysM

```java
package q11066;

public class CompareArrays {
    /**
     * Find the length of the two arrays are equal or not
     *
     *
     * @return result
     */



    public boolean compareArrays(int[] arr1, int[] arr2) {
        //Write your code here
        boolean result = false;

        if(arr1.length == arr2.length){

            result = true;

        } else {

            result = false;

        }
        return result;
    }
}
```

Q19. Write a class CompareArrays with a **public** method compareArrays that takes two parameters arr1 and arr2 are of type int[] and returns true if arr1 and arr2 are of equal length and also have same elements. else returns false

Here are examples for your understanding:
Enter no of elements in the arr1:
3
Enter elements in the arr1 seperated by space:
8 9 7
Enter no of elements in the arr2:
3
Enter elements in the arr2 seperated by space:
8 9 7
true
Enter no of elements in the arr1:
3
Enter elements in the arr1 seperated by space:
3 6 7
Enter no of elements in the arr2:
4
Enter elements in the arr2 seperated by space:
3 6 7 1
false
Hint: Iterate through the first array and compare each element with the corresponding element in the second array.

Sample Test Cases

Test Case 1:
Expected Output:
Enter·no·of·elements·in·the·arr1:
3
Enter·elements·in·the·arr1·seperated·by·space:
8 9 7
Enter·no·of·elements·in·the·arr2:
3
Enter·elements·in·the·arr2·seperated·by·space:
8 9 7
true

Test Case 2:
Expected Output:
Enter·no·of·elements·in·the·arr1:
3
Enter·elements·in·the·arr1·seperated·by·space:
12 54 36
Enter·no·of·elements·in·the·arr2:
4
Enter·elements·in·the·arr2·seperated·by·space:
54 69 78 52
false

Test Case 3:
Expected Output:
Enter·no·of·elements·in·the·arr1:
3
Enter·elements·in·the·arr1·seperated·by·space:
3 6 7
Enter·no·of·elements·in·the·arr2:
4
Enter·elements·in·the·arr2·seperated·by·space:
3 6 7 1
false

Test Case 4:
Expected Output:
Enter·no·of·elements·in·the·arr1:
0
Enter·elements·in·the·arr1·seperated·by·space:
Enter·no·of·elements·in·the·arr2:
0
Enter·elements·in·the·arr2·seperated·by·space:
true

q11067/CompareArrays.ja | q11067/CompareArraysM

```java
package q11067;

public class CompareArrays {
    /** Compare lengths and elements of the arr1 and arr2 are equal or not
     *
     *
     *
     * @return result
     */

    public boolean compareArrays(int[] arr1, int[] arr2) {

        //Write your code here
        int size1 = arr1.length;
        int size2 = arr2.length;
        boolean result = false;

        if(size1 == size2){

            result = true;

            for(int i = 0, j = 0; i < size1; i++, j++){

                if(arr1[i] == arr2[j]){

                    result = true;

                } else{

                    result = false;
                }
            }

        }

        else {
            result = false;
        }


        return result;


    }
}
```

Q20. Write a class ElementCheck with a **public** method checkElement that takes three parameters one is arr of type int[] other are arg1 and arg2 are of type int and returns true if arr contains either arg1 or arg2 elements only.

These are examples for your understanding:
Enter no of elements in the array:
4
Enter elements in the array separated by space:
22 33 22 33
Enter arg1 element:
22
Enter arg2 element:
33
true
Enter no of elements in the array:
5
Enter elements in the array separated by space:
11 22 11 22 11
Enter arg1 element:
11

Enter arg2 element:
12
false

q11068/ElementCheck.jav | q11068/ElementCheckMa

```java
package q11068;
public class ElementCheck {
    /**
     * Find if the arr contains only arg1 or arg2 elements
     *
     *
     * @return result
     */

    public boolean checkElement(int[] arr, int arg1, int arg2) {

        //Write your code here


        boolean result = false;

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

            if(arr[i] != arg1 && arr[i] != arg2){

                result = false;
                break;

            }
            else{

                result = true;
            }
        }

        return result;

    }
}
```

Q21. Write a class CountEvens with a **public** method countEvens that takes one parameter arr of type int[] and returns the count of even numbers present in the arr.

Assumptions:

1. arr is never null

These are examples:

Enter no of elements in the array:
5
Enter elements in the array seperated by space:
1 2 3 4 6
3
Enter no of elements in the array:
5
Enter elements in the array seperated by space:
1 3 7 9 5
0

q11069/CountEvens.java | q11069/CountEvensMain.

```java
package q11069;


public class CountEvens {
    /**
     * Find the count of even numbers in the given array
     *
     *
     *
     * @return count
     *
     */


    public int countEvens(int[] arr) {

        //Write your code here

        int count=0;

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

            if(arr[i]%2 == 0){
                count++;
            }
        }
        return count;


    }
}
```

Q22. Write a class ConcatenateArrays with a **public** method concatenate that takes two parameters arr1 and arr2 are of type int[]. Create a new array that has the elements of both arr1 and arr2 in the same order and print the elements.

Example:
Enter no of elements in the arr1:
3
Enter elements in the arr1 separated by space:
1 2 3
Enter no of elements in the arr2:
3
Enter elements in the arr2 separated by space:
4 5 6
The resultant array is :
1
2
3
4
5
6

q11070/ConcatenateArray | q11070/ConcatenateArray

```java
package q11070;

public class ConcatenateArrays{
    public static void concatenate(int[] arr1, int[] arr2){

        int[] arr3 = new int[arr1.length + arr2.length];

        for(int i=0; i<arr1.length; i++){

            arr3[i] = arr1[i];

        }
        for(int i=0, j=arr1.length; j<arr3.length && i < arr2.length;  i++, j++){

            arr3[j] = arr2[i];
        }


       for(int i=0; i<arr3.length; i++){

        System.out.println(arr3[i]);

       }
    }
}
```

Q23. Write a class ElementDiff with a **public** method findDiff that takes one parameter arr of type int[] and returns the difference between largest and smallest elements in the arr.

Assumptions:

1. arr is never null

Here is an example:

Enter no of elements in the arr:
5
Enter elements in the arr seperated by space:
33 78 95 14 45
Difference between largest and smallest elements in the array is: 81

q11071/ElementDiff.java | q11071/ElementDiffMain.j

```java
package q11071;

public class ElementDiff {
    /**
     * Compute the difference between large and small elements in the given arry
     *
     *
     * @return result
     */

    public int findDiff(int[] arr) {

        //Write your code here

        int largest = arr[0], smallest = arr[0];
        int result=0;

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

            if(arr[i] > largest){

                largest = arr[i];

            }

            if(arr[i] < smallest){

                smallest = arr[i];
            }
        }

        result = largest - smallest;

        return result;

    }
}
```

```
    }
}
```

Q24. Write a class FindDuplicate with a **public** method findDuplicate that takes two parameters one is arr of type int[] and second one is arg of type int and returns true if arg present more than once in the arr. The return type of findDuplicate should be boolean.

Assumptions:

1. arr is never null

Here is an example:
Enter no of elements in the array:
6
Enter elements in the array seperated by space:
999 77 77 88 54 -8
Enter the element you want to search:
77
true

q11073/FindDuplicate.jav | q11073/FindDuplicateMai

```java
package q11073;

public class FindDuplicate {
    /**
     * Find the arg element occures in the arr more than once
     *
     *
     * @return result
     */

    public boolean findDuplicate(int[] arr, int arg) {
        //Write your code here

        int count=0;

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

                if(arr[i] == arg){

                    count++;
                }
        }

        if(count > 1){

            return true;

        } else{

            return false;
        }

    }
}
```

Q25. Write a class FindDuplicate with a **public** method findDuplicate that takes two parameters one is arr of type int[] and second one is arg of type int and returns true if arg present more than once in the arr. The return type of findDuplicate should be boolean.

Assumptions:

1. arr is never null

Here is an example:
Enter no of elements in the array:
6
Enter elements in the array seperated by space:
999 77 77 88 54 -8
Enter the element you want to search:
77
true

q11073/FindDuplicate.jav q11073/FindDuplicateMai

```java
package q11073;

public class FindDuplicate {
    /**
     * Find the arg element occures in the arr more than once
     *
     *
     * @return result
     */

    public boolean findDuplicate(int[] arr, int arg) {
        //Write your code here

        int count=0;

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

                if(arr[i] == arg){

                    count++;
                }
        }

        if(count > 1){

            return true;

        } else{

            return false;
        }

    }
}
```

Q26. Write a class SumOfSameNumber with a **public** method findSumOf that takes two parameters one is arr of type int[] and second one is arg of type int and returns true if sum of all arg elements present in the arr is greater than or equal to 10.

Assumptions:

1. arr is never null

Here is an example:

Enter no of elements in the array:
6
Enter elements in the array seperated by space:
1 3 4 4 4 5
Enter the search element:
4
true

q11074/SumOfSameNum q11074/SumOfSameNum

```java
package q11074;

public class SumOfSameNumber {
    /**
     * Compute the sum of all arg elments in the arr is greater than 10 or not
     *
     *
     *
     * @return result
     */

    public boolean findSumOf(int[] arr, int arg) {
        //Write your code here

        int sum=0;

        boolean result = false;

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

            if(arr[i] == arg){

                sum += arr[i];
            }
        }

        if(sum >= 10 && arr != null){

            result = true;
            return result;

        } else {

            result = false;
            return result;

        }

    }
}
```

Q27. Write a class CountOfTwoNumbers with a **public** method compareCountOf that takes three parameters one is arr of type int[] and other two are arg1 and arg2 are of type int and returns true if count of arg1 is greater than arg2 in arr. The return type of compareCountOf should be boolean.

Assummptions:

1. arr is never null
2. arg1 and arg2 may be same

Here are an example:

Enter no of elements in the array:
6
Enter elements in the array seperated by space:
1 2 2 3 5 2
Enter the arg1 element:
2
Enter the arg2 element:
5
true
Enter no of elements in the array:
4
Enter elements in the array seperated by space:
99 -10 99 -1
Enter the arg1 element:
99
Enter the arg2 element:
99
false

q11075/CountOfTwoNum q11075/CountOfTwoNum

```java
package q11075;

public class CountOfTwoNumbers {
    /**
     * Find the count of arg1 is more than arg2 in the arr or not
     *
     *
     *
     * @return result
     */

    public boolean compareCountOf(int[] arr, int arg1, int arg2) {
        //write your code here

        boolean result = false;
        int count1=0, count2=0;

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

            if(arr[i] == arg1){
                count1++;
            }
            if(arr[i] == arg2){
                count2++;
            }
        }

        if(count1 > count2){

            result = true;

            return result;
        } else {
            result = false;

            return result;
        }

    }
}
```

Q28 . Write a class CheckAlternateNo with a **public** method checkAlternate that takes two parameters one is arr of type int[] and second one is arg of type int and returns true the element arg is present at every odd position of the arr. The return type of checkAlternate is boolean.

Assumptions:

1. arr is never null

Here is an example:

Enter no of elements in the array:
5
Enter elements in the array seperated by space:
32 65 32 84 32
Enter the arg element to find:
32
true

q11076/CheckAlternateN

```java
package q11076;

public class CheckAlternateNo {
    /**
     * Check if the arg element is present at every odd postition of the arr
     *
     *
     * @return result
     */

    public boolean checkAlternate(int[] arr, int arg) {

        //Write your code here

        boolean result = false;

        for(int i = 0; i < arr.length; i += 2){

            if(arr[i] == arg){

                result = true;

            } else{

                result = false;

            }
        }

        return result;

    }
}
```