# IRC_Java_D1_Java 1DArray_PAH_COD

**Test Summary**

- No. of Sections: 1
- No. of Questions: 10
- Total Duration: 120 min

## Section 1 - coding

**Section Summary**

- No. of Questions: 10
- Duration: 120 min

**Additional Instructions:**

None

Q1. Find largest and smallest number in an array.

**Input Format**

Input consists of n+1 integer inputs.
First line of the input describes the array size 'n',
Followed by n number of array elements.

**Output Format**

Output displays the smallest and largest number in the array.

**Sample Input**

```
5
12 4 2 5 22
```

**Sample Output**

```
smallest value: 2
largest value: 22
```

**Sample Input**

```
6
20  30  50  4  71  100
```

**Sample Output**

```
smallest value: 4
largest value: 100
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. Given an array of elements. Find two elements in the array such that their sum is equal to the given element K?

**Input Format**

The first line of the input consists of the value of n.
The next input is the array elements.
The last input is the sum.

**Output Format**

The output prints whether the array has a pair of elements with the given sum.
Refer sample output for formatting specifications.

**Sample Input**

```
6
1 4 45 6 10 -8
16
```

**Sample Output**

```
Array has two elements with given sum 16
```

**Sample Input**

```
6
1 4 45 6 10 -8
60
```

**Sample Output**

```
Array doesn't have two elements with given sum 60
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q3. **Welcome Party**

New Year is shortly arriving and the students of St. Philip’s College of Business are eager to receive the freshers for the coming year. The Welcome party for the freshers is going to be organized in a week’s time and in connection to that the College Management has ordered the students to renovate their class room block. The Class room block has N rooms in it numbered from 1 to N. Each room is currently painted in one of the red, blue or green colors. Students are given configuration of colors of their class room block by an array consisting of N values. In this array, color red will be denoted by '1', green by '2' and blue by '3'.

The Management wanted the class room block to be repainted such that each class room has same color. For painting, Students have all the 3 color paints available and mixing any 2 color paints will result into 3rd color paint i.e

- 1 + 2 = 3
- 2 + 3 = 1
- 3 + 1 = 2

For example, if a room is already painted in green color, painting that room red color, will make the color of the room blue.

Also, students have many buckets of paint of each color. Simply put, you can assume that they will not run out of paint. Students are a bit lazy, so they does not want to work much and therefore, has asked you to find the minimum number of rooms they have to repaint (possibly zero) in order to have all the rooms with same color as told by the Management. Can you please help them?

**Input Format**

First line of input contains an integer N, denoting the number of class rooms in the College’s class room black. Assume that the maximum value for N as 50.

Next line of input contains N values, denoting the current color configuration of rooms.

**Output Format**

Print the minimum number of rooms that need to be painted in order to have all the rooms painted with same color i.e red, blue or green.

**Sample Input**

```
3
1 2 1
```

**Sample Output**

```
1
```

**Sample Input**

```
3
1 1 1
```

**Sample Output**

```
0
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q4. **Welcome Party**

New Year is shortly arriving and the students of St. Philip’s College of Business are eager to receive the freshers for the coming year. The Welcome party for the freshers is going to be organized in a week’s time and in connection to that the College Management has ordered the students to renovate their class room block. The Class room block has N rooms in it numbered from 1 to N. Each room is currently painted in one of the red, blue or green colors. Students are given configuration of colors of their class room block by an array consisting of N values. In this array, color red will be denoted by '1', green by '2' and blue by '3'.

The Management wanted the class room block to be repainted such that each class room has same color. For painting, Students have all the 3 color paints available and mixing any 2 color paints will result into 3rd color paint i.e

- 1 + 2 = 3
- 2 + 3 = 1
- 3 + 1 = 2

For example, if a room is already painted in green color, painting that room red color, will make the color of the room blue.

Also, students have many buckets of paint of each color. Simply put, you can assume that they will not run out of paint. Students are a bit lazy, so they does not want to work much and therefore, has asked you to find the minimum number of rooms they have to repaint (possibly zero) in order to have all the rooms with same color as told by the Management. Can you please help them?

**Input Format**

First line of input contains an integer N, denoting the number of class rooms in the College’s class room black. Assume that the maximum value for N as 50.

Next line of input contains N values, denoting the current color configuration of rooms.

**Output Format**

Print the minimum number of rooms that need to be painted in order to have all the rooms painted with same color i.e red, blue or green.

**Sample Input**

```
3
1 2 1
```

**Sample Output**

```
1
```

**Sample Input**

```
3
1 1 1
```

**Sample Output**

```
0
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. Given an unsorted array of unique integers in the range from 1 to N+1. Find the missing element in the array without sorting the array.

**Input Format**

Input the size of the array

Input the array elements

**Output Format**

Print the missing element in an array

**Sample Input**

```
5
6 4 3 2 1
```

**Sample Output**

```
5
```

**Sample Input**

```
15
4 6 5 7 3 1 2 9 8 10 12 15 14 13 16
```

**Sample Output**

```
11
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q6. Write a program to insert an element at a specified position in the array and find the duplicate values of the array of float values.

**Input Format**

The first line of the input consists of the value of n.
The next input is the array elements.
The third input is the position.
The fourth input is the element to be inserted.

**Output Format**

The first line of the output prints the newly formed array separated by a space.
Then print the duplicate elements in consecutive lines.
Note: There is an extra space at the end of the first line of output.

**Sample Input**

```
10
1 2 3 4 5 6 1 2 9 10
2
8
```

**Sample Output**

```
1.0 8.0 2.0 3.0 4.0 5.0 6.0 1.0 2.0 9.0 10.0
1.0
2.0
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q7. Given an array of numbers. Give an algorithm for finding the first element in the array which is repeated.

**Input Format**

The first line of the input consists of the value of n.
The next input is the array elements.

**Output Format**

The output prints the first repeated element in the array.

**Sample Input**

```
7
10 5 3 5 3 4 6
```

**Sample Output**

```
The first repeating element is 5
```

**Sample Input**

```
5
1 5 6 8 7
```

**Sample Output**

```
There are no repeating elements
```

**Sample Input**

```
8
5 2 4 6 7 2 4 5
```

**Sample Output**

```
The first repeating element is 5
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q8. Given an array A of n elements. Find three elements m, n, and k in the array such that $m^2 + n^2 = k^2$?

**Input Format**

The first line of the input consists of the value of n.
The next input is the array elements.

**Output Format**

The first line of the output prints the sum (square) and the elements that correspond to the sum(squares) separated by a space.
The second line of the output prints the element that corresponds to the above answer.
Else, no such triplet exists.
Refer to the sample output for any formatting specifications.

**Sample Input**

```
5
3 4 8 6 5
```

**Sample Output**

```
25 9 16
5.0 3.0 4.0
```

**Sample Input**

```
5
3 4 8 6 12
```

**Sample Output**

```
No such triplet exists
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q9. Write a program to find the first and last occurrence of an element in a sorted array.

**Input Format**

The first line of the input consists of the value n.
Next input is the array elements.
The last input is the element.

**Output Format**

The output prints the first and last occurrence of the element separated by a space.

**Sample Input**

```
9
1 3 5 5 5 5 67 123 125
5
```

**Sample Output**

```
2 5
```

**Sample Input**

```
9
1 3 5 5 5 5 7 123 125
7
```

**Sample Output**

```
6 6
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q10. Write a program to find all pairs of elements in an array whose sum is equal to the given value. Help Guru to write a program to complete this task.

**Input Format**

Required Sum in first line.
Number of array elements in the second line.
Array elements in third line separated by space.

**Output Format**

Number pair with sum as shown in the sample output.

**Sample Input**

```
30
8
14 -15 9 16 25 45 12 8
```

**Sample Output**

```
14 + 16 =  30
-15 + 45 =  30
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

# Answer Key & Solution

## Section 1 - coding

Q1

**Test Case**

**Input**

```
10
23   34   13   64   72   90   10   15   9   27
```

**Output**

```
smallest value: 9
largest value: 90
```

**Weightage - 15**

**Input**

```
6
20   30   50   4   71   100
```

**Output**

```
smallest value: 4
largest value: 100
```

**Weightage - 15**

**Input**

```
16
20   30   50   4   71   100  23   34   13   64   72   90
```

**Output**

```
smallest value: 1
largest value: 100
```

**Weightage - 15**

**Input**

```
4
7 8 9 40
```

**Output**

```
smallest value: 7
largest value: 40
```

**Weightage - 15**

**Input**

```
5
32 6 5 4 3
```

**Output**

```
smallest value: 3
largest value: 32
```

**Weightage - 15**

**Input**

```
9
98 87 65 74 25 254 45 12 41
```

**Output**

```
smallest value: 12
largest value: 254
```

**Weightage - 15**

**Input**

```
5
12 4 2 5 22
```

**Output**

```
smallest value: 2
largest value: 22
```

**Weightage - 10**

**Sample Input**

```
5
12 4 2 5 22
```

**Sample Output**

```
smallest value: 2
largest value: 22
```

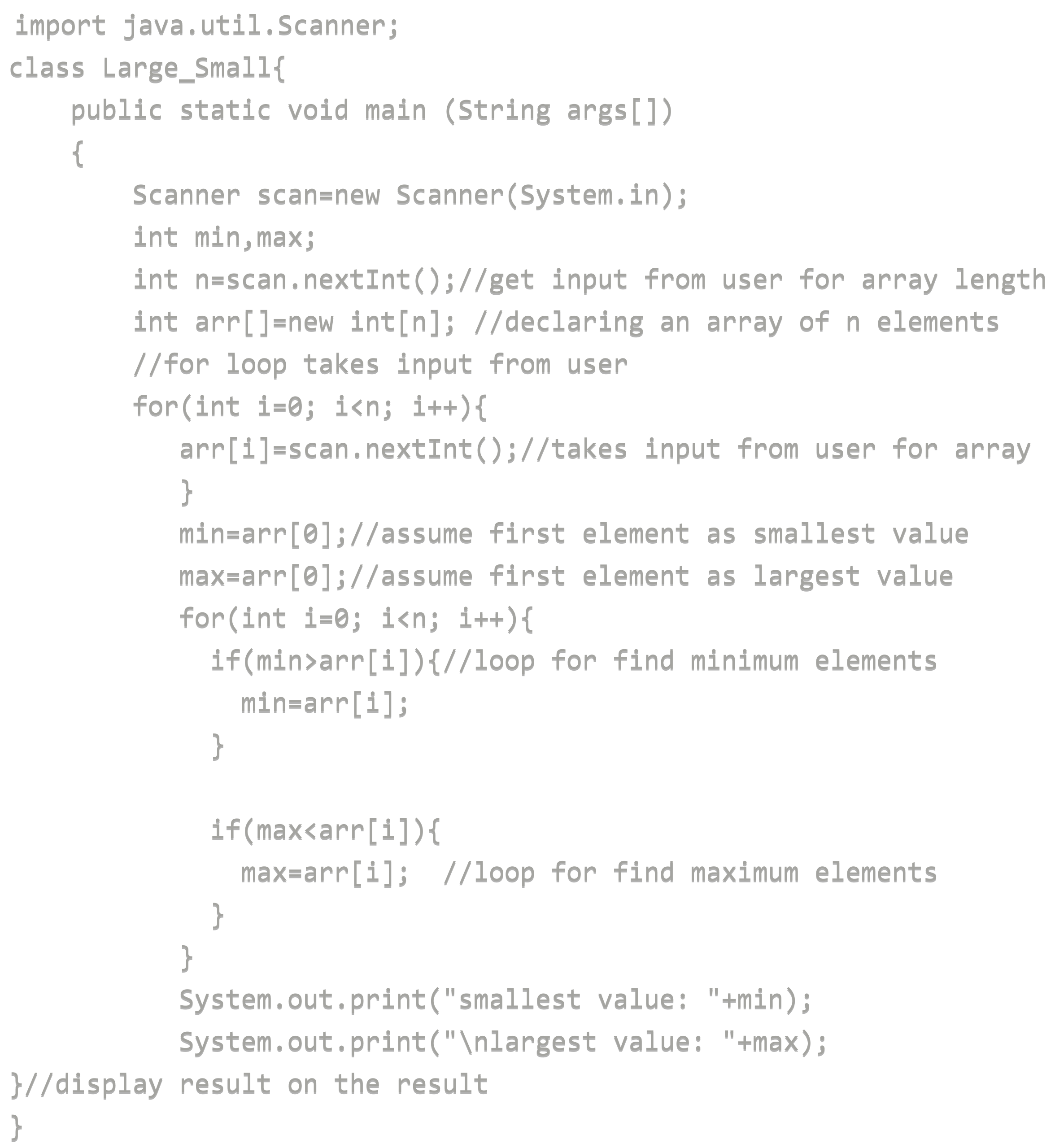
**Sample Input**

```
6
20  30  50  4  71  100
```

**Sample Output**

```
smallest value: 4
largest value: 100
```

**Solution**



```
 import java.util.Scanner;
class Large_Small{
    public static void main (String args[])
    {
        Scanner scan=new Scanner(System.in);
        int min,max;
        int n=scan.nextInt();//get input from user for array length
        int arr[]=new int[n]; //declaring an array of n elements
        //for loop takes input from user
        for(int i=0; i<n; i++){
           arr[i]=scan.nextInt();//takes input from user for array
           }
           min=arr[0];//assume first element as smallest value
           max=arr[0];//assume first element as largest value
           for(int i=0; i<n; i++){
             if(min>arr[i]){//loop for find minimum elements
               min=arr[i];
             }

             if(max<arr[i]){
               max=arr[i];  //loop for find maximum elements
             }
           }
           System.out.print("smallest value: "+min);
           System.out.print("\nlargest value: "+max);
}//display result on the result
}
```

Q2

**Test Case**

**Input**

```
8
12 35 74 96 20 2 6 8
94
```

**Output**

```
Array has two elements with given sum 94
```

**Weightage - 25**

**Input**

```
8
12 35 74 96 20 2 6 8
99
```

**Output**

```
Array doesn't have two elements with given sum 99
```

**Weightage - 25**

**Input**

```
10
12 23 45 56 78 89 14 25 36 85
81
```

**Output**

```
Array has two elements with given sum 81
```

**Weightage - 25**

**Input**

```
10
12 23 45 56 78 89 14 25 36 85
122
```

**Output**

```
Array doesn't have two elements with given sum 12
```

**Weightage - 25**

**Sample Input**

```
6
1 4 45 6 10 -8
16
```

**Sample Output**

```
Array has two elements with given sum 16
```

**Sample Input**

```
6
1 4 45 6 10 -8
60
```

**Sample Output**

```
Array doesn't have two elements with given sum 60
```

**Solution**

```
import java.util.*;
class Test {
    static boolean hasArrayTwoCandidates(
        int A[],
        int arr_size, int sum)
    {
        int l, r;
        Arrays.sort(A);
        l = 0;
        r = arr_size - 1;
        while (l < r) {
            if (A[l] + A[r] == sum)
                return true;
            else if (A[l] + A[r] < sum)
                l++;
            else
                r--;
        }
        return false;
    }
    public static void main(String args[])
    {
        int i,n,sum;
        Scanner sc = new Scanner(System.in);
        n = sc.nextInt();
        int [] arr = new int[n];
        for(i=0;i<n;i++) {
            arr[i] = sc.nextInt();
        }
        sum = sc.nextInt();
        if (hasArrayTwoCandidates(arr, n, sum))
            System.out.println("Array has two "
                            + "elements with given sum "+sum);
        else
            System.out.println("Array doesn't have "
                            + "two elements with given sum "+sum);
    }
}
```

Q3

**Test Case**

**Input**

```
3
1 2 1
```

**Output**

```
1
```

**Weightage - 10**

**Input**

```
3
1 1 1
```

**Output**

```
0
```

**Weightage - 10**

**Input**

```
8
1 2 1 1 3 2 1 1
```

**Output**

```
3
```

**Weightage - 10**

**Input**

```
16
1 2 3 2 1 3 3 1 3 1 2 2 3 3 3 3
```

**Output**

```
8
```

**Weightage - 15**

**Input**

```
26
1 2 3 1 2 2 2 1 3 2 2 1 2 2 3 2 2 2 2 3 2 2 1
```

**Output**

```
10
```

**Weightage - 15**

**Input**

```
34
1 2 1 1 1 2 1 1 1 1 3 2 1 1 1 1 2 3 3 1 1 1 1
```

**Output**

```
10
```

**Weightage - 20**

**Input**

```
50
1 2 3 1 1 2 3 1 1 1 2 3 1 1 1 1 2 1 3 1 1 1 1
```

**Output**

```
20
```

**Weightage - 20**

**Sample Input**

```
3
1 2 1
```

**Sample Output**

```
1
```

**Sample Input**

```
3
1 1 1
```

**Sample Output**

```
0
```

**Solution**

```
 import java.io.*;
import java.util.*;
class Welcomepaarty {
    public static void main(String [] args) {
        int i,j=0,n,k=0,c=0;
        Scanner sc = new Scanner(System.in);
        n=sc.nextInt();
        int a[] = new int[n];
        for(i=0;i<n;i++) {
            a[i] = sc.nextInt();
        }
        for(i=0;i<n;i++)
        {
            if(a[i]==1) {
                k++;
                }
            else if(a[i]==2) {
                j++;
                }
            else {
                c++;
                }
        }
        if(k>j)
        {
            if(k>c)
            {
                System.out.println(n-k);
            }
            else {
                System.out.println(n-c);
                }
        }
        else if(j>c) {
            System.out.println(n-j);
            }
        else {
```

```
        System.out.println(n-c);
        }

    }
}
```

Q4

**Test Case**

**Input** **Output**

```
3
1 2 1
```

```
1
```

**Weightage - 10**

**Input** **Output**

```
3
1 1 1
```

```
0
```

**Weightage - 10**

**Input** **Output**

```
8
1 2 1 1 3 2 1 1
```

```
3
```

**Weightage - 10**

**Input** **Output**

```
16
1 2 3 2 1 3 3 1 3 1 2 2 3 3 3 3
```

```
8
```

**Weightage - 15**

**Input** **Output**

```
26
1 2 3 1 2 2 2 1 3 2 2 1 2 2 3 2 2 2 2 3 2 2 1
```

```
10
```

**Weightage - 15**

**Input** **Output**

```
34
1 2 1 1 1 2 1 1 1 1 3 2 1 1 1 1 2 3 3 1 1 1 1
```

```
10
```

**Weightage - 20**

**Input** **Output**

```
50
1 2 3 1 1 2 3 1 1 1 2 3 1 1 1 1 2 1 3 1 1 1 1
```

```
20
```

**Weightage - 20**

**Sample Input**

```
3
1 2 1
```

**Sample Output**

```
1
```

**Sample Input**

```
3
1 1 1
```

**Sample Output**

```
0
```

**Solution**

```
 import java.io.*;
import java.util.*;
class Welcomepaarty {
    public static void main(String [] args) {
        int i,j=0,n,k=0,c=0;
        Scanner sc = new Scanner(System.in);
        n=sc.nextInt();
        int a[] = new int[n];
        for(i=0;i<n;i++) {
            a[i] = sc.nextInt();
        }
        for(i=0;i<n;i++)
        {
            if(a[i]==1) {
                k++;
                }
            else if(a[i]==2) {
                j++;
                }
            else {
                c++;
                }
        }
        if(k>j)
        {
            if(k>c)
            {
                System.out.println(n-k);
            }
            else {
                System.out.println(n-c);
                }
        }
        else if(j>c) {
            System.out.println(n-j);
            }
        else {
            System.out.println(n-c);
            }

    }
}
```

Q5

**Test Case**

**Input**

```
6
1 2 4 5 6 7
```

**Output**

```
3
```

**Weightage - 20**

**Input**

```
5
1 2 4 5 6
```

**Output**

```
3
```

**Weightage - 20**

**Input**

```
20
2 5 6 7 9 8 1 3 11 13 15 17 19 20 12 14 16 18
```

**Output**

```
4
```

**Weightage - 30**

**Input**

```
10
10 6 5 7 1 2 3 4 9 11
```

**Output**

```
8
```

**Weightage - 30**

**Sample Input**

```
5
6 4 3 2 1
```

**Sample Output**

```
5
```

**Sample Input**

```
15
4 6 5 7 3 1 2 9 8 10 12 15 14 13 16
```

**Sample Output**

```
11
```

**Solution**

```
 import java.util.*;
class Main
{
    public static void main(String[] args) {
    int sum = 0;
    int idx = -1;
    int num;
    Scanner sc=new Scanner(System.in);
    num=sc.nextInt();
    int arr[]=new int[num];
    for(int i=0;i<num;i++)
    {
        arr[i]=sc.nextInt();
        sum += arr[i];
    }
    int total = (num+2)*(num+1)/ 2;

    System.out.println("" + (total - sum));
    }
}
```

Q6

**Test Case**

**Input**

```
10
1 2 3 4 5 6 1 2 9 10
2
8
```

**Output**

```
1.0 8.0 2.0 3.0 4.0 5.0 6.0 1.0 2.0 9.0 10.0
1.0
2.0
```

**Weightage - 25**

**Input**

```
12
12 25 44 85 76 95 20 22 44 76 20 99
5
88
```

**Output**

```
12.0 25.0 44.0 85.0 88.0 76.0 95.0 20.0 22.0 44.
44.0
76.0
```

**Weightage - 25**

**Input**

```
15
123 456 789 987 654 321 123 456 789 147 248 369
10
```

**Output**

```
123.0 456.0 789.0 987.0 654.0 321.0 123.0 456.0 7
123.0
456.0
```

**Weightage - 25**

**Input**

```
20
10 20 30 40 50 60 70 80 90 100 110 120 130 140
20
```

**Output**

```
10.0 20.0 30.0 40.0 50.0 60.0 70.0 80.0 90.0 100.
180.0
```

**Weightage - 25**

**Sample Input**

```
10
1 2 3 4 5 6 1 2 9 10
2
8
```

**Sample Output**

```
1.0 8.0 2.0 3.0 4.0 5.0 6.0 1.0 2.0 9.0 10.0
1.0
2.0
```

**Solution**

```
import java.io.*;
import java.util.*;
class main {
    public static void main(String [] args) {
        int i,j,n,pos,ele;
        Scanner sc = new Scanner(System.in);
        n = sc.nextInt();
        double arr[] = new double[n+1];
        for(i=0;i<n;i++) {
            arr[i] = sc.nextDouble();
        }
        pos = sc.nextInt();
        ele = sc.nextInt();
        n++;
         for (i = n-1; i >= pos; i--) {
        arr[i] = arr[i - 1];
         }
        arr[pos - 1] = ele;
        for(i=0;i<n;i++) {
            System.out.print(arr[i]+" ");
        }
        System.out.println();
        for(i=0;i<n;i++) {
            for(j=i+1;j<n;j++) {
                if(arr[i] == arr[j]) {
                    System.out.println(arr[j]);
                }
            }
        }
    }
}
```

Q7

**Test Case**

**Input**

```
7
10 5 3 5 3 4 6
```

**Output**

```
The first repeating element is 5
```

**Weightage - 25**

**Input**

```
5
8 4 6 2 3
```

**Output**

```
There are no repeating elements
```

**Weightage - 25**

**Input**

```
8
8 2 4 6 7 2 4 8
```

**Output**

```
The first repeating element is 8
```

**Weightage - 25**

**Input**

```
10
4 2 4 6 8 2 1 4 5 6
```

**Output**

```
The first repeating element is 4
```

**Weightage - 25**

**Sample Input**

```
7
10 5 3 5 3 4 6
```

**Sample Output**

```
The first repeating element is 5
```

**Sample Input**

```
5
1 5 6 8 7
```

**Sample Output**

```
There are no repeating elements
```

**Sample Input**

```
8
5 2 4 6 7 2 4 5
```

**Sample Output**

```
The first repeating element is 5
```

**Solution**

```java
import java.util.*;
class Main
{
    static void printFirstRepeating(int arr[])
    {
        int min = -1;
        HashSet<Integer> set = new HashSet<>();
        for (int i=arr.length-1; i>=0; i--)
        {
            if (set.contains(arr[i]))
                min = i;

            else
                set.add(arr[i]);
        }
        if (min != -1)
        System.out.println("The first repeating element is " + arr[min]);
        else
        System.out.println("There are no repeating elements");
    }
    public static void main (String[] args) throws java.lang.Exception
    {
        int i,n;
        Scanner sc = new Scanner(System.in);
        n = sc.nextInt();
        int [] arr = new int[n];
        for(i=0;i<n;i++) {
            arr[i] = sc.nextInt();
        }
        printFirstRepeating(arr);
    }
}
```

Q8

**Test Case**

**Input**

```
5
3 4 8 6 5
```

**Output**

```
25 9 16
5.0 3.0 4.0
```

**Weightage - 25**

**Input**

```
5
3 4 8 6 12
```

**Output**

```
No such triplet exists
```

**Weightage - 25**

**Input**

```
8
15 20 42 85 75 96 35 25
```

**Output**

```
625 225 400
25.0 15.0 20.0
```

**Weightage - 25**

**Input**

```
8
12 34 56 78 89 34 67 15
```

**Output**

```
No such triplet exists
```

**Weightage - 25**

**Sample Input**

```
5
3 4 8 6 5
```

**Sample Output**

```
25 9 16
5.0 3.0 4.0
```

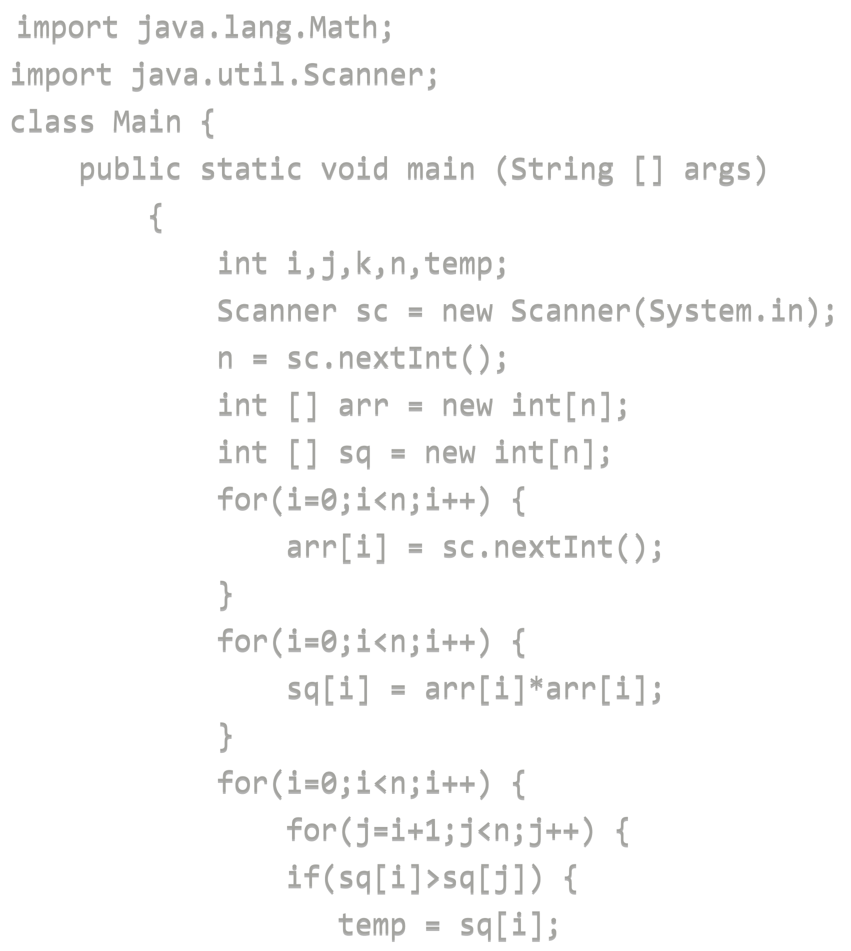
**Sample Input**

```
5
3 4 8 6 12
```

**Sample Output**

```
No such triplet exists
```

**Solution**



```
 import java.lang.Math;
import java.util.Scanner;
class Main {
    public static void main (String [] args)
        {
            int i,j,k,n,temp;
            Scanner sc = new Scanner(System.in);
            n = sc.nextInt();
            int [] arr = new int[n];
            int [] sq = new int[n];
            for(i=0;i<n;i++) {
                arr[i] = sc.nextInt();
            }
            for(i=0;i<n;i++) {
                sq[i] = arr[i]*arr[i];
            }
            for(i=0;i<n;i++) {
                for(j=i+1;j<n;j++) {
                if(sq[i]>sq[j]) {
                   temp = sq[i];
                   sq[i] = sq[j];
                   sq[j] = temp;
                }
            }
            }
            int flag =0;
             for ( i = n - 1; i >= 0; i--) {
                 j = 0;
                 k = i - 1;
                while (j < k) {
                    if (sq[i] == sq[j] + sq[k]) {
                        System.out.println(sq[i]+" "+sq[j]+" "+sq[k]);
                        System.out.println(Math.sqrt(sq[i])+" "+Math.sqrt(sq[j])+" "+Math.sqrt(sq[k]));
                        flag = 1;
                        break;
                    }
                     else if (sq[i] > sq[j] + sq[k])
                        j += 1;
                    else
                        k -= 1;
                }
            }
            if(flag == 0)
                System.out.println("No such triplet exists");
        }
}
```

Q9

**Test Case**

**Input**

```
9
1 3 5 5 5 5 67 123 125
5
```

**Output**

```
2 5
```

**Weightage - 25**

**Input**

```
9
1 3 5 5 5 5 7 123 125
7
```

**Output**

```
6 6
```

**Weightage - 25**

**Input**

```
10
1 2 2 2 2 3 4 7 8 8
8
```

**Output**

```
8 9
```

**Weightage - 25**

**Input**

```
15
1 2 2 3 4 5 5 6 6 8 123 125 168 178 220
8
```

**Output**

```
9 9
```

**Weightage - 25**

**Sample Input**

```
9
1 3 5 5 5 5 67 123 125
5
```

**Sample Output**

```
2 5
```

**Sample Input**

```
9
1 3 5 5 5 5 7 123 125
7
```

**Sample Output**

```
6 6
```

**Solution**

```
import java.io.*;
import java.util.*;
class Main {
    public static void findFirstAndLast(int arr[], int x)
    {
        int n = arr.length;
        int first = -1, last = -1;
        for (int i = 0; i < n; i++) {
            if (x != arr[i])
                continue;
            if (first == -1)
                first = i;
            last = i;
        }
        if (first != -1) {
            System.out.println(first+" "+last);
        }
        else
            System.out.println("Not Found");
    }

    public static void main(String[] args)
    {
        int i,n,x;
        Scanner sc = new Scanner(System.in);
        n = sc.nextInt();
        int arr[] = new int[n];
        for(i=0;i<n;i++) {
            arr[i] = sc.nextInt();
        }
        x = sc.nextInt();
        findFirstAndLast(arr, x);
    }
}
```

Q10 **Test Case**

**Input**

```
20
8
14 -15 9 16 25 45 12 8
```

**Output**

```
12 + 8 =  20
```

**Weightage - 10**

**Input**

```
50
10
1 2 3 40 5 6 7 8 9 10
```

**Output**

```
40 + 10 =  50
```

**Weightage - 10**

**Input**

```
89
5
88 88 87 1 2
```

**Output**

```
88 + 1 =  89
88 + 1 =  89
87 + 2 =  89
```

**Weightage - 10**

**Input**

```
9
10
1 1 1 1 1 1 1 1 1 8
```

**Output**

```
1 + 8 =  9
1 + 8 =  9
1 + 8 =  9
1 + 8 =  9
```

**Weightage - 10**

**Input**

```
5
25
1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2
```

**Output**

```
1 + 4 =  5
1 + 4 =  5
1 + 4 =  5
1 + 4 =  5
```

**Weightage - 15**

**Input**

```
10
50
1 2 3 9 8 1 2 3 9 8 1 2 3 9 8 1 2 3 9 8 1 2
```

**Output**

```
1 + 9 =  10
1 + 9 =  10
1 + 9 =  10
1 + 9 =  10
```

**Weightage - 15**

**Input**

```
5
50
-2 -5 10 7 6 -2 -5 10 7 6 -2 -5 10 7 6 -2 -5 1
```

**Output**

```
-2 + 7 =  5
-2 + 7 =  5
-2 + 7 =  5
-2 + 7 =  5
```

**Weightage - 30**

**Sample Input**

```
30
8
14 -15 9 16 25 45 12 8
```

**Sample Output**

```
14 + 16 =  30
-15 + 45 =  30
```

**Solution**

```
import java.util.Scanner;
```

```
class PairOfInt {
static void  pairs_value(int inputArray[], int inputNumber)
  {
   for (int i =  0; i < inputArray.length; i++)
     {
        for (int j  = i+1; j < inputArray.length; j++)
          {
            if(inputArray[i]+inputArray[j] == inputNumber)
            {
            System.out.println(inputArray[i]+" + "+inputArray[j]+" =  "+inputNumber);
            }
          }
     }
  }

  public static void  main(String[] args)
  {
    Scanner myObj  = new Scanner(System.in);
    int sum = myObj.nextInt();
    int n = myObj.nextInt();
    int array1[];
    array1 =new int[n];
    for(int i=0;i<n;i++)
        array1[i]=myObj.nextInt();

    pairs_value(array1, sum);

  }
}
```