Find all Combination

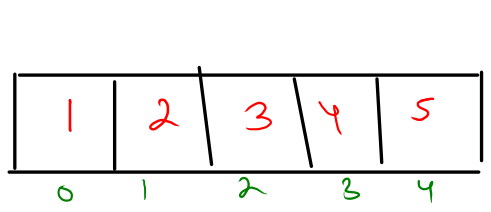
Given condition is that the array contains all the unique elements. Then take the sum as an integer input and print all the combinations of the pairs that add up to the given sum.

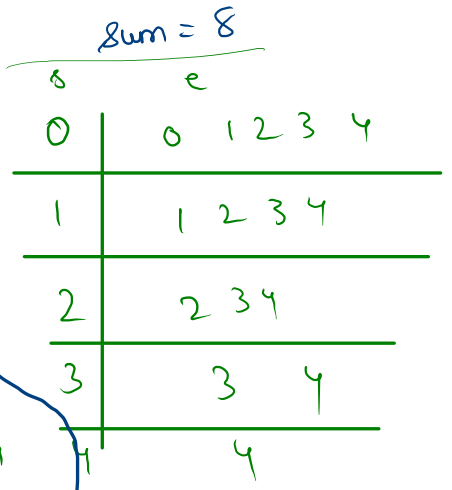
Sample Input 0

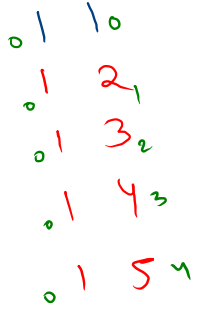
```
5
1 2 3 4 5
8
```

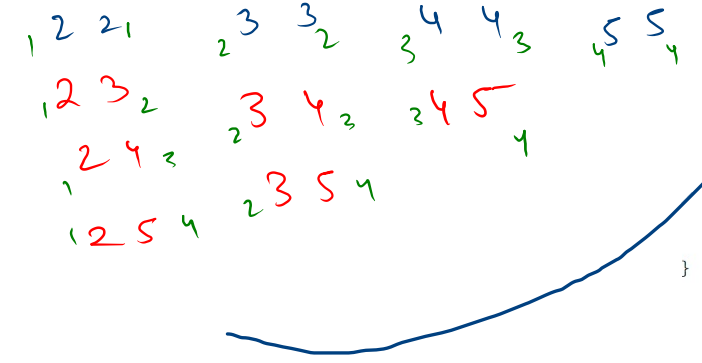
Sample Output 0

```
3 5
4 4
```



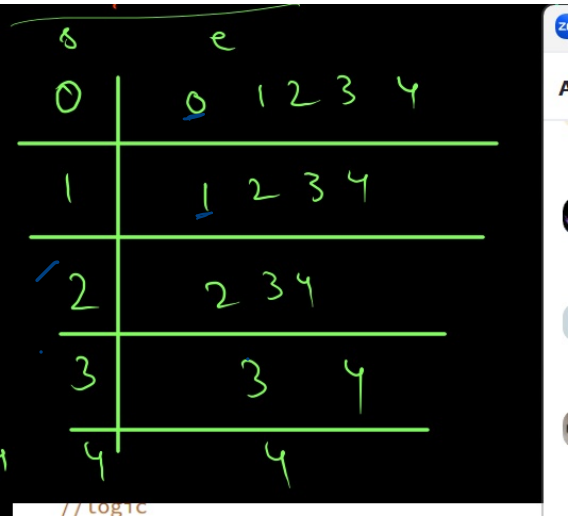


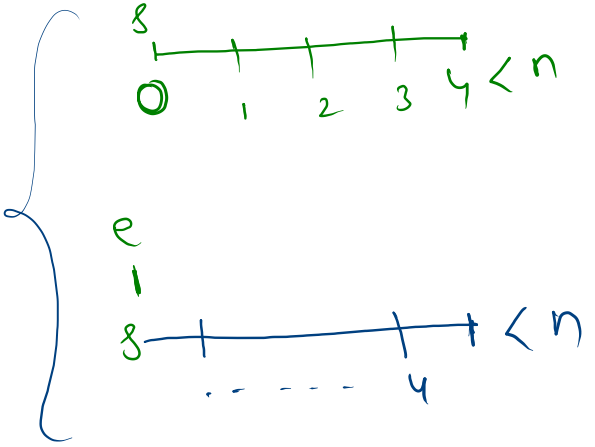




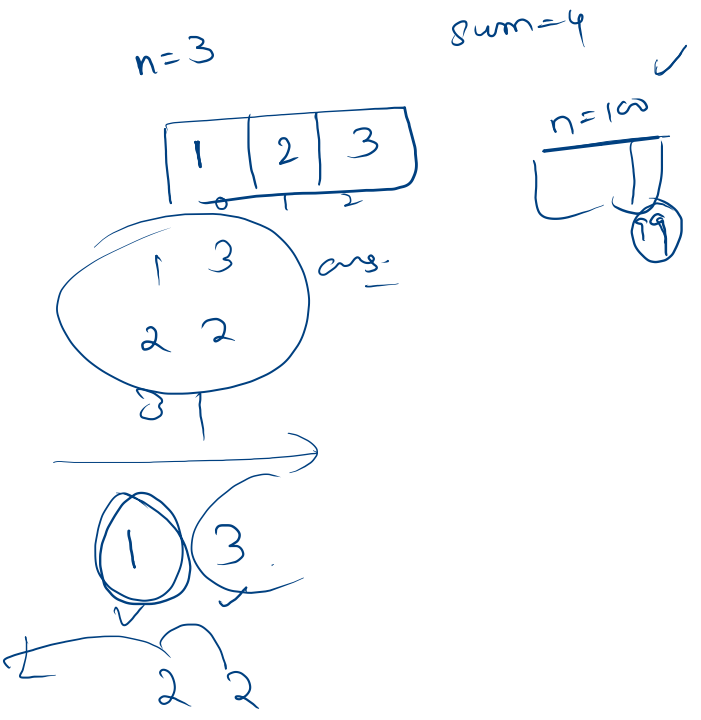
```
//Logic
for(int s = 0; s < n-1; s++){
    for(int e = s + 1; e < n; e++){
        System.out.println(A[s] + " " + A[e]);
    }
}
}
```

n=5





```java
import java.io.*;
import java.util.*;

public class Solution {

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

        //print all pairs
        for(int s = 0; s < n; s++){
            for(int e = s; e < n; e++){
                if(A[s] + A[e] == sum){
                    System.out.println(A[s] + " " + A[e]);
                }
            }
        }

    }
}
```



Greater Than Me

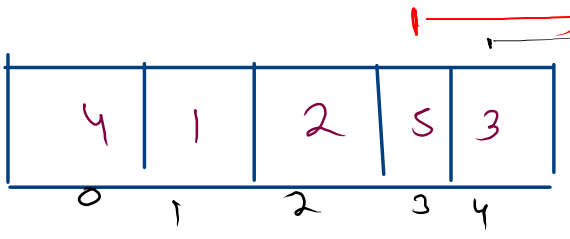
Given an **array** then for each index print the **count** of the elements which are strictly **greater than the element present at that index**.

eg $n = 5$

4 1 2 5 3

ans → 1 3 2 0 0

How many no. of right are greater than



1 3 2 0 0

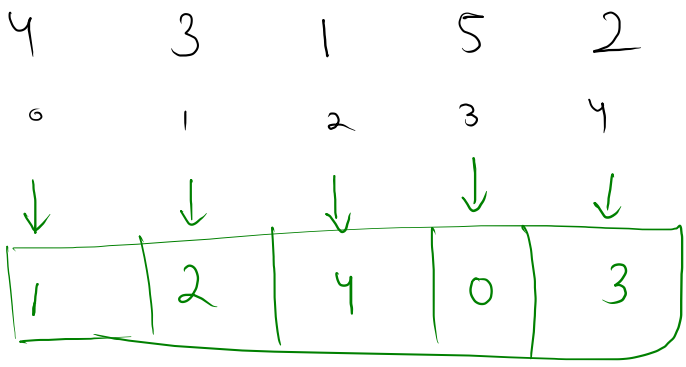
4	1	2	5	3
0	1	2	3	4

i

i \ j	1	2	3	4
0	2	3	4	
1	3	4		
2	4			
3				

$j = [\, i+1 \quad < n \,]$

Greater than Me-



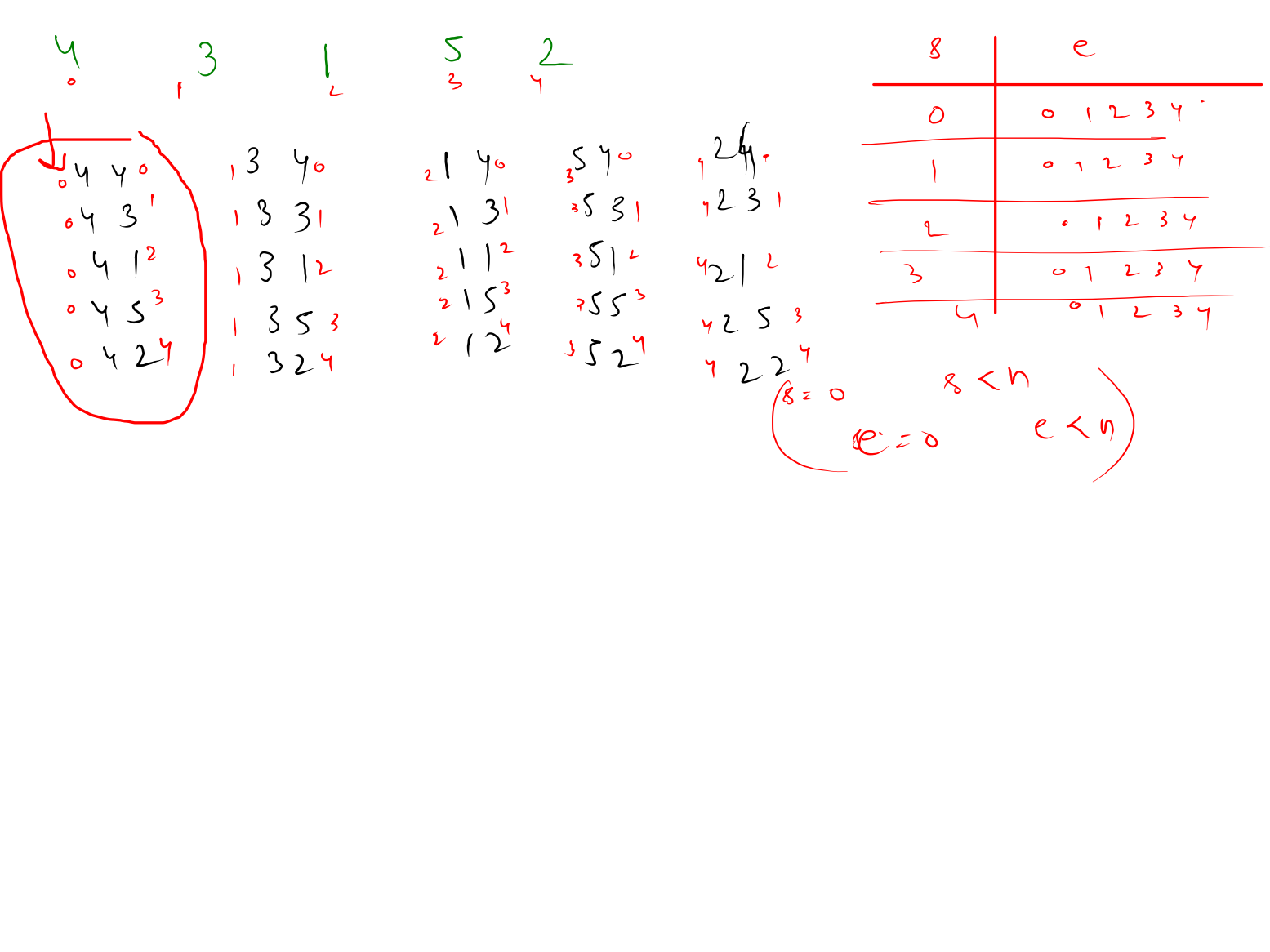
4<4
4<3
4<1
4<5
4<2
#1

3<4 ✓
3<3
3<1
3<5 ✓
3<2
#2

1<4
1<3
1<1
1<5
1<2
#4

5<4
5<3
5<1
5<5
5<2
0

2<4
2<3
2<1
2<5
2<2
#3


4 3 1 5 2
8 e
0 0 1 2 3 4
1 0 1 2 3 4
2 0 1 2 3 4
3 0 1 2 3 4
4 0 1 2 3 4
8 < n
e < n
8 = 0
e = 0

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        //logic
        for(int i = 0; i < n; i++){
            int count = 0;
            for(int j = 0; j < n; j++){
                if(A[i] < A[j]){
                    count++;
                }
            }
            System.out.print(count + " ");
        }
    }
}
```

Greater At Right

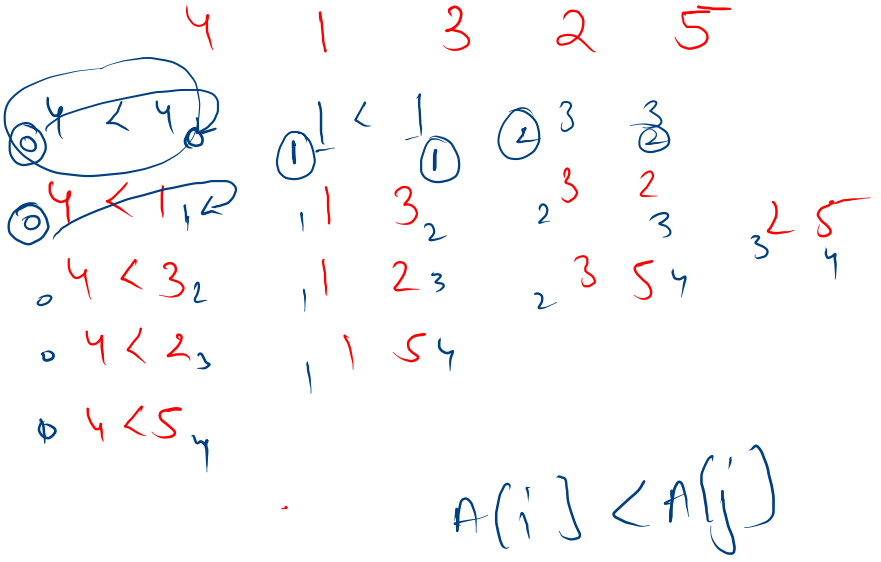
Given an array, for each index, **replace** the element with the **count** of greater elements present to the right of that element.

Sample Input 0

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

Sample Output 0

```
4 3 2 1 0
```



Language: Java 8

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        //logic
        for(int i = 0; i < n; i++){
            int count = 0;
            for(int j = i+1; j < n; j++){
                if(A[i] < A[j]){
                    count++;
                }
            }
            System.out.print(count + " ");
        }
    }
}
```

Max Count 3

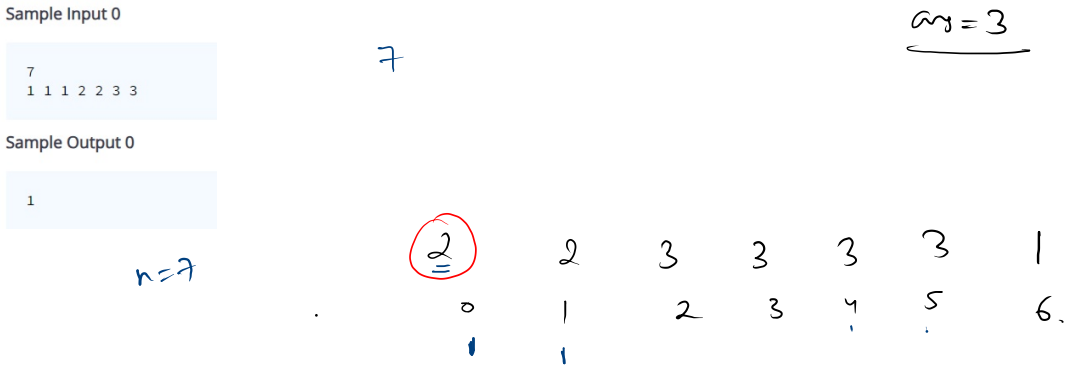
Take an array of size n with integer elements. And **Print** an element in the array which occurs for the **maximum** number of times.

Sample Input 0

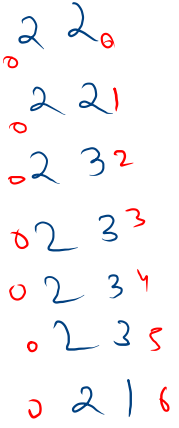
```
7
1 1 1 2 2 3 3
```

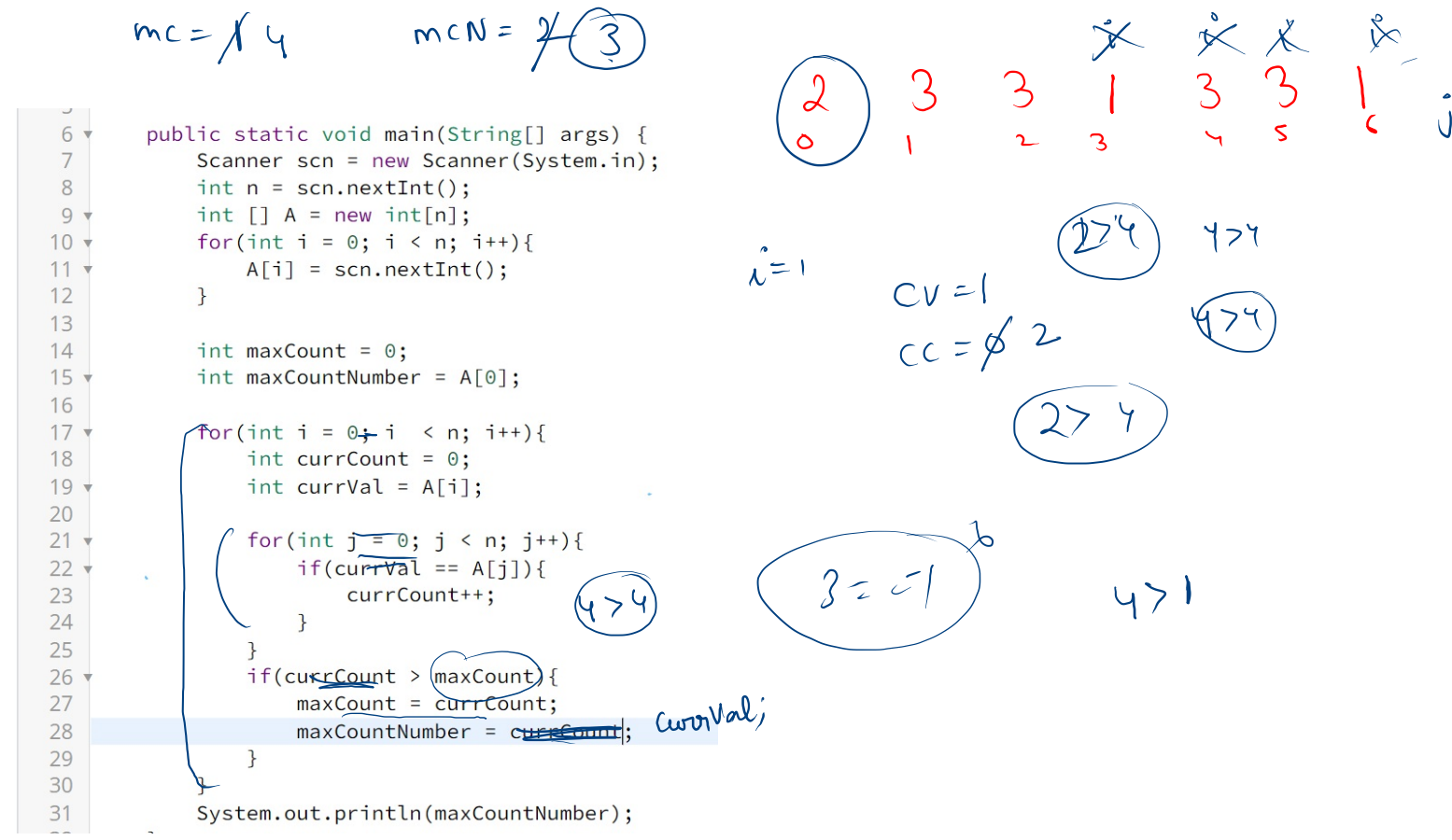
Sample Output 0

```
1
```

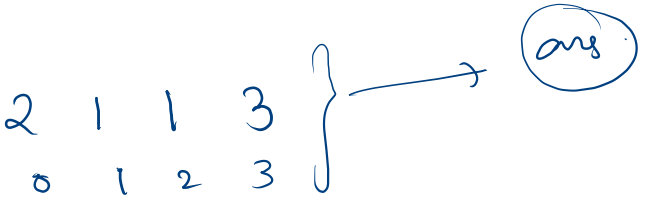


```java
public class Solution {

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

        int maxCount = 0;
        int maxCountNumber = A[0];

        for(int i = 0; i  < n; i++){
            int count = 0;
            for(int j = 0; j < n; j++){
                if(A[i] == A[j]){
                    count++;
                }
            }
            if(count > maxCount){
                maxCount = count;
                maxCountNumber = A[i];
            }
        }
        System.out.println(maxCountNumber);
    }
}
```



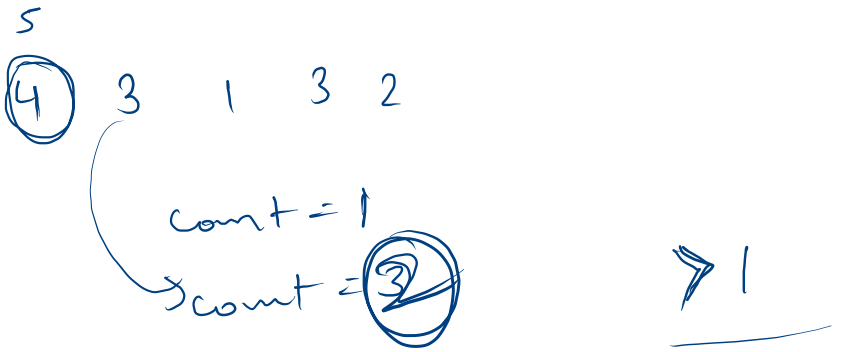

mc = 1 4
mcN = 2 3
2 3 3 1 3 3 1
0 1 2 3 4 5 6
i = 1
cv = 1
cc = 0 2
2 > 4
4 > 4
2 > 4
3 = -1
4 > 1
public static void main(String[] args) {
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
int [] A = new int[n];
for(int i = 0; i < n; i++){
A[i] = scn.nextInt();
}
int maxCount = 0;
int maxCountNumber = A[0];
for(int i = 0; i < n; i++){
int currCount = 0;
int currVal = A[i];
for(int j = 0; j < n; j++){
if(currVal == A[j]){
currCount++;
}
}
if(currCount > maxCount){
maxCount = currCount;
maxCountNumber = currVal;
}
}
System.out.println(maxCountNumber);

```java
import java.io.*;
import java.util.*;

public class Solution {

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

        int maxCount = 0;
        int maxCountNumber = A[0];

        for(int i = 0; i  < n; i++){
            int currCount = 0;
            int currVal = A[i];

            for(int j = 0; j < n; j++){
                if(currVal == A[j]){
                    currCount++;
                }
            }
            if(currCount > maxCount){
                maxCount = currCount;
                maxCountNumber = currVal;
            }
        }
        System.out.println(maxCountNumber);
    }
}
```



Find Duplicate 3

Take an array of size **n** with integer input. And Print **"true"** if the array contains a duplicate element and print **"false"**. if the array doesn't contain a duplicate element.



```java
public class Solution {
    public static boolean duplicateCheck(int [] A){
        int n = A.length;
        for(int i = 0;  i < n; i++){
            int currVal = A[i];
            int currCount = 0;
            for(int j = 0; j < n; j++){
                if(currVal == A[j]){
                    currCount++;
                }
            }

            if(currCount > 1){
                return true;
            }
        }
        return false;
    }

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