

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

0 1 10

1 2 1

0 1 3 2

1 4 3

0 1 5 4

n=5

1	2	3	4	5
0	1	2	3	4

sum = 8

s	e	
0		0 1 2 3 4
1		1 2 3 4
2		2 3 4
3		3 4
4		4

1 2 2 1

2 3 3 2

3 4 4 3

4 5 5 4

1 2 3 2

2 3 4 3

3 4 5 4

1 2 4 3

2 3 5 4

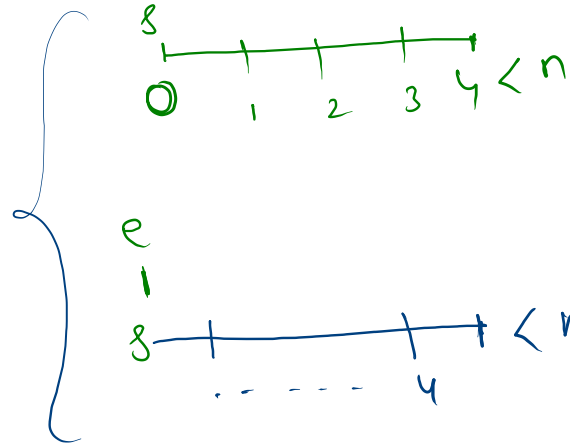
1 2 5 4

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

$h=5$

s	e				
0	0	1	2	3	4
1	1	2	3	4	
2	2	3	4		
3	3	4			
4	4				

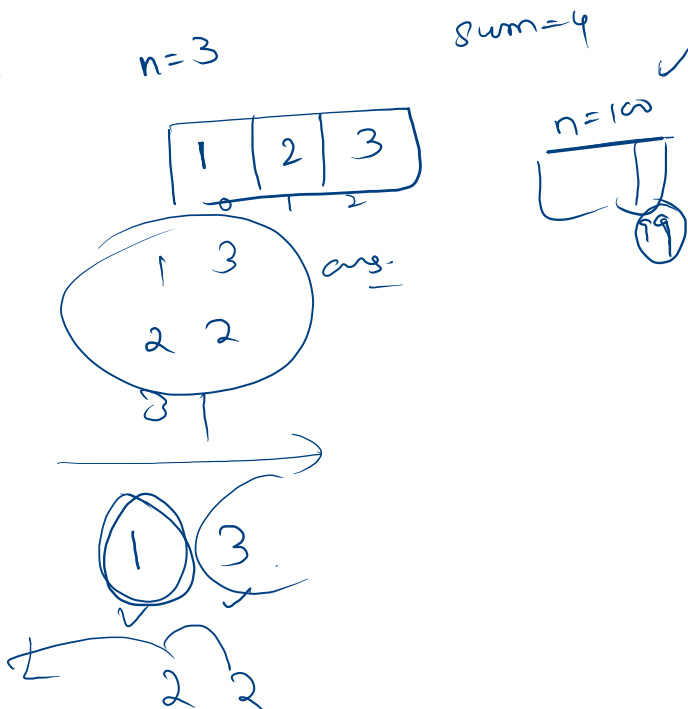
//logic



```

1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         int [] A = new int[n];
10        for(int i = 0; i < n; i++){
11            A[i] = scn.nextInt();
12        }
13        int sum = scn.nextInt();
14
15        //print all pairs
16        for(int s = 0; s < n; s++){
17            for(int e = s; e < n; e++){
18                if(A[s] + A[e] == sum){
19                    System.out.println(A[s] + " " + A[e]);
20                }
21            }
22        }
23    }
24 }
25 }

```



Greater Than Me

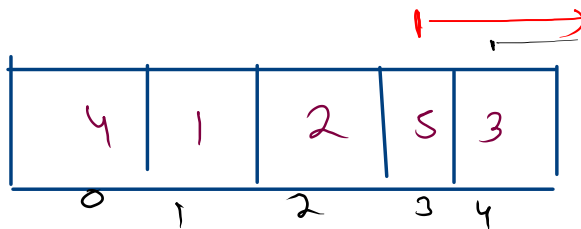
Given an **array** then for each index print the **count** of the elements which are strictly **greater than the element** present at that index.

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

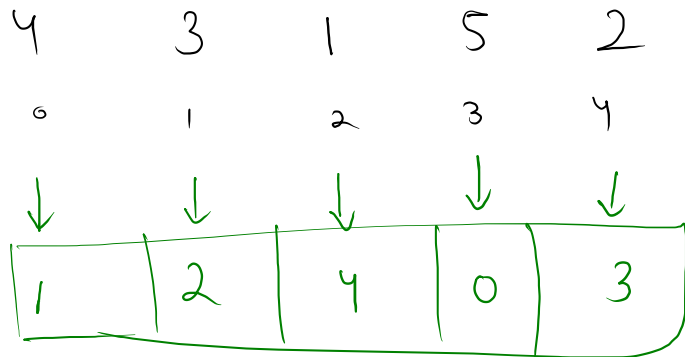


4 1 2 5 3
1 2 3 4
i

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

$$j = [i+1, n]$$

Greater than Me.



$$4 < 4$$

$$4 < 3$$

$$4 < 1$$

$$4 < 5$$

$$4 < 2$$

#1

$$3 < 4 \checkmark$$

$$3 < 3$$

$$3 < 1$$

$$3 < 5 \checkmark$$

$$3 < 2$$

#2

$$1 < 4$$

$$1 < 3$$

$$1 < 1$$

$$1 < 5$$

$$1 < 2$$

#4

$$5 < 4$$

$$5 < 3$$

$$5 < 1$$

$$5 < 5$$

$$5 < 2$$

0

$$2 < 4$$

$$2 < 3$$

$$2 < 1$$

$$2 < 5$$

$$2 < 2$$

$$2 < 2$$

#3

4 3 1

5 2

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

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

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

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

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

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

$8 = 0$
 $e = 0$
 $8 < n$
 $e < n$

```
1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         int [] A = new int[n];
10        for(int i = 0; i < n; i++){
11            A[i] = scn.nextInt();
12        }
13        //logic
14        for(int i = 0; i < n; i++){
15            int count = 0;
16            for(int j = 0; j < n; j++){
17                if(A[i] < A[j]){
18                    count++;
19                }
20            }
21            System.out.print(count + " ");
22        }
23    }
24 }
```


Greater At Right

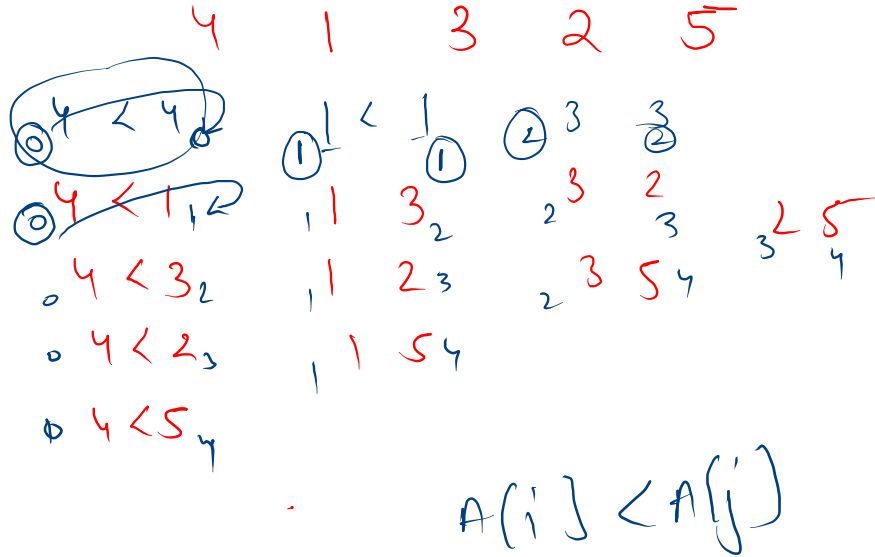
Given an array, for each index, **replace** the element with the **count** of greater elements present to the right of that element.

Sample Input 0

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

Sample Output 0

```
4 3 2 1 0
```



Language: Java 8

```
1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         int [] A = new int[n];
10        for(int i = 0; i < n; i++){
11            A[i] = scn.nextInt();
12        }
13        //logic
14        for(int i = 0; i < n; i++){
15            int count = 0;
16            for(int j = i+1; j < n; j++){
17                if(A[i] < A[j]){
18                    count++;
19                }
20            }
21            System.out.print(count + " ");
22        }
23    }
24 }
```

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
```

7

ans = 3

Sample Output 0

```
1
```

n=7

2

2

3

3

3

3

1

0

1

2

3

4

5

6

```
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         int [] A = new int[n];
10        for(int i = 0; i < n; i++){
11            A[i] = scn.nextInt();
12        }
13
14        int maxCount = 0;
15        int maxCountNumber = A[0];
16
17        for(int i = 0; i < n; i++){
18            int count = 0;
19            for(int j = 0; j < n; j++){
20                if(A[i] == A[j]){
21                    count++;
22                }
23            }
24            if(count > maxCount){
25                maxCount = count;
26                maxCountNumber = A[i];
27            }
28        }
29        System.out.println(maxCountNumber);
30    }
31 }
```

2 2 0
2 2 1
2 3 2
2 3 3
2 3 4
2 3 5
2 1 6

mc = 4

mcN = 3

2
0

3
1

3
2

1
3

3
4

3
5

1
6

```
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);
```

i=1

CV=1

CC=0 2

2 > 4

4 > 4

4 > 4

2 > 4

3 = 5

4 > 1

4 > 4

```

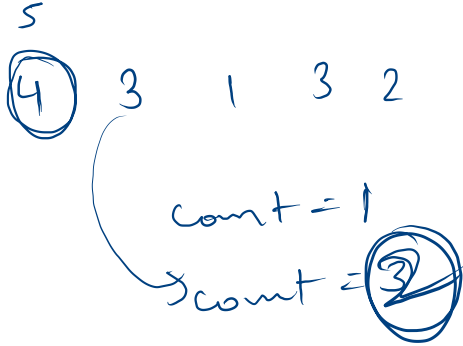
1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         int [] A = new int[n];
10        for(int i = 0; i < n; i++){
11            A[i] = scn.nextInt();
12        }
13
14        int maxCount = 0;
15        int maxCountNumber = A[0];
16
17        for(int i = 0; i < n; i++){
18            int currCount = 0;
19            int currVal = A[i];
20
21            for(int j = 0; j < n; j++){
22                if(currVal == A[j]){
23                    currCount++;
24                }
25            }
26            if(currCount > maxCount){
27                maxCount = currCount;
28                maxCountNumber = currVal;
29            }
30        }
31        System.out.println(maxCountNumber);
32    }
33 }

```

2 1 1 3 } → ans
 0 1 2 3

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.



> 1

dry last (2)

n = 1

7

0
0
0
0

CV = 7
CC = 1

1 > 1

2

```
3
4 public class Solution {
5     public static boolean duplicateCheck(int [] A){
6         int n = A.length;
7         for(int i = 0; i < n; i++){
8             int currVal = A[i];
9             int currCount = 0;
10            for(int j = 0; j < n; j++){
11                if(currVal == A[j]){
12                    currCount++;
13                }
14            }
15
16            if(currCount > 1){
17                return true;
18            }
19        }
20        return false;
21    }
22
23    public static void main(String[] args) {
24        Scanner scn = new Scanner(System.in);
25        int n = scn.nextInt();
26        int [] A = new int[n];
27        for(int i = 0; i < n; i++){
28            A[i] = scn.nextInt();
29        }
30
31    }
```