Greater Than Me

(Permutation with/ without Repetation)

↳ for each index, count greater elements than myself

$$
\begin{array}{rcccccc}
 & & & & i & & \\
 & & & & \downarrow & & \\
arr = & [& 5, & 3, & -2, & 6, & 4 \,] \\
 & & 0 & 1 & 2 & 3 & 4
\end{array}
$$

arr = [1 , 3 , 4 , 0 , 2] ans

greater   5 :- $\boxed{5}$, 6, 7, ... ∞

stricky greater   5 :- 6, 7, 8, ... ∞

```
1) ──────────→
    for( int i=0; i<n; i++){
2) ──────────→
        for(int j=0; j<n; j++){
3) ──────────→
            if( arr[j] > arr[i] ){
                count++;
            }
        }
    }
```

Note:- arr[i] = myself
arr[j] = other

code

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

    greaterThanMe(arr, n);
}
public static void greaterThanMe(int[] arr, int n) {
    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = 0; j < n; j++) {
            if ( arr[j] > arr[i] ) {
                count++;
            }
        }
        System.out.print(count + " ");
    }
}
```

i ↓

arr = [3 , -2 , 4 , 1]

$$\begin{array}{cccc} 3 & -2 & 4 & 1 \\ 0 & 1 & 2 & 3 \end{array}$$

↑ j

count = ~~0~~ 1
i = 0 , j = 0 (3 > 3)
j = 1 (-2 > 3)
j = 2 (4 > 3) ✓
j = 3 (1 > 3)

i = 2
count = 0
j = 0 , (3 > 4) ×
j = 1 , (-2 > 4) ×
j = 2 , (4 > 4) ×
j = 3 , (1 > 4) ×

count = ~~0~~ ~~1~~ ~~2~~ 3
i = 1 , j = 0 (3 > -2) ✓
j = 1 (-2 > -2)
j = 2 (4 > -2) ✓
j = 3 (1 > -2) ✓

i = 3
count = ~~0~~ ~~1~~ 2
j = 0 , (3 > 1) ✓
j = 1 , (-2 > 1) ×
j = 2 , (4 > 1) ✓
j = 3 , (1 > 1) ×

o/p

```
1
3
0
2
```

Greater At Right

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

    greaterThanMe(arr, n);
}
public static void greaterThanMe(int[] arr, int n) {
    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = i + 1; j < n; j++) {
            if ( arr[j] > arr[i] ) {
                count++;
            }
        }
        System.out.print(count + " ");
    }
}
```

maximum difference between the two elements (Imp)

↳ find maximum diff. in a pair

↳ larger element should be on the right side

ans = ~~-∞~~ ~~1~~ 8

pairs

2,3 = 1
2,10 = 8
2,6 = 4
2,4 = 2
2,8 = 6
3,10 = 7
3,6 = 3
3,4 = 1
3,8 = 5
6,8 = 2
4,8 = 4

n = 7

arr =

0	1	2	3	4	5	6
2	3	10	6	4	8	1

faith :- arr[i] = myself (smaller element)

arr[j] = other (larger element)

Code

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

    int ans = maxDiff(arr, n);
    System.out.println(ans);
}
public static int maxDiff(int[] arr, int n) {
    int ans = Integer.MIN_VALUE;
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if ( arr[j] > arr[i] ) {
                int diff = arr[j] - arr[i];
                if ( diff > ans ) {
                    ans = diff;
                }
            }
        }
    }
    return ans;
}
```

Comb

Max Count 3

$$arr = [\underset{0}{2},\ \underset{1}{1},\ \underset{2}{4},\ \underset{3}{2},\ \underset{4}{2},\ \underset{5}{1},\ \underset{6}{4},\ \underset{7}{2},\ \underset{8}{4}]$$

Permutation

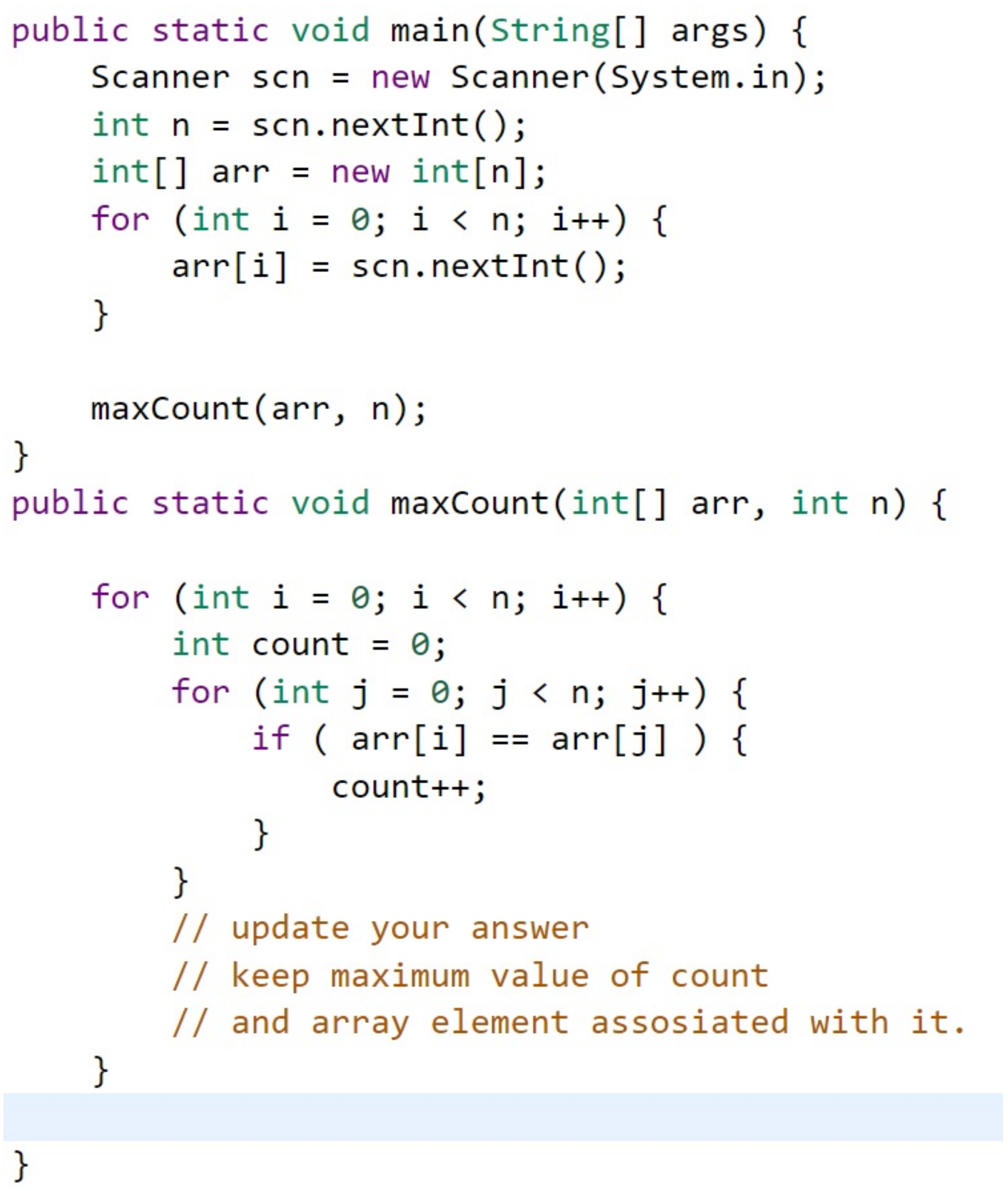
$2 \longrightarrow 4$

$1 \longrightarrow 2$

$4 \longrightarrow 3$

ans = 2

code



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

    maxCount(arr, n);
}
public static void maxCount(int[] arr, int n) {

    for (int i = 0; i < n; i++) {
        int count = 0;
        for (int j = 0; j < n; j++) {
            if ( arr[i] == arr[j] ) {
                count++;
            }
        }
        // update your answer
        // keep maximum value of count
        // and array element assosiated with it.
    }

}
```