Problem Statement **Suggest Edit**

You are given a sorted array Arr consisting of N integers and an integer X, you need to find the first and last position of occurrence of X in the array.

Note:

1. The array follows 0-based indexing, so you need to return 0-based indices.
2. If X is not present in the array, return “-1 -1”.
3. If X is only present once in the array, the first and last position of its occurrence will be the same.

Follow Up:

Try to solve the problem in O(log(n)) time complexity.

Input Format:

The first line of the input contains an integer T denoting the number of test cases.

The first line of each test case contains the integer N, denoting the size of the sorted array.

The second line of each test case contains N space-separated integers denoting the array elements.

The third and last line of each test case contains the value X, whose first and last position of occurrence you need to find.

Output Format:

The only line of output of each test case should contain two space-separated integers, where the first and second integer will be the first and the last position of occurrence of X respectively in the array.

Note:

Just implement the given function. You do not need to print anything, it has already been taken care of.

Constraints:

```
1 <= T <= 50
1 <= N <= 10^4
-10^9 <= Arr[i] <= 10^9
-10^9 <= X <= 10^9
Time Limit: 1sec
```

Sample Input 1:

```
1
5
-10 -5 -5 -5 2
-5
```

Sample Output 1:

```
1 3
```

Explanation For Sample Input 1:

```
The given array’s 0-based indexing is as follows:
-10    -5    -5    -5     2
 ↓      ↓     ↓     ↓     ↓
 0      1     2     3     4

So, the first occurrence of -5 is at index 1, and the last occurrence of -5 is at
index 3.
```

Sample Input 2:

```
2
4
1 2 3 4
-1
4
1 2 3 4
3
```

Sample Output 2:

```
-1 -1
2 2
```