https://course.acciojob.com/idle?question=5a206680-d1df-4e82-8196-dd13f35ba139

- **MEDIUM**
- **Max Score: 30 Points**

Find First and Last Position of Element in Sorted Array

Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value.

If target is not found in the array, print[-1, -1].

You must write an algorithm with O(log n) runtime complexity.

Input Format

The first line contains two integers n (number of elements in the array) and target.

The second line contains n integers (value of elements in the array).

Output Format

Print two space separated integers denoting the first and last index of target.

Example 1

Input

```
6 8
 5 7 7 8 8 10
```

Output

```
3 4
```

Explanation

8 occurs for the first time at index 3 and at index 4 for the last time.

Example 2

Input

```
6 6
 5 7 7 8 8 10
```

Output

```
-1 -1
```

Explanation

6 doesn't occur in the given array, hence we return -1 -1

Constraints

0 <= nums.length <= 10^5

-10^9 <= nums[i] <= 10^9

nums is a non-decreasing array.

-10^9 <= target <= 10^9

Topic Tags

- **Binary Search**
- **Arrays**

My code

```
// n java
/*import java.util.*;
import java.lang.*;
import java.io.*;
```

```
public class Main
{
        public static void main (String[] args) throws
java.lang.Exception
        {
                //your code here
   /*
               it is also corect


           Scanner s=new Scanner(System.in);
      int n=s.nextInt();
       int k=s.nextInt();
      int c=-1,d=-1;
      int arr[]=new int[n];
      for(int i=0;i<n;i++)
        {arr[i]=s.nextInt();
         if(arr[i]==k)
          {
           if(c==-1) c=i;
           d=i;
          }
        }
      System.out.print(c+" "+d);


       Scanner s=new Scanner(System.in);
      int n=s.nextInt();
       int k=s.nextInt();
      int c=-1,d=-1;
int arr[]=new int[n];
for(int i=0;i<n;i++)
  arr[i]=s.nextInt();

           int lp=0,rp=n,mid=0,flag=0;
     while(lp<=rp)
              {
                       mid=(lp+rp)/2;
                      if(mid==0 && arr[mid]==k)
                      {
                                 System.out.print(mid+" ");
                               flag=1;
                              break;
                      }
                      if(arr[mid]==k && arr[mid-1]< k)
                      {
                              System.out.print(mid+" ");
                              flag=1;
                              break;
                      }
                      else if(arr[mid]>=k)
                              rp=mid-1;
                      else lp=mid+1;
                 if(lp==rp) break;
              }
          if(flag==0)
      System.out.print("-1 -1");
          else
          {
        lp=0;rp=n;mid=0;
     while(lp<=rp)
          {
               mid=(lp+rp)/2;
              if(mid==n-1 && arr[mid]==k)
              {
                     System.out.print(mid);
                  break;
              }
              if(arr[mid]==k && arr[mid+1]> k)
              {
                  System.out.print(mid);
                  break;
              }
              else if(arr[mid]>k)
                  rp=mid-1;
              else lp=mid+1;
           if(lp==rp) break;
          }
        }
    }
}*/
import java.util.*;
import java.lang.*;
import java.io.*;

public class Main {
  public static void main(String[] args) throws java.lang.Exception {
    // your code here
    /*
     * it is also corect
```

```
 *
 *
 * Scanner s=new Scanner(System.in);
 * int n=s.nextInt();
 * int k=s.nextInt();
 * int c=-1,d=-1;
 * int arr[]=new int[n];
 * for(int i=0;i<n;i++)
 * {arr[i]=s.nextInt();
 * if(arr[i]==k)
 * {
 * if(c==-1) c=i;
 * d=i;
 * }
 * }
 * System.out.print(c+" "+d);
 */

Scanner s = new Scanner(System.in);
int n = s.nextInt();
int k = s.nextInt();
int c = -1, d = -1;
int arr[] = new int[n];
for (int i = 0; i < n; i++)
    arr[i] = s.nextInt();

int lp = 0, rp = n - 1, mid = 0, flag = 0;
while (lp <= rp) {
    mid = (lp + rp) / 2;
    if (arr[mid] < k) {
```

```
                lp = mid + 1;
            } else {
                if (arr[mid] == k)
                    c = mid;
                rp = mid - 1;
            }
        }
        if (c == -1) {
            System.out.print("-1 -1");
            return;
        }
        lp = 0;
        rp = n - 1;
        mid = 0;
        while (lp <= rp) {
            mid = (lp + rp) / 2;
            if (arr[mid] > k) {
                rp = mid - 1;
            } else {
                if (arr[mid] == k)
                    d = mid;
                lp = mid + 1;
            }
        }
        System.out.println(c + " " + d);
    }

}
```