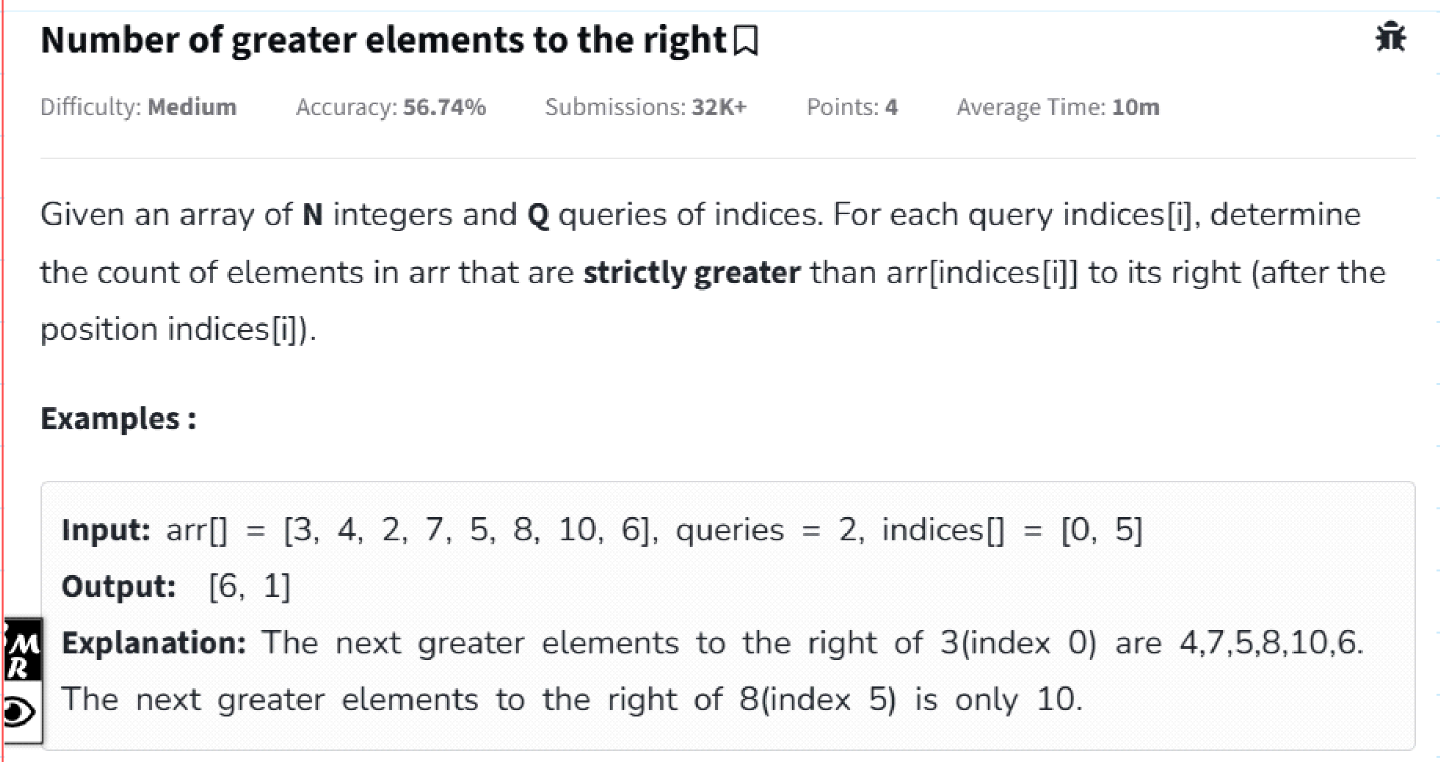
Number of NGEs to the right

18 February 2025 17:00



Number of greater elements to the right

Difficulty: **Medium** Accuracy: **56.74%** Submissions: **32K+** Points: **4** Average Time: **10m**

Given an array of **N** integers and **Q** queries of indices. For each query indices[i], determine the count of elements in arr that are **strictly greater** than arr[indices[i]] to its right (after the position indices[i]).

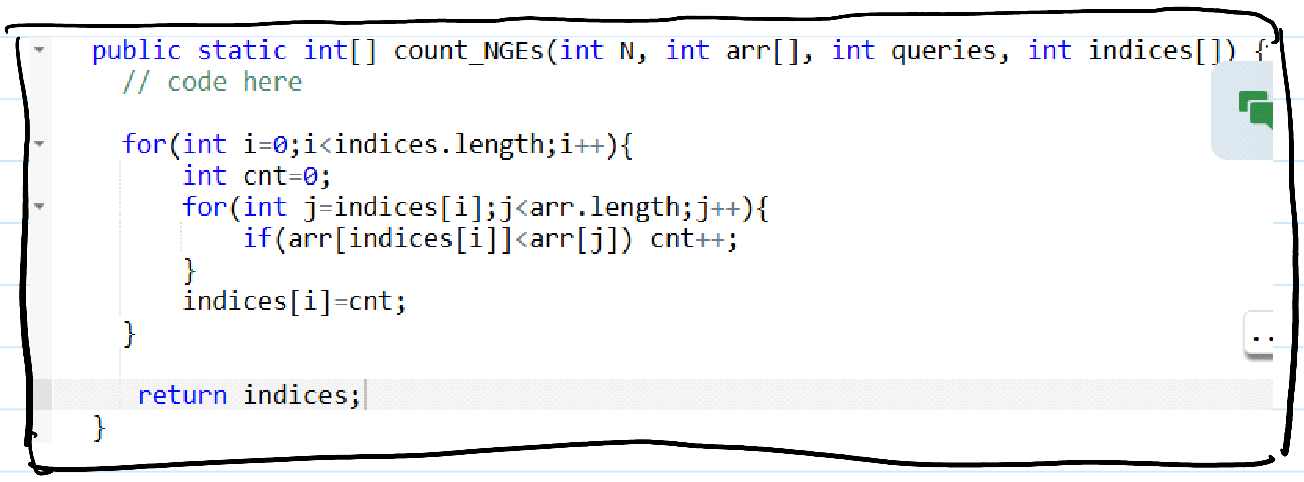
Examples :

```
Input: arr[] = [3, 4, 2, 7, 5, 8, 10, 6], queries = 2, indices[] = [0, 5]
Output:  [6, 1]
Explanation: The next greater elements to the right of 3(index 0) are 4,7,5,8,10,6.
The next greater elements to the right of 8(index 5) is only 10.
```

Method 1.
Dry Run

3 → (4, 7, 5, 8, 10, 6) — (idx = 0)

8 → (10) — (idx = 5)



```java
public static int[] count_NGEs(int N, int arr[], int queries, int indices[]) {
  // code here

  for(int i=0;i<indices.length;i++){
      int cnt=0;
      for(int j=indices[i];j<arr.length;j++){
          if(arr[indices[i]]<arr[j]) cnt++;
      }
      indices[i]=cnt;
  }

   return indices;
}
```

Method - 2
Dry Run

(3, 4, 2, 7, 5, 8, 10, 6)

→ Answer arr -

[, 0]

Node 1. 6

→ [, 0, 0]

Node2. 6 — 10

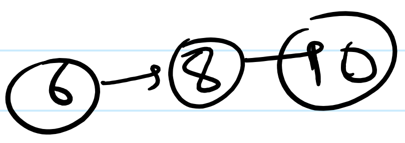
18 February 2025 23:54

Array

(3, 4, 2, 7, 5, 8, 10, 6)

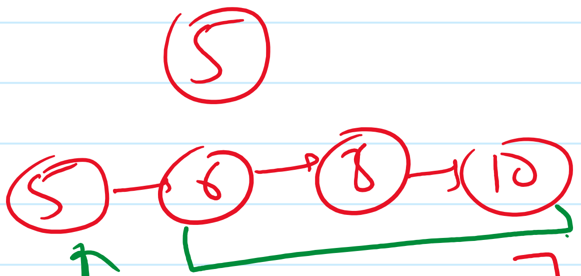
↑

Node



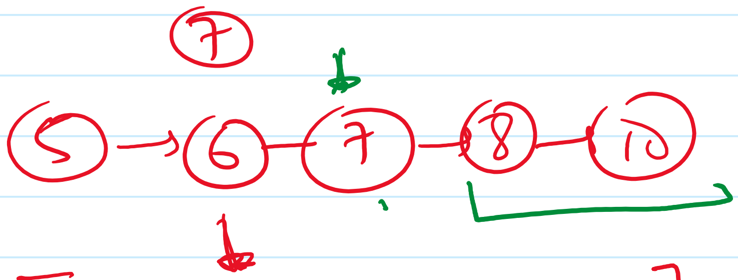
Ans [1, 0 0]

Node



Ans [,3, 1, 0, 0]

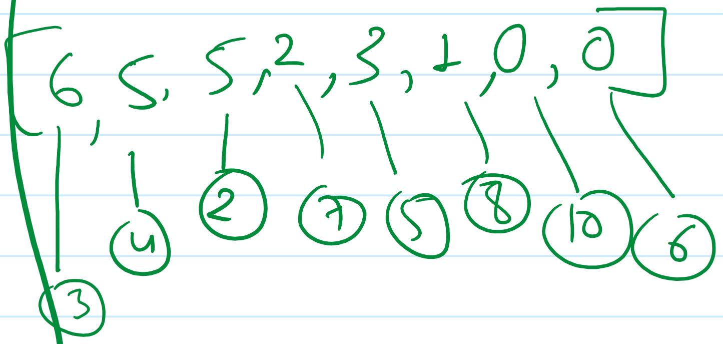
Node



[2, 3, 1, 0, 0]

→ so on.

finally we get



sodo code

create function -

1st who which return the idx of current element

```
public static int search(List<Integer> list , int key){
    int st=0, ei=list.size()-1 ;
    while(st<=ei){
        int mid=(st+ei)/2;

        if(list.get(mid)<=key){
            st=mid+1;
        }else{
            ei=mid-1;
        }
    }
    return st;
}
```

19 February 2025 00:48

② function who return the Number is great
(or) function - which count the Greater element next to it (it contain the No.(or) cnt).

```
public static int[] nge(int[] arr){
    List<Integer> list=new ArrayList<>();
    int[] ans=new int[arr.length];

    for(int i=arr.length-1;i>=0;i--){
        int idx=search(list,arr[i]);

        list.add(idx,arr[i]);

        ans[i]=list.size()-idx-1;

    }

    return ans;
}
```

③ (required idx value cnt) main function in arr.

```
public static int[] count_NGEs(int N, int arr[], int queries, int indices[]) {
  // code here

    int[] ngeArr=nge(arr);

    int[] ans=new int[indices.length];
   for(int i=0;i<indices.length;i++){
       ans[i]=ngeArr[indices[i]];
   }

  return ans;

}
```

TC

for Binary
↳ (Log N)
↳ for each element
$|N * \log N|$
↳ store the value direct to idx.

TC - $O(N * \log N)$
SC - $O(N) + O(\text{index arr})$

↓ Store cnt arr

↓ (having answer size)

Store
cnr all

(having answer
size)