1539. Kth Missing Positive Number

Hint

Easy 5.4K 347

Companies

Given an array `arr` of positive integers sorted in a **strictly increasing order**, and an integer `k`.

Return *the* `k^th` ***positive*** *integer that is* ***missing*** *from this array.*

Example 1:

```
Input: arr = [2,3,4,7,11], k = 5
Output: 9
Explanation: The missing positive integers are [1,5,6,8,9,10,12,13,...]. The 5th missing positive integer is 9.
```

Example 2:

```
Input: arr = [1,2,3,4], k = 2
Output: 6
Explanation: The missing positive integers are [5,6,7,...]. The 2nd missing positive integer is 6.
```

Constraints:

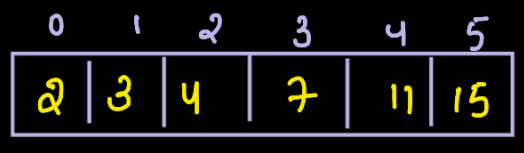
- `1 <= arr.length <= 1000`
- `1 <= arr[i] <= 1000`
- `1 <= k <= 1000`
- `arr[i] < arr[j]` for `1 <= i < j <= arr.length`

Follow up:

Could you solve this problem in less than O(n) complexity?

Accepted 311.1K | Submissions 531.2K | Acceptance Rate 58.6%

Approach 1:



1+3=10

1-4=3

K=3

How do we find no of elements missed upto this

specific index?

no. of elements missed upto index i
$= a[i] - (i+1)$

now what we can do is perform linear scan over the array until $a[i] - (i+1) > k$.
After finding the index where the above condition became true we stop scanning and return $a[i-1] + k - (a[i-1] - (i+1))$
i.e. $k + i + 1$.

```
for(i:0 to n-1)
{
    if (a[i] -(i+1) ≥k)  break
}
```

return $i == 0 \ ? \ k : k + i + 1$

↳ try with $k=1$ for above array

$O(n)$

Approach 2:

Using the same idea as above but instead of linear scan we can use binary search.

```
class Solution {
public:
    int findKthPositive(vector<int>& nums, int k) {
        int l=0,r=nums.size()-1;
        while(l<=r){
            int mid=(l+r)/2;
            if((nums[mid]-(mid+1))<k) l=mid+1;
```

→ means the no. of missed

```
            else r=mid-1;
        }

        return r<0?k:(k+r+1);
    }
};
```

elements upto mid are less than k. So we move right.

O(log n)

Try with k=1 for above array