Max chunks to make array sorted - II

768. Max Chunks To Make Sorted II

Solved

Hard | Topics | Companies | Hint

You are given an integer array `arr`.

We split `arr` into some number of **chunks** (i.e., partitions), and individually sort each chunk. After concatenating them, the result should equal the sorted array.

Return *the largest number of chunks we can make to sort the array.*

Example 1:

```
Input: arr = [5,4,3,2,1]
Output: 1
Explanation:
Splitting into two or more chunks will not return the required result.
For example, splitting into [5, 4], [3, 2, 1] will result in [4, 5, 1, 2, 3], which isn't sorted.
```

Example 2:

```
Input: arr = [2,1,3,4,4]
Output: 4
Explanation:
We can split into two chunks, such as [2, 1], [3, 4, 4].
However, splitting into [2, 1], [3], [4], [4] is the highest number of chunks possible.
```

Everything is same as type - I, problem

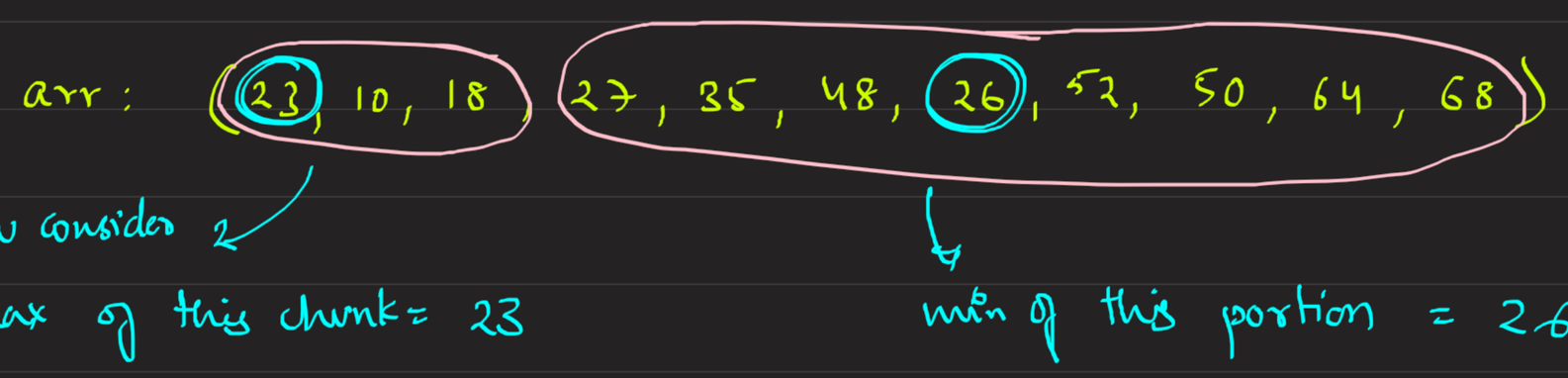
But, array index values are not permutation of indexes,

like → we can have any value, it can't bound ~~index~~ max

Ex : (23, 10, 18, 27, 35, 48, 26, 52, 50, 64, 68) → arr[]

if you take this array, there is nothing to do with indexes and array values like previous problem

Intution :



23 < 26

So, if you sort portion I, II seperately, they won't merge

Because,

in portion - I (max = 23)

in portion - II (min = 26)

if portion - II get sorted, individually, the first element will be 26 and that definitly lies after max element of portion - I

This is the main logic :

So, at which ever index, the max element (upto till that index) is less than the min element (from that next index to n-1)

⇓

We can make chunk

How do we keep track of which is max & particular index and which is min from back till that before index ??

we do prefixMax & suffixMin

Ex:

arr[] : (23, 10, 18, 27, 35, 48, 26, 52, 50, 64, 68)

prefixMax[] : (23, 23, 23, 27, 35, 48, 48, 52, 52, 64, 68)

suffixMin[] : (10, 10, 18, 26, 26, 26, 26, 50, 50, 64, 68)

23 <= 26 ✓ → So we chunk

Result : prefixMax[i] <= suffixMin[i+1]

i	prefixMax[i]	suffixMin[i-1]	Result	chunk
0	23	10	False	0
1	23	18	False	0
2	23	26	True	~~0~~ 1
3	27	26	False	1
4	35	26	False	1
5	48	26	False	1
6	48	50	True	~~1~~ 2
7	53	50	False	2
8	52	64	True	~~2~~ 3
9	64	68	False	3
10	—	—	—	—

At last, no need to check, as there won't have suffixMin[i+1] for 10th index as it is last

Return ans with +1, because we don't count last chunk