Problem 768: Max Chunks To Make Sorted II

Problem Information

Difficulty: Hard
Acceptance Rate: 54.59%
Paid Only: No
Tags: Array, Stack, Greedy, Sorting, Monotonic Stack

Problem Description

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.

Constraints:

* `1 <= arr.length <= 2000` * `0 <= arr[i] <= 108`

Code Snippets

C++:

```
class Solution {
public:
int maxChunksToSorted(vector<int>& arr) {

}
};
```

Java:

```
class Solution {
public int maxChunksToSorted(int[] arr) {

}
}
```

Python3:

```
class Solution:
def maxChunksToSorted(self, arr: List[int]) -> int:
```