

769. Max Chunks To Make Sorted

Description

You are given an integer array `arr` of length `n` that represents a permutation of the integers in the range `[0, n - 1]`.

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 = [4,3,2,1,0]`

Output: 1

Explanation:

Splitting into two or more chunks will not return the required result.

For example, splitting into `[4, 3]`, `[2, 1, 0]` will result in `[3, 4, 0, 1, 2]`, which isn't sorted.

Example 2:

Input: `arr = [1,0,2,3,4]`

Output: 4

Explanation:

We can split into two chunks, such as `[1, 0]`, `[2, 3, 4]`.

However, splitting into `[1, 0]`, `[2]`, `[3]`, `[4]` is the highest number of chunks possible.

Constraints:

- `n == arr.length`
- `1 <= n <= 10`
- `0 <= arr[i] < n`
- All the elements of `arr` are **unique**.

