Given an array arr that is a permutation of [0, 1, ..., arr.length - 1], we split the array into some number of "chunks" (partitions), and individually sort each chunk.  After concatenating them, the result equals the sorted array.

What is the most number of chunks we could have made?

**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.

**Note:**

* arr will have length in range [1, 10].
* arr[i] will be a permutation of [0, 1, ..., arr.length - 1].