Description | Editorial | Solutions | Submissions

975. Odd Even Jump

Hard | Topics | Companies

You are given an integer array `arr`. From some starting index, you can make a series of jumps. The (1st, 3rd, 5th, ...) jumps in the series are called **odd-numbered jumps**, and the (2nd, 4th, 6th, ...) jumps in the series are called **even-numbered jumps**. Note that the **jumps** are numbered, not the indices.

You may jump forward from index `i` to index `j` (with `i < j`) in the following way:

- During **odd-numbered jumps** (i.e., jumps 1, 3, 5, ...), you jump to the index `j` such that `arr[i] <= arr[j]` and `arr[j]` is the smallest possible value. If there are multiple such indices `j`, you can only jump to the **smallest** such index `j`.
- During **even-numbered jumps** (i.e., jumps 2, 4, 6, ...), you jump to the index `j` such that `arr[i] >= arr[j]` and `arr[j]` is the largest possible value. If there are multiple such indices `j`, you can only jump to the **smallest** such index `j`.
- It may be the case that for some index `i`, there are no legal jumps.

A starting index is **good** if, starting from that index, you can reach the end of the array (index `arr.length - 1`) by jumping some number of times (possibly 0 or more than once).

Return *the number of **good** starting indices*.

Example 1:

```
Input: arr = [10,13,12,14,15]
Output: 2
Explanation:
```

2.1K | 58 | 13 Online

Code

Python3 • Auto

```
1  class Solution:
2      def oddEvenJumps(self, arr: List[int]) -> int:
3
```

Ln 1, Col 1 | Saved

Run | Submit

You need to Login / Sign up to run or submit

Testcase | Test Result

Case 1 | Case 2 | Case 3 +

arr =

```
[10,13,12,14,15]
```

Source