5732. Maximum Element After Decreasing and Rearranging

My Submissions (/contest/biweekly-contest-51/problems/maximum-element-after-decreasing-and-rearranging/submissions/)

Back to Contest (/contest/biweekly-contest-51/)

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Medium

You are given an array of positive integers `arr`. Perform some operations (possibly none) on `arr` so that it satisfies these conditions:

- The value of the **first** element in `arr` must be `1`.
- The absolute difference between any 2 adjacent elements must be **less than or equal to** `1`. In other words, `abs(arr[i] - arr[i - 1]) <= 1` for each `i` where `1 <= i < arr.length` (**0-indexed**). `abs(x)` is the absolute value of `x`.

There are 2 types of operations that you can perform any number of times:

- **Decrease** the value of any element of `arr` to a **smaller positive integer**.
- **Rearrange** the elements of `arr` to be in any order.

Return *the **maximum** possible value of an element in* `arr` *after performing the operations to satisfy the conditions*.

Example 1:

```
Input: arr = [2,2,1,2,1]
Output: 2
Explanation:
We can satisfy the conditions by rearranging arr so it becomes [1,2,2,2,1].
The largest element in arr is 2.
```

Example 2:

```
Input: arr = [100,1,1000]
Output: 3
Explanation:
One possible way to satisfy the conditions is by doing the following:
1. Rearrange arr so it becomes [1,100,1000].
2. Decrease the value of the second element to 2.
3. Decrease the value of the third element to 3.
Now arr = [1,2,3], which satisfies the conditions.
The largest element in arr is 3.
```

Example 3:

```
Input: arr = [1,2,3,4,5]
Output: 5
Explanation: The array already satisfies the conditions, and the largest element is 5.
```

Constraints:

- `1 <= arr.length <= 10^5`
- `1 <= arr[i] <= 10^9`

JavaScript

```
/**
 * @param {number[]} arr
 * @return {number}
 */
var maximumElementAfterDecrementingAndRearranging = function(arr) {

};
```