

Problem 3072: Distribute Elements Into Two Arrays II

Problem Information

Difficulty: Hard

Acceptance Rate: 29.97%

Paid Only: No

Tags: Array, Binary Indexed Tree, Segment Tree, Simulation

Problem Description

You are given a **1-indexed** array of integers `nums` of length `n`.

We define a function `greaterCount` such that `greaterCount(arr, val)` returns the number of elements in `arr` that are **strictly greater** than `val`.

You need to distribute all the elements of `nums` between two arrays `arr1` and `arr2` using `n` operations. In the first operation, append `nums[1]` to `arr1`. In the second operation, append `nums[2]` to `arr2`. Afterwards, in the `ith` operation:

* If `greaterCount(arr1, nums[i]) > greaterCount(arr2, nums[i])`, append `nums[i]` to `arr1`. * If `greaterCount(arr1, nums[i]) < greaterCount(arr2, nums[i])`, append `nums[i]` to `arr2`. * If `greaterCount(arr1, nums[i]) == greaterCount(arr2, nums[i])`, append `nums[i]` to the array with a **lesser** number of elements. * If there is still a tie, append `nums[i]` to `arr1`.

The array `result` is formed by concatenating the arrays `arr1` and `arr2`. For example, if `arr1 == [1,2,3]` and `arr2 == [4,5,6]`, then `result = [1,2,3,4,5,6]`.

Return _the integer array_ `result`.

Example 1:

Input: `nums = [2,1,3,3]` **Output:** ` [2,3,1,3]` **Explanation:** After the first 2 operations, `arr1 = [2]` and `arr2 = [1]`. In the 3rd operation, the number of elements greater than 3 is zero in both arrays. Also, the lengths are equal, hence, append `nums[3]` to `arr1`. In the 4th operation, the number of elements greater than 3 is zero in both arrays. As the length of `arr2` is lesser,

hence, append nums[4] to arr2. After 4 operations, arr1 = [2,3] and arr2 = [1,3]. Hence, the array result formed by concatenation is [2,3,1,3].

****Example 2:****

****Input:**** nums = [5,14,3,1,2] ****Output:**** [5,3,1,2,14] ****Explanation:**** After the first 2 operations, arr1 = [5] and arr2 = [14]. In the 3rd operation, the number of elements greater than 3 is one in both arrays. Also, the lengths are equal, hence, append nums[3] to arr1. In the 4th operation, the number of elements greater than 1 is greater in arr1 than arr2 ($2 > 1$). Hence, append nums[4] to arr1. In the 5th operation, the number of elements greater than 2 is greater in arr1 than arr2 ($2 > 1$). Hence, append nums[5] to arr1. After 5 operations, arr1 = [5,3,1,2] and arr2 = [14]. Hence, the array result formed by concatenation is [5,3,1,2,14].

****Example 3:****

****Input:**** nums = [3,3,3,3] ****Output:**** [3,3,3,3] ****Explanation:**** At the end of 4 operations, arr1 = [3,3] and arr2 = [3,3]. Hence, the array result formed by concatenation is [3,3,3,3].

****Constraints:****

* `3 <= n <= 105` * `1 <= nums[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
vector<int> resultArray(vector<int>& nums) {
}
};
```

Java:

```
class Solution {
public int[] resultArray(int[] nums) {
}
}
```

Python3:

```
class Solution:  
    def resultArray(self, nums: List[int]) -> List[int]:
```