

88. Merge Sorted Array

[Easy](#) [🏷️ Topics](#) [🏢 Companies](#) [💡 Hint](#)

You are given two integer arrays `nums1` and `nums2`, sorted in **non-decreasing order**, and two integers `m` and `n`, representing the number of elements in `nums1` and `nums2` respectively.

Merge `nums1` and `nums2` into a single array sorted in **non-decreasing order**.

The final sorted array should not be returned by the function, but instead be *stored inside the array* `nums1`. To accommodate this, `nums1` has a length of `m + n`, where the first `m` elements are valid and the last `n` elements are reserved.

Example 1:

Input: `nums1 = [1,2,3,0,0,0]`, `m = 3`, `nums2 = [2,5,6]`, `n = 3`
Output: `[1,2,2,3,5,6]`
Explanation: The arrays we are merging are `[1,2,3]` and `[2,5,6]`.
The result of the merge is `[1,2,2,3,5,6]` with the underlined elements coming from `nums1`.

Example 2:

Input: `nums1 = [1]`, `m = 1`, `nums2 = []`, `n = 0`
Output: `[1]`
Explanation: The arrays we are merging are `[1]` and `[]`.
The result of the merge is `[1]`.

Example 3:

Input: `nums1 = [0]`, `m = 0`, `nums2 = [1]`, `n = 1`
Output: `[1]`
Explanation: The arrays we are merging are `[]` and `[1]`.
The result of the merge is `[1]`.
Note that because `m = 0`, there are no elements in `nums1`. The `0` is only there to ensure the merge result is of length `m + n`.

Constraints:

- `nums1.length == m + n`
- `nums2.length == n`
- `0 <= m, n <= 200`
- `1 <= m + n <= 200`
- `-109 <= nums1[i], nums2[j] <= 109`

Follow up: Can you come up with an algorithm that runs in $O(m + n)$ time?

Seen this question in a real interview before? 1/4

☒ Yes ☐ No

Accepted **2.8M** Submissions **5.8M** Acceptance Rate **49.0%**

[🏷️ Topics](#)

[🏢 Companies](#)

[💡 Hint 1](#)

[💡 Hint 2](#)

[🔖 Similar Questions](#)