Yunyeong Kim

Question

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 denote the elements that should be merged, and the last `n` elements are set to `0` and should be ignored. `nums2` has a length of `n`.

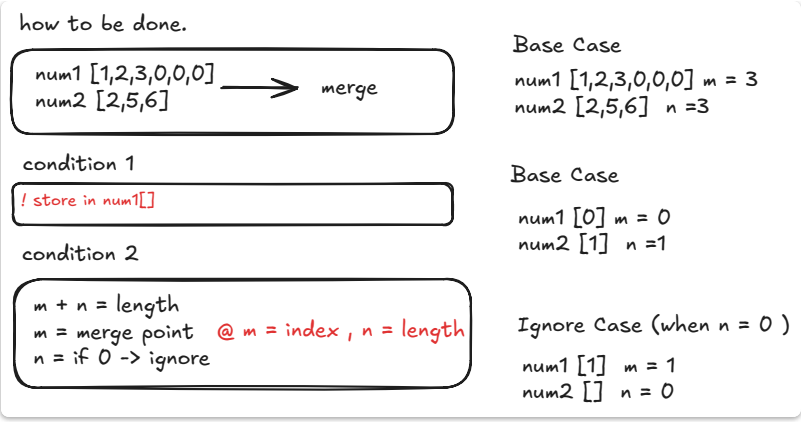
Examples

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.

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].

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 can fit in nums1.

Definition



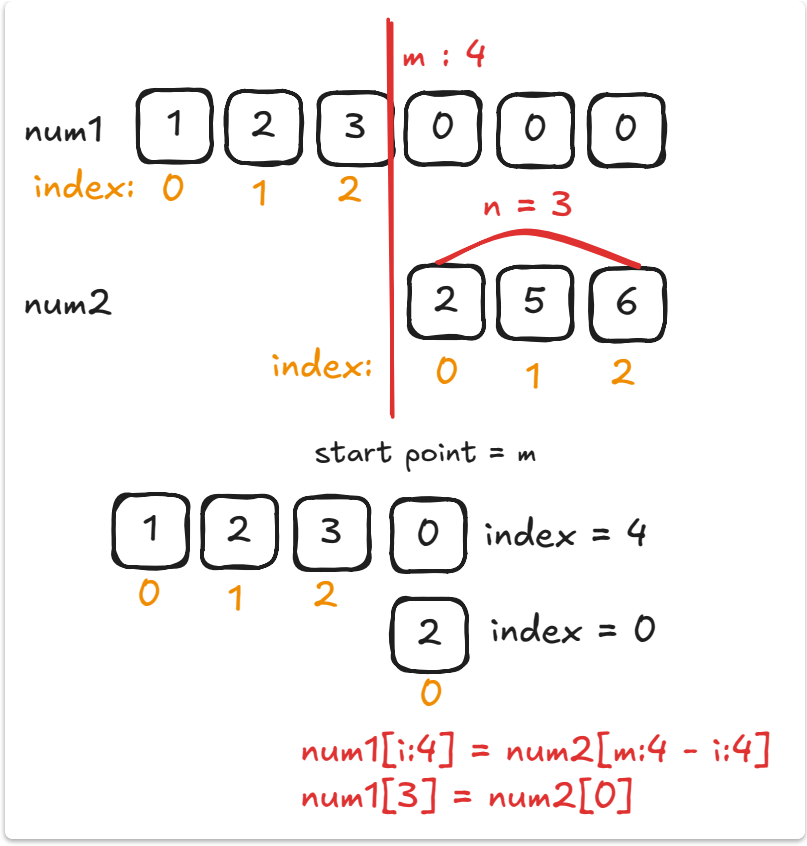
- Merge 2 array `num1`, `num2`
- Merge Index is given with `m`
- Merge to `num1`
- `m` + `n` = length
- If `n = 0` ignore.
- Non-Decreasing order
- ! `m` = index `n` = length

First Code

```
class Solution:

    def merge(self, nums1: list[int], m: int, nums2: list[int], n: int) ->
None:
        if n == 0:
            return None
        for i in range(m, m+n):
            nums1[i] = nums2[m-i]
            nums1.sort()
# 100% beats.
```

First Code Explain



start point = m

num1[i:4] = num2[m:4 - i:4]

num1[3] = num2[0]