088 Merge Sorted Array

2018年3月30日 14:02

Question:

Given two sorted integer arrays *nums1* and *nums2*, merge *nums2* into *nums1* as one sorted array.

Note:

You may assume that *nums1* has enough space (size that is greater or equal to *m* + *n*) to hold additional elements from *nums2*. The number of elements initialized in *nums1* and *nums2* are *m* and *n* respectively.

来自 <https://leetcode.com/problems/merge-sorted-array/description/>

给定两个有序整数数组 *nums1* 和 *nums2*，将 *nums2* 合并到 *nums1*中，使得 *num1 成*为一个有序数组。

注意:

你可以假设 *nums1*有足够的空间（空间大小大于或等于*m + n*）来保存 *nums2* 中的元素。在 *nums1* 和 *nums2* 中初始化的元素的数量分别是 *m* 和 *n*。

Solution for Python3:

```python
class Solution:
    def merge(self, nums1, m, nums2, n):
        """
        :type nums1: List[int]
        :type m: int
        :type nums2: List[int]
        :type n: int
        :rtype: void Do not return anything, modify nums1 in-place instead.
        """
        t = m + n
        while m and n:
            if nums1[m - 1] > nums2[n - 1]:
                nums1[t - 1] = nums1[m - 1]
                m -= 1
            else:
                nums1[t - 1] = nums2[n - 1]
                n -= 1
            t -= 1
        nums1[:t] = nums2[:t] if n else nums1[:t]


class Solution2:
    def merge(self, nums1, m, nums2, n):
        """
        :type nums1: List[int]
        :type m: int
        :type nums2: List[int]
        :type n: int
        :rtype: void Do not return anything, modify nums1 in-place instead.
        """
        while n > 0:
            if m > 0 and nums1[m-1] > nums2[n-1]:
                nums1[m+n-1] = nums1[m-1]
                m -= 1
            else:
                nums1[m+n-1] = nums2[n-1]
                n -= 1
```

Solution for C++:

```cpp
class Solution {
public:
    void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {
        int i = m - 1, j = n - 1, tar = m + n - 1;
        while (j >= 0) {
            nums1[tar--] = i >= 0 && nums1[i] > nums2[j] ? nums1[i--] : nums2[j--];
        }
    }
};
```