Author: Xiaoqiang Yan
Date: 08/06/2018

LeetCode 88. Merge Sorted Array

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

Note:

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

```
Input:
nums1 = [1,2,3,0,0,0], m = 3
nums2 = [2,5,6],       n = 3

Output: [1,2,2,3,5,6]
```

Method

The easiest solution is to traverse the elements in these two arrays using two pointers, compare each element and save the smaller one in a new arraylist. But we are required to merge nums2 into nums1.

We can traverse two arrays from head. When nums2[j] < nums1[i], we need to insert nums[j] before nums[i]. Thus the length of nums1 and the positions of left numbers will be changed. So we need to find a method not to change the original positions of nums1.

We know nums1 has enough space to put the merged array, so we can put the numbers of nums2 in the unused space. If we store new numbers from tail to head, because biggest number is in the last, we also need to traverse the arrays from the end, and store the bigger one from the (m+n-1)th position.

Code

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

```
        while(j>=0)
            nums1[last--] = nums2[j--];
    }
};
```

Note

1. Don’t forget the left numbers in two arrays. In this question, because we merge nums2 into nums1, so we only need to check left numbers in nums2.
2. Remember the idea of “try tail if head is not good”.
3. we also can’t exchange the numbers in two arrays. By doing that, array nums2 will not be a sorted array.([1,5,6,0,0,0] [2,3,4], 3, 3)

Summary

Double pointer model