# 题目

给你两个有序整数数组nums1 和 nums2，请你将 nums2 合并到nums1中，使 nums1 成为一个有序数组。

**说明:**

初始化nums1 和 nums2 的元素数量分别为m 和 n 。

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。

**示例:**

输入:

nums1 = [1,2,3,0,0,0], m = 3

nums2 = [2,5,6], n = 3

输出: [1,2,2,3,5,6]

# 分析

## 方法一：双指针

思路：

代码：

class Solution {

public:

void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {

int i = m - 1, j = n - 1, k = m + n - 1; // 初始化指针，分别指向 nums1 的末尾（有效元素的最后一个位置）、nums2 的末尾、合并后数组的末尾

while (i >= 0 && j >= 0) { // 从后往前比较 nums1 和 nums2 的元素

if (nums1[i] > nums2[j]) { // 如果 nums1 的当前元素大于 nums2 的当前元素

nums1[k--] = nums1[i--]; // 将 nums1 的当前元素放入合并后数组的末尾，并将对应指针向前移动一位

} else {

nums1[k--] = nums2[j--]; // 如果 nums2 的当前元素大于等于 nums1 的当前元素，将 nums2 的当前元素放入合并后数组的末尾，并将对应指针向前移动一位

}

}

while (j >= 0) { // 如果 nums2 还有剩余元素

nums1[k--] = nums2[j--]; // 将 nums2 剩余的元素依次填入合并后数组的前面

}

}

};

## 方法二：模拟

class Solution {

public:

void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {

int p1 = 0,p2 = 0;

int cur;

int tmpNum[m+n] ; // 需要用到额外空间，不太好

while(p1 < m || p2 < n)

{

if(p1 == m) // p1遍历到头了，p2还有

{

cur = nums2[p2++];

}

else if(p2 == n)

{

cur = nums1[p1++];

}

else if(nums1[p1] < nums2[p2])

{

cur = nums1[p1++];

}

else

{

cur = nums2[p2++];

}

tmpNum[p1+p2-1] = cur;

}

for(int i=0;i<m+n;i++)

{

nums1[i] = tmpNum[i];

}

}

};