321. Create Maximum Number

题目描述：https://leetcode.com/problems/create-maximum-number/

> 给定两个数组s1, s2，要求不改变数字的相对顺序组成最大的k位数。
> 例如：

```
Example 1:
nums1 = [3, 4, 6, 5]
nums2 = [9, 1, 2, 5, 8, 3]
k = 5
return [9, 8, 6, 5, 3]

Example 2:
nums1 = [6, 7]
nums2 = [6, 0, 4]
k = 5
return [6, 7, 6, 0, 4]

Example 3:
nums1 = [3, 9]
nums2 = [8, 9]
k = 3
return [9, 8, 9]
```

解题思路：

遍历第一个数组，用前i位构成满足条件的数，判断i的位置。

```
maxArray(vector<int> nums, int i) //找到nums可以组成的最大的i位数
greater(vector<int> nums1, vector<int> nums2, int i, int j) //s1的i位以后和s2的j位以后的形成的数字比较，哪个更大
merge(vector<int> nums1, vector<int> nums2, int k) // 两个数组合并使得形成的k位数的最大
```

代码：

```
class Solution {
public:
    bool greater(vector<int> nums1, vector<int> nums2, int i, int j) {
        while(i < nums1.size() && j < nums2.size() && nums1[i] == nums2[j]) {
            i++; j++;
        }
        return j == nums2.size() || (i < nums1.size() && nums1[i] > nums2[j]);
    }
    vector<int> maxArray(vector<int> &nums, int k) {
        int n = nums.size();
        vector<int> ans(k);
        for(int i = 0, j = 0; i < n; i++) {
            while(n - i + j > k && j > 0 && ans[j-1] < nums[i]) j--;
            if(j < k) ans[j++] = nums[i];
        }
        return ans;
    }
    vector<int> merge(vector<int> nums1, vector<int> nums2, int k) {
        vector<int> ans(k);
        for(int i = 0, j = 0, r = 0; r < k; r++) {
            ans[r] = greater(nums1, nums2, i, j) ? nums1[i++] : nums2[j++];
        }
        return ans;
    }
    vector<int> maxNumber(vector<int>& nums1, vector<int>& nums2, int k) {
        int n = nums1.size(), m = nums2.size();
        vector<int> ans(k);
        for(int i = max(0, k - m); i <= k && i <= n; i++) {
            vector<int> candidate;
            candidate = merge(maxArray(nums1, i), maxArray(nums2, k-i), k);
            if(greater(candidate, ans, 0, 0)) {
                ans = candidate;
            }
        }
        return ans;
    }
};
```