350 两个数组的交集II

label：hash、双指针
给定两个数组，编写一个函数来计算它们的交集。
输入：nums1 = [1,2,2,1], nums2 = [2,2]
输出：[2,2]

输入：nums1 = [4,9,5], nums2 = [9,4,9,8,4]
输出：[4,9]

输出结果中每个元素出现的次数，应与元素在两个数组中出现次数的最小值一致。
我们可以不考虑输出结果的顺序。

- hash

```
class Solution {
    public int[] intersect(int[] nums1, int[] nums2) {
        if (nums1.length > nums2.length) {
            return intersect(nums2, nums1); // 交换一下，保证 nums1更短
        }

        Map<Integer, Integer> map = new HashMap<Integer, Integer>();
        for (int num : nums1) {  // 短的做为记录
            int count = map.getOrDefault(num, 0) + 1;
            map.put(num, count);
        }

        int[] intersection = new int[nums1.length]; // 相交最多不会超过
nums1.length
        int index = 0; // 保留index方便截取

        for (int num : nums2) {
            int count = map.getOrDefault(num, 0);

            if (count > 0) {
                intersection[index++] = num;
                count--;
                if (count > 0) {
                    map.put(num, count);
                } else {
                    map.remove(num);
                }
            }
        }

        return Arrays.copyOfRange(intersection, 0, index);
    }
}
```

- 排序+双指针

```java
class Solution {
    public int[] intersect(int[] nums1, int[] nums2) {
        Arrays.sort(nums1);
        Arrays.sort(nums2);

        List<Integer> ans = new ArrayList<>();

        int i = 0, j = 0;
        while (i != nums1.length && j != nums2.length) {
            if (nums1[i] == nums2[j]) {
                ans.add(nums1[i]);  // 不用去重，两个指针都往后移动就行
                i++;
                j++;
            } else if (nums1[i] < nums2[j]) {
                i++;
            } else {
                j++;
            }
        }

        return  ans.stream().mapToInt(c -> c).toArray();
    }
}
```