# 题目

给你两个只包含1到9之间数字的数组nums1和nums2，每个数组中的元素互不相同，请你返回最小的数字，两个数组都至少包含这个数字的某个数位。

示例 1：

输入：nums1 = [4,1,3], nums2 = [5,7]

输出：15

解释：数字 15 的数位 1 在 nums1 中出现，数位 5 在 nums2 中出现。15 是我们能得到的最小数字。

示例 2：

输入：nums1 = [3,5,2,6], nums2 = [3,1,7]

输出：3

解释：数字 3 的数位 3 在两个数组中都出现了。

提示：

1 <= nums1.length, nums2.length <= 9

1 <= nums1[i], nums2[i] <= 9

每个数组中，元素 互不相同 。

# 分析

## 方法一：哈希表

思路：

如果 nums1与 nums2有交集，那么答案就是交集的最小值。

如果没有交集，设 nums1的最小值为x，nums2的最小值为y，答案就是

min(10x+y,10y+x)

代码：

class Solution {

public:

int minNumber(vector<int>& nums1, vector<int>& nums2) {

unordered\_set<int> set1(nums1.begin(), nums1.end());

unordered\_set<int> set2(nums2.begin(), nums2.end());

// 检查是否有交集

vector<int> intersection;

for (int num : set1) {

if (set2.count(num)) {

intersection.push\_back(num);

}

}

if (!intersection.empty()) {

// 如果有交集，返回交集中的最小数字

return \*min\_element(intersection.begin(), intersection.end());

}

// 没有交集，分别找到两个数组中的最小数字

int min1 = \*min\_element(nums1.begin(), nums1.end());

int min2 = \*min\_element(nums2.begin(), nums2.end());

// 返回拼接后的两个数字中的较小值

return min(min1 \* 10 + min2, min2 \* 10 + min1);

}

};

## 方法二：位运算