# 题目

给你一个下标从 0 开始的数组 nums ，它包含 n 个 互不相同 的正整数。请你对这个数组执行 m 个操作，在第 i 个操作中，你需要将数字 operations[i][0] 替换成 operations[i][1] 。

题目保证在第 i 个操作中：

operations[i][0] 在 nums 中存在。

operations[i][1] 在 nums 中不存在。

请你返回执行完所有操作后的数组。

示例 1：

输入：nums = [1,2,4,6], operations = [[1,3],[4,7],[6,1]]

输出：[3,2,7,1]

解释：我们对 nums 执行以下操作：

- 将数字 1 替换为 3 。nums 变为 [3,2,4,6] 。

- 将数字 4 替换为 7 。nums 变为 [3,2,7,6] 。

- 将数字 6 替换为 1 。nums 变为 [3,2,7,1] 。

返回最终数组 [3,2,7,1] 。

示例 2：

输入：nums = [1,2], operations = [[1,3],[2,1],[3,2]]

输出：[2,1]

解释：我们对 nums 执行以下操作：

- 将数字 1 替换为 3 。nums 变为 [3,2] 。

- 将数字 2 替换为 1 。nums 变为 [3,1] 。

- 将数字 3 替换为 2 。nums 变为 [2,1] 。

返回最终数组 [2,1] 。

提示：

n == nums.length

m == operations.length

1 <= n, m <= 10^5

nums 中所有数字 互不相同 。

operations[i].length == 2

1 <= nums[i], operations[i][0], operations[i][1] <= 10^6

在执行第 i 个操作时，operations[i][0] 在 nums 中存在。

在执行第 i 个操作时，operations[i][1] 在 nums 中不存在。

# 分析

## 方法一：模拟

思路：

1、使用哈希表 index\_map 记录每个数字在数组 nums 中的下标。

2、遍历操作数组 operations，对于每个操作 [x, y]，将 nums[index\_map[x]] 修改为 y，并更新 index\_map。

3、返回修改后的数组 nums。

代码：

class Solution {

public:

vector<int> arrayChange(vector<int>& nums, vector<vector<int>>& operations) {

unordered\_map<int, int> index\_map; // 记录每个数字在nums中的下标

for (int i = 0; i < nums.size(); ++i) {

index\_map[nums[i]] = i;

}

for (auto& op : operations) {

int x = op[0], y = op[1];

nums[index\_map[x]] = y; // 修改nums中x的位置为y

index\_map[y] = index\_map[x]; // 更新y在nums中的下标为x原来的下标

index\_map.erase(x); // 删除x在index\_map中的记录

}

return nums;

}

};

下述方法会超时：

class Solution {

public:

vector<int> arrayChange(vector<int>& nums, vector<vector<int>>& operations) {

for (auto& op : operations) {

int x = op[0], y = op[1];

int idx = 0;

while (nums[idx] != x) {

idx++;

}

nums[idx] = y;

}

return nums;

}

};