# 题目

给你一个下标从0开始的整数数组nums，数组长度为偶数，由数目相等的正整数和负整数组成。

你需要返回满足下述条件的数组nums：

任意连续的两个整数符号相反

对于符号相同的所有整数，保留它们在nums中的顺序。

重排后数组以正整数开头。

重排元素满足上述条件后，返回修改后的数组。

示例 1：

输入：nums = [3,1,-2,-5,2,-4]

输出：[3,-2,1,-5,2,-4]

解释：

nums中的正整数是[3,1,2]，负整数是[-2,-5,-4]。

重排的唯一可行方案是[3,-2,1,-5,2,-4]，能满足所有条件。

像[1,-2,2,-5,3,-4]、[3,1,2,-2,-5,-4]、[-2,3,-5,1,-4,2]这样的其他方案是不正确的，因为不满足一个或者多个条件。

示例 2：

输入：nums = [-1,1]

输出：[1,-1]

解释：

1是nums中唯一一个正整数，-1是nums中唯一一个负整数。

所以nums重排为[1,-1]。

提示：

2 <= nums.length <= 2 \* 105

nums.length是偶数

1 <= |nums[i]| <= 105

nums由相等数量的正整数和负整数组成

不需要原地进行修改。

# 分析

## 方法一：辅助数组

思路：

我们可以按照以下步骤进行操作：

1、创建一个新的结果数组 result，其长度与输入数组 nums 相同。

2、将 nums 中的正整数和负整数分别存储到两个辅助数组 positives 和 negatives 中。

3、遍历 positives 和 negatives，按顺序将正整数和负整数交替4、放入 result 数组中，确保连续的两个整数符号相反。

5、返回修改后的 result 数组。

代码：

class Solution {

public:

std::vector<int> rearrangeArray(std::vector<int>& nums) {

std::vector<int> result(nums.size());

std::vector<int> positives;

std::vector<int> negatives;

// 将正整数和负整数分别存储到两个辅助数组中

for (int num : nums) {

if (num > 0) {

positives.push\_back(num);

} else {

negatives.push\_back(num);

}

}

int posIndex = 0;

int negIndex = 0;

// 交替放入正整数和负整数到结果数组中

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

if (i % 2 == 0) {

result[i] = positives[posIndex++];

} else {

result[i] = negatives[negIndex++];

}

}

return result;

}

};

或：

class Solution {

public:

vector<int> rearrangeArray(vector<int>& nums) {

vector<int> positive, negative;

for (int num : nums) {

if (num > 0) {

positive.push\_back(num);

} else {

negative.push\_back(num);

}

}

vector<int> result;

int i = 0, j = 0;

while (i < positive.size() && j < negative.size()) {

result.push\_back(positive[i++]);

result.push\_back(negative[j++]);

}

while (i < positive.size()) {

result.push\_back(positive[i++]);

}

while (j < negative.size()) {

result.push\_back(negative[j++]);

}

return result;

}

};

## 方法二：双指针

思路：

我们可以使用两个指针对数组left进行遍历，其中指针pos负责遍历所有的正数，指针neg负责遍历所有的负数。

记数组nums的长度为n，那么其中分别包含n/2个正数和负数。因此我们只需要重复如下操作n/2次：

指针pos不断向后移动，直到遇到一个正数为止，并将该正数放入答案数组；

指针neg不断向后移动，直到遇到一个负数为止，并将该负数放入答案数组。

pos和neg初识时均指向nums的首个元素。

代码：

class Solution {

public:

vector<int> rearrangeArray(vector<int>& nums) {

int n = nums.size();

int pos = 0, neg = 0;

vector<int> ans;

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

while (nums[pos] < 0) {

++pos;

}

ans.push\_back(nums[pos]);

++pos;

while (nums[neg] > 0) {

++neg;

}

ans.push\_back(nums[neg]);

++neg;

}

return ans;

}

};

复杂度分析

时间复杂度：O(n)，其中n是数组nums的长度。

空间复杂度：O(1)。这里不计入返回值需要使用的空间。