# 题目

给出一个整数数组 A 和一个查询数组 queries。

对于第 i 次查询，有 val = queries[i][0], index = queries[i][1]，我们会把 val 加到 A[index] 上。然后，第 i 次查询的答案是 A 中偶数值的和。

（此处给定的 index = queries[i][1] 是从 0 开始的索引，每次查询都会永久修改数组 A。）

返回所有查询的答案。你的答案应当以数组 answer 给出，answer[i] 为第 i 次查询的答案。

示例：

输入：A = [1,2,3,4], queries = [[1,0],[-3,1],[-4,0],[2,3]]

输出：[8,6,2,4]

解释：

开始时，数组为 [1,2,3,4]。

将 1 加到 A[0] 上之后，数组为 [2,2,3,4]，偶数值之和为 2 + 2 + 4 = 8。

将 -3 加到 A[1] 上之后，数组为 [2,-1,3,4]，偶数值之和为 2 + 4 = 6。

将 -4 加到 A[0] 上之后，数组为 [-2,-1,3,4]，偶数值之和为 -2 + 4 = 2。

将 2 加到 A[3] 上之后，数组为 [-2,-1,3,6]，偶数值之和为 -2 + 6 = 4。

提示：

1 <= A.length <= 10000

-10000 <= A[i] <= 10000

1 <= queries.length <= 10000

-10000 <= queries[i][0] <= 10000

0 <= queries[i][1] < A.length

# 分析

## 方法一：模拟

思路：

1、遍历数组 nums，计算初始偶数值的和 sum，即将数组中所有偶数值加起来。

2、遍历查询数组 queries，对于每次查询，获取查询的值 val 和索引 index。

3、如果原来的值 nums[index] 是偶数，则从 sum 中减去它。

4、将 val 加到 nums[index] 上。

5、如果新值 nums[index] 是偶数，则将它加到 sum 中。

6、将每次查询后的 sum 加入结果数组中。

7、返回结果数组。

代码：

class Solution {

public:

vector<int> sumEvenAfterQueries(vector<int>& nums, vector<vector<int>>& queries) {

vector<int> result;

int sum = 0; // 初始偶数值的和

for (int num : nums) {

if (num % 2 == 0) {

sum += num;

}

}

for (auto& query : queries) {

int val = query[0];

int index = query[1];

int num = nums[index];

if (num % 2 == 0) {

sum -= num;

}

nums[index] += val;

if (nums[index] % 2 == 0) {

sum += nums[index];

}

result.push\_back(sum);

}

return result;

}

};