# 题目

给你一个整数数组 nums ，返回出现最频繁的偶数元素。

如果存在多个满足条件的元素，只需要返回 最小 的一个。如果不存在这样的元素，返回 -1 。

示例 1：

输入：nums = [0,1,2,2,4,4,1]

输出：2

解释：

数组中的偶数元素为 0、2 和 4 ，在这些元素中，2 和 4 出现次数最多。

返回最小的那个，即返回 2 。

示例 2：

输入：nums = [4,4,4,9,2,4]

输出：4

解释：4 是出现最频繁的偶数元素。

示例 3：

输入：nums = [29,47,21,41,13,37,25,7]

输出：-1

解释：不存在偶数元素。

提示：

1 <= nums.length <= 2000

0 <= nums[i] <= 10^5

# 分析

## 方法一：哈希表

思路：

遍历数组nums，并且使用哈希表count记录偶数元素的出现次数。使用ans和ct分别记录当前出现次数最多的元素值以及对应的出现次数。遍历哈希表中的元素，如果元素的出现次数大于ct或者出现次数等于ct且元素值小于ans，那么用ans记录当前遍历的元素值，并且用ct 记录当前遍历的元素的出现次数。

代码：

class Solution {

public:

int mostFrequentEven(vector<int>& nums) {

unordered\_map<int, int> count;

for (auto x : nums) {

if (x % 2 == 0) {

count[x]++;

}

}

int ans = -1, ct = 0;

for (auto &[k,v] : count) {

if (ans == -1 || v > ct || v == ct && ans > k) {

ans = k;

ct = v;

}

}

return ans;

}

};

复杂度分析

时间复杂度：O(n)，其中n是数组nums的长度。遍历数组与哈希表都需要O(n)。

空间复杂度：O(n)。保存哈希表需要O(n)。