# 题目

给定一个大小为 n 的数组，找到其中的多数元素。多数元素是指在数组中出现次数大于 ⌊ n/2 ⌋ 的元素。

你可以假设数组是非空的，并且给定的数组总是存在多数元素。

**示例 1:**

**输入:** [3,2,3]

**输出:** 3

**示例 2:**

**输入:** [2,2,1,1,1,2,2]

**输出:** 2

# 分析：

## 方法一：排序+中位数

可以将数组先排序，然后出现次数超过一半的必定在数组中间位置。

**代码：**

class Solution {

public:

int majorityElement(vector<int>& nums) {

std::sort(nums.begin(),nums.end());

return nums.at(nums.size()/2);

}

};

## 方法二：哈希

思路：

**代码：**

class Solution {

public:

int majorityElement(vector<int>& nums) {

unordered\_map <int,int> mp;

for(int n:nums)

if(++ mp[n] > nums.size()/2) return n;

return -1;

}

};

## 方法三：位运算

思路：

代码：

class Solution {

public:

int majorityElement(vector<int>& nums) {

int res=0;

for(int i=0;i<32;i++)

{

int ones=0;

for(int n:nums)

ones += (n >> i) & 1;

//位运算法统计每个位置上1出现的次数，每次出现则ones+1

//出现则&后为1，可以作为计数值，否则为0（巧妙！）

res += (ones > nums.size()/2) << i;

//如果1出现次数大于2分之1数组长，1即为这个位置的目标数字

}

return res;

}

};