Majority Element

Medium Accuracy: **27.82%** Submissions: **440K+** Points: **4**

Given an array **A** of **N** elements. Find the majority element in the array. A majority element in an array A of size N is an **element that appears more than N/2 times in the array**.

Example 1:

Input:
N = 3
A[] = {1,2,3}
Output:
-1
Explanation:
Since, each element in
{1,2,3} appears only once so there
is no majority element.

Example 2:

Input:
N = 5
A[] = {3,1,3,3,2}
Output:
3
Explanation:
Since, 3 is present more
than N/2 times, so it is
the majority element.

169. Majority Element

Easy 14.2K 437

Companies

Given an array `nums` of size `n`, return *the majority element*.

The majority element is the element that appears more than `⌊n / 2⌋` times. You may assume that the majority element always exists in the array.

Example 1:

```
Input: nums = [3,2,3]
Output: 3
```

Example 2:

```
Input: nums = [2,2,1,1,1,2,2]
Output: 2
```

Constraints:

- `n == nums.length`
- `1 <= n <= 5 * 10^4`
- `-10^9 <= nums[i] <= 10^9`

Follow-up: Could you solve the problem in linear time and in `O(1)` space?

Accepted 1.7M | Submissions 2.6M | Acceptance Rate 63.9%

Element which occurs more than half of the length of the array

Brute force approach :-

→ Run nested loop First loop for selection of element and 2nd loop to count no. of times that element occurs.
→ After complete iteration of 2nd loop check if count is more $n/2$ if True return Ans

```
for(0 → n)
    cnt=0
    for(0 → n)
        if(a[j]==a[i])
            cnt++;
    }
if(cnt > n/2) return True, Break
```

T.C → $O(n^2)$, S.C → $O(1)$

→ Better Approach (Hashing)

Iterate over array the Array and insert element in Hashmap, then Run loop on hashmap And Search the element which is more than $n/2$

[2, 2, 1, 1, 1, 2, 2]

map[a[i]]++

2 : ~~x~~ 4	
1 : ~~x~~ 3	

T.C → $O(n) + O(\log N)$, S.C → $O(n)$

Optimal approach (Moore Voting algorithm)

⇒ If the array contains majority element, its occurance must be greater n/2

⇒ Count the minority element and majority element are equal up to certain point

[2, 2, 1, 1, 1, 2, 2]

ele = x x 2

cnt = 1 x x 0 x 0 x 2

cnt > 1 return ele
else -1

⇒ count - For tracking count of element
element - for which element we are counting

⇒ Traverse :-

- If count is 0 then store the current element of the array as Element
- If the current element and element are the same increase the count by 1.
- If they are different decrease by 1

3. The integer present in element should be the result.

```
class Solution {
public:
    int majorityElement(vector<int>& nums) {
        int count=0,res=0;
        for(auto num:nums){
            if(count==0){
                res=num;
            }

            if(res==num){
                count++;
            }
            else{
                count--;
            }
            // cout<<"c: "<<count<<endl;
        }
        return res;
    }
};
```

If count become zero then new element is there so need to count new element

If same element found ++

not found count --

T.C - O(n)

S.C - O(1)