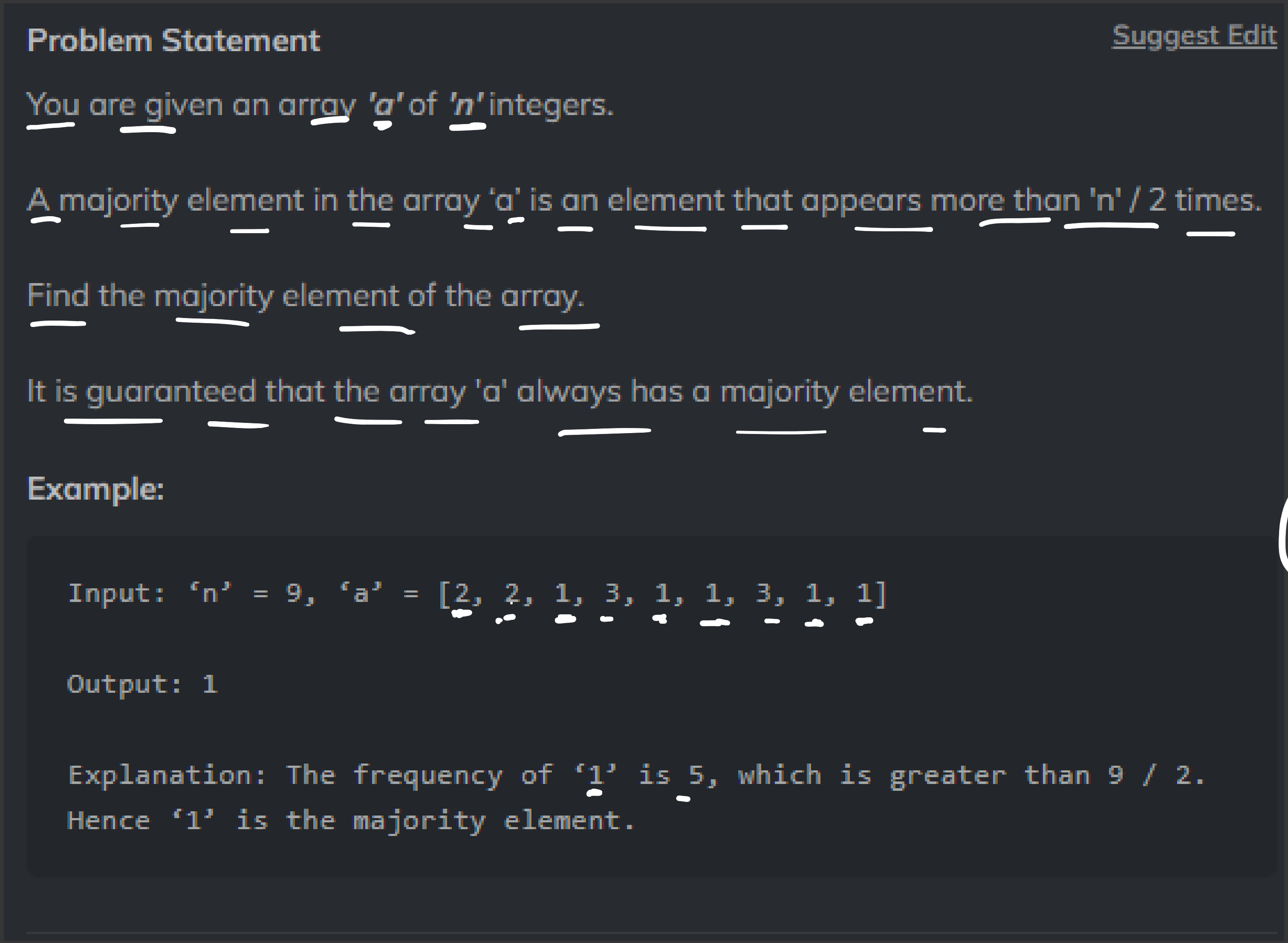
Find the Majority Element that occurs more than n/2 times



**Problem Statement**

Suggest Edit

You are given an array *'a'* of *'n'* integers.

A majority element in the array 'a' is an element that appears more than 'n' / 2 times.

Find the majority element of the array.

It is guaranteed that the array 'a' always has a majority element.

**Example:**

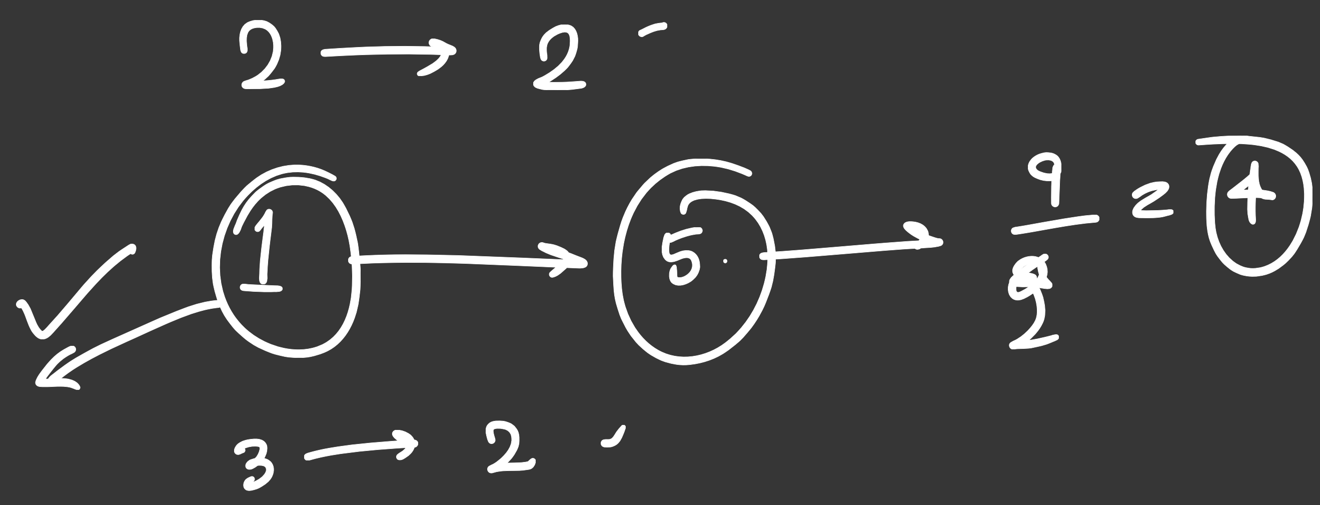
```
Input: ‘n’ = 9, ‘a’ = [2, 2, 1, 3, 1, 1, 3, 1, 1]

Output: 1

Explanation: The frequency of ‘1’ is 5, which is greater than 9 / 2.
Hence ‘1’ is the majority element.
```

9

$9/2 = 4$



ans : 1

eg: $n = 5$

arr[] : {-53, 75, 56, 56, 56}

56

$n/2 = 5/2 = 2$

-53 → 1

75 → 1

56 → 3 →

## Brute force.

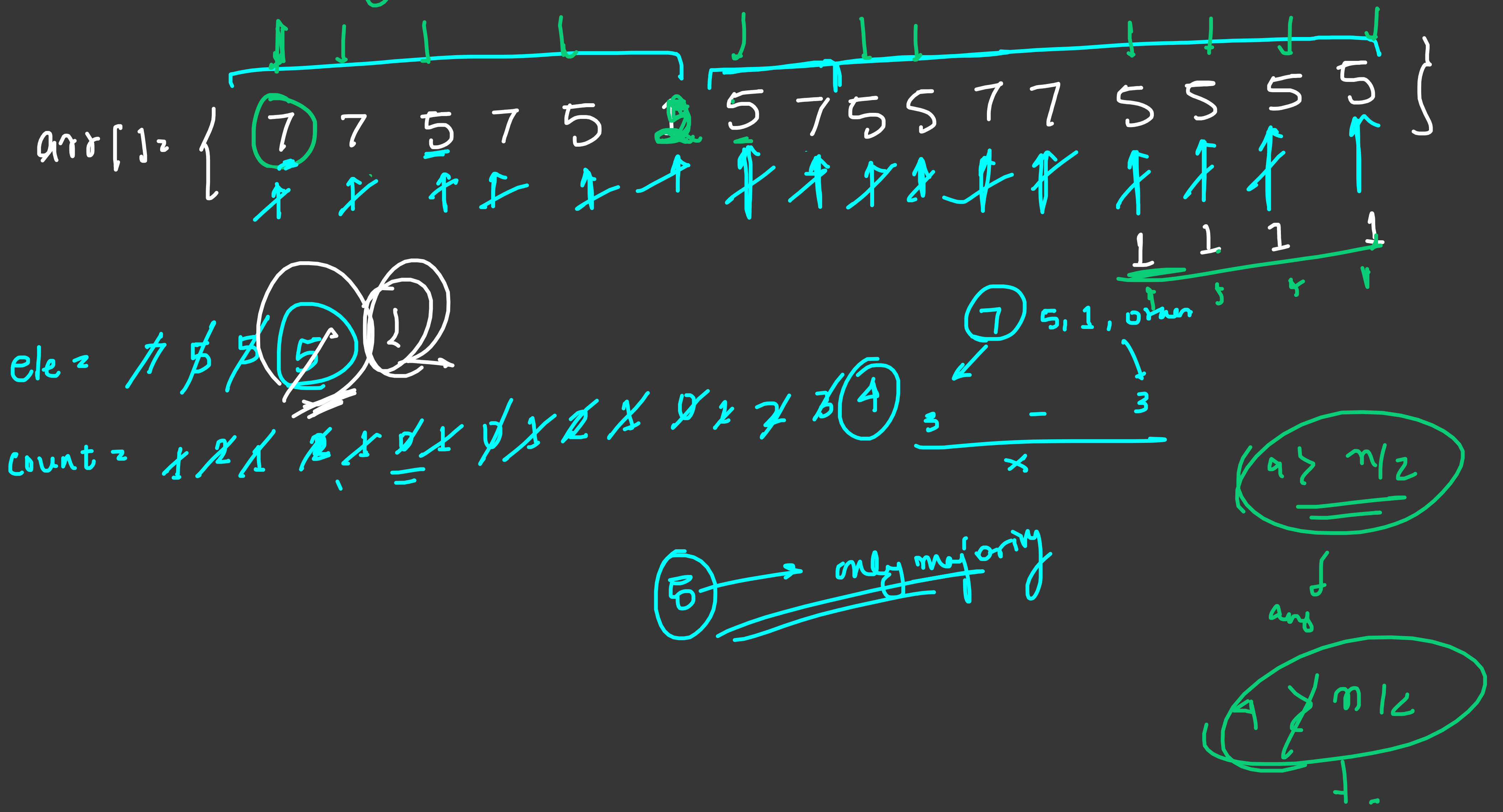
hashmaps (arr[i], freq)

Count the freq → $O(N^2)$

Hashmap → $O(2N)$

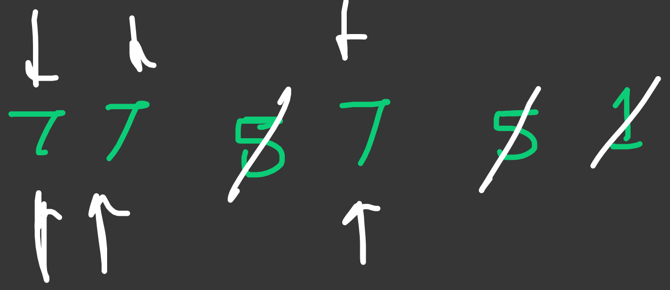
## Optimal Approach:

- Moore's Voting Algorithm:



## • Moori's Voting Algorithm:

→ We count the current element ~~when~~ and when element is not found we reduce the count, when count we reaches zero, we update the element.



7 → 3

5, 1 → 3

} It cancel out

1. Maintain element and count variable.

2. Increase count if found the element or else decrease if count reaches zero update the element.

3. The element at last we got can be a majority element.

4. So, we iterate and check if the count of element is the greater than $n/2$ then we return elem or else $-1$

## Pseudocode:-

```
ele, count = 0

for(i=0 → n)
        if( count==0) ele = arr[i], count=1.
        else if( arr[i]==ele) count++.
            else. count--;

int count1 = 0

 for (i=0 → n)
          if( arr[i]==ele) : count1++ .

 if (count1 > n/2) return ele,
  else return -1.
```