**229. Majority Element II: -**

Medium Accepted: 518.5K Submissions: 1.1M Acceptance Rate: 48.6%

Given an integer array of size n, find all elements that appear more than ⌊ n/3 ⌋ times.

**Example 1:**

**Input:** nums = [3,2,3]

**Output:** [3]

**Example 2:**

**Input:** nums = [1]

**Output:** [1]

**Example 3:**

**Input:** nums = [1,2]

**Output:** [1,2]

**Constraints:**

* 1 <= nums.length <= 5 \* 104
* -109 <= nums[i] <= 109

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

**Code: -**

class Solution {

public:

    vector<int> majorityElement(vector<int>& nums) {

        // using Boyer-Moore's Majority Voting Algorithm

        int n = nums.size();

        int num1, num2, count1=0, count2=0;

        for(int i=0; i<n; ++i){

            if(num1 == nums[i])

                ++count1;

            else if(num2 == nums[i])

                ++count2;

            else if(count1 == 0){

                count1 = 1;

                num1 = nums[i];

            }

            else if(count2 == 0){

                count2 = 1;

                num2 = nums[i];

            }

            else{

                --count1;

                --count2;

            }

        }

        count1 = count2 = 0;

        for(auto &i : nums){

            if(i == num1)

                ++count1;

            else if(i == num2)

                ++count2;

        }

        vector<int> ans;

        if(count1 > n/3)        ans.push\_back(num1);

        if(count2 > n/3)        ans.push\_back(num2);

        return ans;

    }

};

**T.C: - O(N)**

**S.C: - O(1)**