Majority Element II - LeetCode 229

Problem: Find all elements that appear more than ■ n/3 ■ times in the array.

Key Idea: At most 2 elements can appear more than n/3 times.
We explore two solutions:
1. HashMap Counting (O(n) time, O(n) space)
2. Boyer-Moore Voting Algorithm (O(n) time, O(1) space)

Solution 1: HashMap Counting

Idea: Count frequencies of each element using a HashMap. Any element with frequency ≥ ■n/3■ + 1 is added to the answer.

Time Complexity: O(n)
Space Complexity: O(n)

```
class Solution {
public:
    vector<int> majorityElement(vector<int>& nums) {
        int n = nums.size();
        int minimum = n / 3 + 1;
        unordered_map<int, int> hashMap;
        vector<int> ans;
        for(int i = 0; i < n; i++){
            hashMap[nums[i]]++;
            if(hashMap[nums[i]] == minimum){
                ans.push_back(nums[i]);
            }
        }
        return ans;
    }
};
```

Solution 2: Boyer-Moore Voting Algorithm

Idea: At most 2 majority elements exist. Track two candidates with counters. After one pass to select candidates, perform a second pass to verify counts.

Time Complexity: O(n)
Space Complexity: O(1)

```
class Solution {
public:
    vector<int> majorityElement(vector<int>& nums) {
        int count1 = 0, count2 = 0;
        int element1 = INT_MIN, element2 = INT_MIN, n = nums.size();
        vector<int> ans;

        for(int i = 0; i < n; i++){
            if(count1 == 0 && nums[i] != element2){
                count1 = 1;
                element1= nums[i];
            }
            else if(count2 == 0 && nums[i] != element1){
                count2 = 1;
                element2= nums[i];
            }
            else if(nums[i] == element1){
                count1++;
            }
            else if(nums[i] == element2){
                count2++;
            }
            else{
                count1--;
                count2--;
            }
        }

        int minimum = n / 3 + 1;
        count1 = 0, count2 = 0;
        for(int i = 0; i < n; i++){
            if(nums[i] == element1) count1++;
            else if(nums[i] == element2) count2++;
        }

        if(count1 >= minimum) ans.push_back(element1);
        if(count2 >= minimum) ans.push_back(element2);

        return ans;
    }
};
```