


## HW 12: Majority Element (Leetcode-169)

### PROBLEM STATEMENT:

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.

 **ISKA MTLB YE HAI:** majority element always equal to (mid) index par exist krega jab array sorted form me hogaa;

### Example 1:

Input: nums = [3,2,3]

Output: 3

### Example 2:

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

Output: 2

### **BRUTE FORCE APPROACH 01:**

Step 01: count each elements

Step 02: compare each elements

Step 03: return highest counted element = majority element

**Time Complexity:**  $O(N^2)$ , where  $N$  is Length of array

**Space Complexity:**  $O(1)$ , no extra space used

### **BINARY SEARCH APPROACH 02:**

Step 01: first occurrence of mid

Step 02: Last occurrence of mid

Step 03: total occurrence of mid = majority element

**Time Complexity:**  $O(N \log N)$ , where  $N$  is Length of array

**Space Complexity:**  $O(1)$ , no extra space used

### **SORTING APPROACH 03:**

Step 01: sort the array

Step 02: return the mid index element = majority element

**Time Complexity:**  $O(N \log N)$ , where  $N$  is Length of array

**Space Complexity:**  $O(1)$ , no extra space used

DRY Run

Example 1:

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

Output: 3

3	2	3
0	1	2

BRUTE FORCE APPROACH 01:

3 → 2

2 → 1

Output: 3

SORTING APPROACH 03:

2	3	3
0	1	2

Always mid Index  
element is majority  
element = 3

BINARY SEARCH APPROACH 02:

SORT

2	3	3
0	1	2

$$\text{Mid} = \frac{0+2}{2} = 1$$

$$\text{Mid Index element} = \text{nums}[\text{mid}] = 3$$

$$\text{Last occ. of mid} = 2$$

$$\text{First occ. of mid} = 1$$

$$\text{Total occ. of mid} = 2 - 1 + 1 = 2$$

$$\Rightarrow \text{Return } \text{nums}[2] = 3 \text{ Output}$$

DRY RUN

**Example 2:**

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

Output: 2

2	2	1	1	1	2	2
0	1	2	3	4	5	6

BRUTE FORCE APPROACH 01:

2 → 4  
1 → 3

Output = 2

SORTING APPROACH 03:

1	1	1	<del>2</del>	2	2	2
0	1	2	3	4	5	6

Always mid Index

Element is majority

Element = 2 at mid Index

BINARY SEARCH APPROACH 02:

SORT

1	1	1	2	2	2	2
0	1	2	3	4	5	6

$$\text{mid} = \frac{0+6}{2} = 3$$

$$\text{mid Index Element} = \text{nums}[\text{mid}]$$

$$= 2$$

$$\text{Last occ. of mid} = 6$$

$$\text{First occ. of mid} = 3$$

$$\text{Total occ. of mid} = 6 - 3 + 1$$

$$= 4$$

$$\rightarrow \text{Return } \text{nums}[4] = 2 \quad \text{Output}$$

### SORTING APPROACH 03: PROGRAM

```
// HW 12: Majority Element (Leetcode-169)
//  SORTING APPROACH 03:
class Solution {
public:
    int majorityElement(vector<int>& nums) {
        sort(nums.begin(), nums.end());
        int midIndex = (0 + nums.size()-1)/2;
        return nums[midIndex];
    }
};
```