Started on	Friday, 4 October 2024, 1:51 PM
State	Finished
Completed on	Friday, 4 October 2024, 1:52 PM
Time taken	49 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

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`
- `-2^31 <= nums[i] <= 2^31 - 1`

For example:

Input	Result
3 3 2 3	3
7 2 2 1 1 1 2 2	2

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
void sort(int arr[],int temp[],int low,int high)
{
    for(int i=low; i<=high;i++)
    {
        arr[i]=temp[i-low];
    }
}

void merge(int arr[],int low,int mid,int high)
{
    int temp[high+1];
    int p=low,q=mid+1,s=0;
    while(p<=mid && q<=high)
    {
        if(arr[p]<arr[q])
        {
            temp[s]=arr[p];
            p++;
        }
        else
        {
            temp[s]=arr[q];
            q++;
        }
        s++;
    }
    while(p<=mid)
    {
        temp[s]=arr[p];
        s++;
        p++;
    }
    while(q<=high)
```

```
    {
        temp[s]=arr[q];
        s++;
        q++;
    }
    sort(arr,temp,low,high);
}

void mergesort(int arr[],int low,int high)
{
    if(low < high)
    {
        int mid=(low+high)/2;
        mergesort(arr,low,mid);
        mergesort(arr,mid+1,high);
        merge(arr,low,mid,high);
    }
}
```

	Input	Expected	Got	
✔	3 3 2 3	3	3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ 1-Number of Zeros in a Given Array

Jump to...

3-Finding Floor Value ►