Divide and conquer

Roll no.:241801266

Name : shyam ganesh d

# 1-Number of Zeros in a Given Array

| | |
|---|---|
| Started on | Friday, 19 September 2025, 10:09 AM |
| State | Finished |
| Completed on | Friday, 19 September 2025, 10:22 AM |
| Time taken | 13 mins 54 secs |
| Marks | 1.00/1.00 |
| Grade | 10.00 out of 10.00 (100%) |

Question 1 | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

**Problem Statement**

Given an array of 1s and 0s this has all 1s first followed by all 0s. Aim is to find the number of 0s. Write a program using Divide and Conquer to Count the number of zeroes in the given array.

Input Format

First Line Contains Integer m – Size of array

Next m lines Contains m numbers – Elements of an array

Output Format

First Line Contains Integer – Number of zeroes present in the given array.

```c
#include<stdio.h>

int findzero(int a[],int low,int high){
    if(low>high)
        return -1;
    int mid=(low+high)/2;
    if(a[mid]==0 &&(mid==0 || a[mid-1]==1))
        return mid;
    if(a[mid]==1)
        return findzero(a,mid+1,high);
    return findzero(a,low,mid-1);
}
int count(int a[],int m){
    int total=findzero(a,0,m-1);
    if(total==-1){
        return 0;
    }

    return m-total;
}

int main(){
    int m;
    scanf("%d ",&m);
    int a[m];
    for(int i=0;i<m;i++){
        scanf("%d",&a[i]);
    }
    printf("%d ",count(a,m));
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✔ | 5<br>1<br>1<br>1<br>0<br>0 | 2 | 2 | ✔ |
| ✔ | 10<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1 | 0 | 0 | ✔ |
| ✔ | 8<br>0<br>0<br>0<br>0<br>0<br>0<br>0<br>0 | 8 | 8 | ✔ |
| ✔ | 17<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>1<br>0<br>0 | 2 | 2 | ✔ |

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

# 2-Majority Element

|  |  |
|---|---|
| Started on | Friday, 19 September 2025, 10:24 AM |
| State | Finished |
| Completed on | Friday, 19 September 2025, 10:41 AM |
| Time taken | 16 mins 41 secs |
| Marks | 1.00/1.00 |
| Grade | **10.00** out of 10.00 (**100%**) |

**Question 1** | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

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<br>3 2 3 | 3 |
| 7<br>2 2 1 1 1 2 2 | 2 |

```c
#include<stdio.h>
int count(int a[],int m){
    int num=-1,count=0;
    for(int i=0;i<m;i++){
        if(count==0){
            num=a[i];
            count=1;


        }


        else if(a[i]==num){
            count++;
        }
        else{
            count--;
        }
    }
    return num;

}
int main(){
    int m;
    scanf("%d ",&m);
    int a[m];
    for(int i=0;i<m;i++){
        scanf("%d ",&a[i]);
    }
    printf("%d ",count(a,m));
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✔ | 3<br>3 2 3 | 3 | 3 | ✔ |

Passed all tests! ✔

# 3-Finding Floor Value

| Started on | Friday, 19 September 2025, 10:42 AM |
| State | Finished |
| Completed on | Friday, 19 September 2025, 10:57 AM |
| Time taken | 15 mins 24 secs |
| Marks | 1.00/1.00 |
| Grade | **10.00** out of 10.00 (**100**%) |

**Question 1** | Correct   Mark 1.00 out of 1.00   🚩 Flag question

**Problem Statement:**

Given a sorted array and a value x, the floor of x is the largest element in array smaller than or equal to x. Write divide and conquer algorithm to find floor of x.

**Input Format**

    First Line Contains Integer n – Size of array

    Next n lines Contains n numbers – Elements of an array

    Last Line Contains Integer x – Value for x

**Output Format**

    First Line Contains Integer – Floor value for x

```c
#include<stdio.h>
int find(int a[],int low,int high,int x){
    if(low>high){
        return -1;
    }
    int mid=(low+high)/2;
    if(a[mid]==x){
        return a[mid];
    }
    else if(a[mid]>x){
        return find(a,low,mid-1,x);
    }
    else{
        int total=find(a,mid+1,high,x);
        if(total==-1 || (total>x)){
            return a[mid];
        }
        else{
            return total;
        }
    }

}
int main(){
    int m;
    scanf("%d ",&m);
    int a[m];
    for(int i=0;i<m;i++){
        scanf("%d ",&a[i]);
    }
    int x;
    scanf("%d ",&x);
    int final=find(a,0,m-1,x);
    printf("%d ",final);
}
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✔ | 6<br>1<br>2<br>8<br>10<br>12<br>19<br>5 | 2 | 2 | ✔ |
| ✔ | 5<br>10<br>22<br>85<br>108<br>129<br>100 | 85 | 85 | ✔ |
| ✔ | 7<br>3<br>5<br>7<br>9<br>11<br>13<br>15<br>10 | 9 | 9 | ✔ |

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

# 4-Two Elements sum to x

## Problem Statement:

Given a sorted array of integers say arr[] and a number x. Write a recursive program using divide and conquer strategy to check if there exist two elements in the array whose sum = x. If there exist such two elements then return the numbers, otherwise print as "No".

Note: Write a Divide and Conquer Solution

### Input Format

  First Line Contains Integer n – Size of array

  Next n lines Contains n numbers – Elements of an array

  Last Line Contains Integer x – Sum Value

### Output Format

  First Line Contains Integer – Element1

  Second Line Contains Integer – Element2 (Element 1 and Elements 2 together sums to value "x")

**Answer:** (penalty regime: 0 %)

```c
1  #include<stdio.h>
2  void find(int a[],int low,int high,int x){
3      if(low>=high){
4          printf("No");
5          return;
6      }
7      int sum=a[low]+a[high];
8      if(sum==x){
9          printf("%d\n%d\n",a[low],a[high]);
10     }
11     else if(sum<x){
12         return find(a,low+1,high,x);
13     }
14     else{
15         return find(a,low,high+1,x);
16     }
17
18 }
19 int main(){
20     int n;
21     scanf("%d ",&n);
22     int a[n];
23     for(int i=0;i<n;i++){
24         scanf("%d ",&a[i]);
25     }
26     int x;
27     scanf("%d ",&x);
28     find(a,0,n-1,x);
29     return 0;
30 }
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✔ | 4<br>2<br>4<br>8<br>10<br>14 | 4<br>10 | 4<br>10 | ✔ |
| ✔ | 5<br>2<br>4<br>6<br>8<br>10<br>100 | No | No | ✔ |

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00

# 5-Implementation of Quick Sort

| | |
|---|---|
| **Started on** | Friday, 19 September 2025, 6:19 PM |
| **State** | Finished |
| **Completed on** | Friday, 19 September 2025, 6:27 PM |
| **Time taken** | 8 mins 33 secs |
| **Marks** | 1.00/1.00 |
| **Grade** | **10.00** out of 10.00 (**100%**) |

**Question 1** | Correct   Mark 1.00 out of 1.00   ⚑ Flag question

Write a Program to Implement the Quick Sort Algorithm

Input Format:
The first line contains the no of elements in the list-n
The next n lines contain the elements.

Output:
Sorted list of elements

**For example:**

| Input | Result |
|---|---|
| 5<br>67 34 12 98 78 | 12 34 67 78 98 |

```c
1   #include <stdio.h>
2
3   void swap(int *a, int *b) {
4       int temp = *a;
5       *a = *b;
6       *b = temp;
7   }
8
9   int partition(int arr[], int low, int high) {
10      int pivot = arr[high];
11      int i = low - 1;
12      for (int j = low; j < high; j++) {
13          if (arr[j] <= pivot) {
14              i++;
15              swap(&arr[i], &arr[j]);
16          }
17      }
18      swap(&arr[i + 1], &arr[high]);
19      return i + 1;
20  }
21
22  void quickSort(int arr[], int low, int high) {
23      if (low < high) {
24          int p = partition(arr, low, high);
25          quickSort(arr, low, p - 1);
26          quickSort(arr, p + 1, high);
27      }
28  }
29
30  int main() {
31      int n;
32      scanf("%d", &n);
33      int arr[n];
34      for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
35      quickSort(arr, 0, n - 1);
36      for (int i = 0; i < n; i++) printf("%d ", arr[i]);
37      return 0;
38  }
39
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✔ | 5<br>67 34 12 98 78 | 12 34 67 78 98 | 12 34 67 78 98 | ✔ |
| ✔ | 10<br>1 56 78 90 32 56 11 10 90 114 | 1 10 11 32 56 56 78 90 90 114 | 1 10 11 32 56 56 78 90 90 114 | ✔ |
| ✔ | 12<br>9 8 7 6 5 4 3 2 1 10 11 90 | 1 2 3 4 5 6 7 8 9 10 11 90 | 1 2 3 4 5 6 7 8 9 10 11 90 | ✔ |

Passed all tests! ✔