

**Question 1** | Correct Mark 1.00 out of 1.00**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.

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

```

1  #include<stdio.h>
2  int FindFirstZero(int a[],int low,int high)
3  {
4      if(low>high)
5      {
6          return -1;
7      }
8      int mid=(low+high)/2;
9      if(a[mid]==0)
10     {
11         if(mid==0||a[mid-1]==1)
12             return mid;
13         else
14             return FindFirstZero(a,low,mid-1);
15     }
16     else
17     {
18         return FindFirstZero(a,mid+1,high);
19     }
20 }
21 int main()
22 {
23     int m;
24     scanf("%d",&m);
25     int a[m];
26     for(int i=0;i<m;i++)
27     {
28         scanf("%d",&a[i]);
29     }
30     int index=FindFirstZero(a,0,m-1);
31     if(index==-1)
32         printf("0\n");
33     else
34         printf("%d\n",m-index);
35     return 0;
36 }

```

	Input	Expected	Got	
✓	5	2	2	✓
	1			
	1			
	1			
	0			
	0			

	Input	Expected	Got	
✓	10 1 1 1 1 1 1 1 1 1 1 1	0	0	✓
✓	8 0 0 0 0 0 0 0 0 0	8	8	✓
✓	17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0	2	2	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)

**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  $\lfloor n / 2 \rfloor$  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 * 104`
- `-231 <= nums[i] <= 231 - 1`

**For example:**

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

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

```

1  #include<stdio.h>
2  int MajorityElement(int*nums,int numsSize)
3  {
4      int candidate=0,count=0;
5      for(int i=0;i<numsSize;i++)
6      {
7          if(count==0)
8          {
9              candidate=nums[i];
10             }
11             count+=(nums[i]==candidate)?1:-1;
12         }
13         return candidate;
14     }
15 int main()
16 {
17     int n;
18     scanf("%d",&n);
19     int nums[n];
20     for(int i=0;i<n;i++)
21     {
22         scanf("%d",&nums[i]);
23     }
24     int result=MajorityElement(nums,n);
25     printf("%d\n",result);
26     return 0;
27 }
```

	Input	Expected	Got	
✓	3	3	3	✓
	3 2 3			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)

**Question 1** | Correct Mark 1.00 out of 1.00**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

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

```
1  #include<stdio.h>
2  int FindFloor(int a[],int low,int high,int x)
3  {
4      if(low>high)
5      {
6          return -1;
7      }
8      int mid=low+(high-low)/2;
9      if(a[mid]==x)
10     {
11         return a[mid];
12     }
13     if(a[mid]>x)
14     {
15         return FindFloor(a,low,mid-1,x);
16     }
17     int FloorRight=FindFloor(a,mid+1,high,x);
18     if(FloorRight!=-1)
19     {
20         return a[mid];
21     }
22     else
23     {
24         return FloorRight;
25     }
26 }
27 int main()
28 {
29     int n;
30     scanf("%d",&n);
31     int a[n];
32     for(int i=0;i<n;i++)
33     {
34         scanf("%d",&a[i]);
35     }
36     int x;
37     scanf("%d",&x);
38     int FloorVal=FindFloor(a,0,n-1,x);
39     printf("%d\n",FloorVal);
40     return 0;
41 }
```



	Input	Expected	Got	
✓	6	2	2	✓
	1			
	2			
	8			
	10			
	12			
	19			
	5			
✓	5	85	85	✓
	10			
	22			
	85			
	108			
	129			
	100			
✓	7	9	9	✓
	3			
	5			
	7			
	9			
	11			
	13			
	15			
	10			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)

**Question 1** | Correct Mark 1.00 out of 1.00**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 %)

```

1  #include<stdio.h>
2  int FindPair(int a[],int left,int right,int x,int*e1,int*e2)
3  {
4      if(left>=right)
5      {
6          return 0;
7      }
8      int sum=a[left]+a[right];
9      if(sum==x)
10     {
11         *e1=a[left];
12         *e2=a[right];
13         return 1;
14     }
15     else if(sum<x)
16     {
17         return FindPair(a,left+1,right,x,e1,e2);
18     }
19     else
20     {
21         return FindPair(a,left,right-1,x,e1,e2);
22     }
23 }
24 int main()
25 {
26     int n;
27     scanf("%d",&n);
28     int a[n];
29     for(int i=0;i<n;i++)
30     {
31         scanf("%d",&a[i]);
32     }
33     int x;
34     scanf("%d",&x);
35     int e1,e2;
36     if(FindPair(a,0,n-1,x,&e1,&e2))
37     {
38         printf("%d\n%d\n",e1,e2);
39     }
40     else
41     {
42         printf("No\n");
43     }
44     return 0;
45 }
```



	Input	Expected	Got	
✓	4	4	4	✓
	2	10	10	
	4			
	8			
	10			
	14			
✓	5	No	No	✓
	2			
	4			
	6			
	8			
	10			
	100			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)



**Question 1** | Correct Mark 1.00 out of 1.00

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	12 34 67 78 98
67 34 12 98 78	

Answer:

```

1  #include<stdio.h>
2  void swap(int*a,int*b)
3  {
4      int temp=*a;
5      *a=*b;
6      *b=temp;
7  }
8  int partition(int a[],int low,int high)
9  {
10     int pivot=a[high];
11     int i=low-1;
12     for(int j=low;j<=high-1;j++)
13     {
14         if(a[j]<pivot)
15         {
16             i++;
17             swap(&a[i],&a[j]);
18         }
19     }
20     swap(&a[i+1],&a[high]);
21     return(i+1);
22 }
23 void QuickSort(int a[],int low,int high)
24 {
25     if(low<high)
26     {
27         int pi=partition(a,low,high);
28         QuickSort(a,low,pi-1);
29         QuickSort(a,pi+1,high);
30     }
31 }
32 int main()
33 {
34     int n;
35     scanf("%d",&n);
36     int a[n];
37     for(int i=0;i<n;i++)
38     {
39         scanf("%d",&a[i]);
40     }
41     QuickSort(a,0,n-1);
42     for(int i=0;i<n;i++)
43     {
44         printf("%d",a[i]);
45         if(i!=n-1)
46         {
47             printf(" ");
48         }
49     }
50     printf("\n");
51     return 0;
52 }

```

	Input	Expected	Got	
✓	5 67 34 12 98 78	12 34 67 78 98	12 34 67 78 98	✓
✓	10 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 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! ✓

Correct

Marks for this submission: 1.00/1.00.

[Back to Course](#)