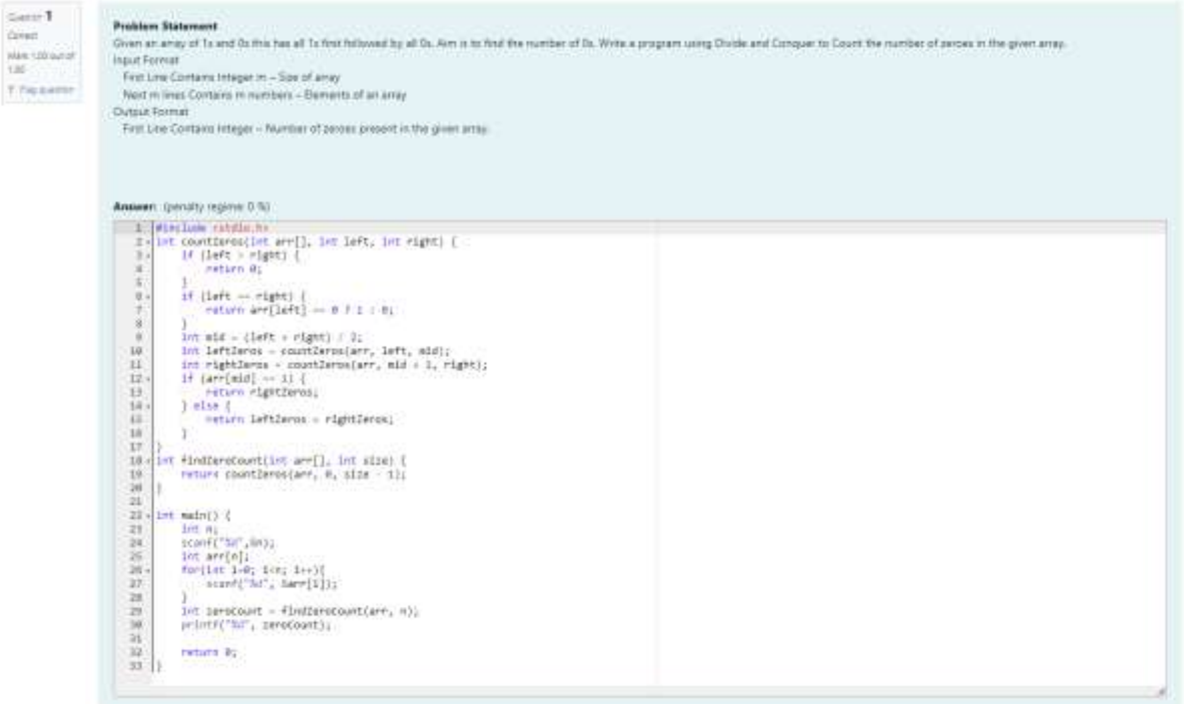
DIVIDE AND CONQUER

1.



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

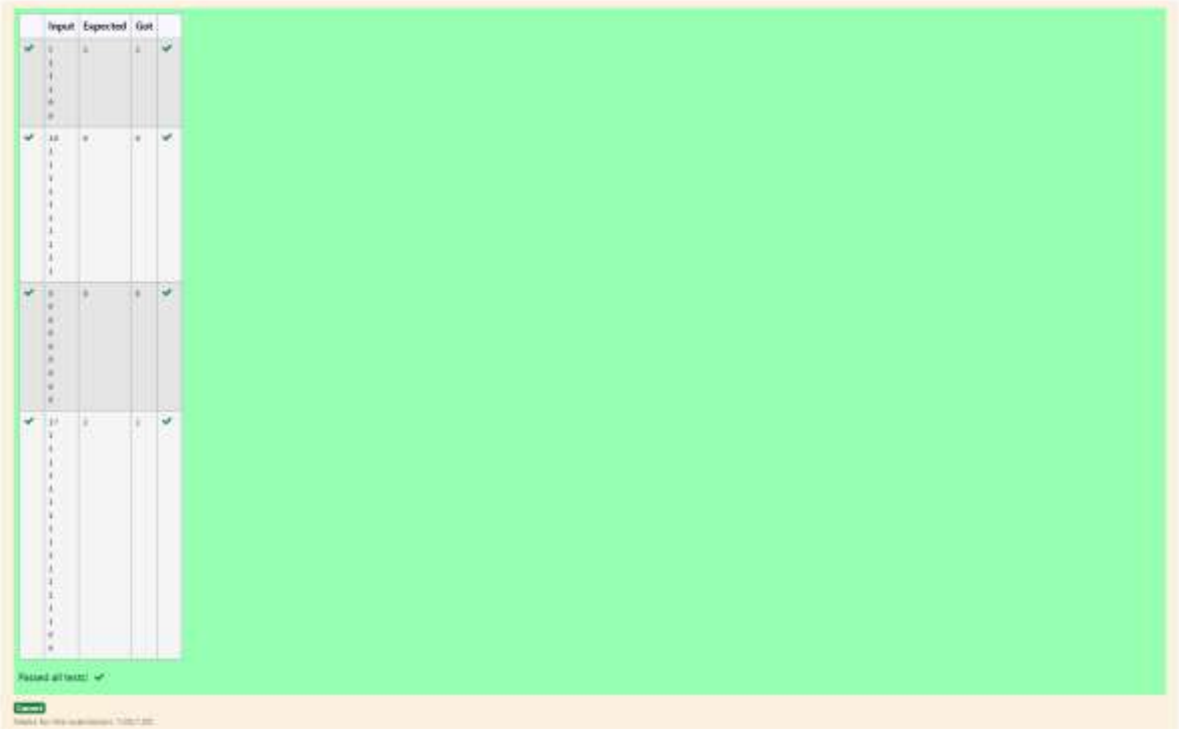
First Line Contains Integer – Number of zeroes present in the given array.

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
int countZeros(int arr[], int left, int right) {
    if (left > right) {
        return 0;
    }
    if (left == right) {
        return arr[left] == 0 ? 1 : 0;
    }
    int mid = (left + right) / 2;
    int leftZeros = countZeros(arr, left, mid);
    int rightZeros = countZeros(arr, mid + 1, right);
    if (arr[mid] == 1) {
        return rightZeros;
    } else {
        return leftZeros + rightZeros;
    }
}
int findZeroCount(int arr[], int size) {
    return countZeros(arr, 0, size - 1);
}

int main() {
    int n;
    scanf("%d",&n);
    int arr[n];
    for(int i=0; i<n; i++){
        scanf("%d", &arr[i]);
    }
    int zeroCount = findZeroCount(arr, n);
    printf("%d", zeroCount);

    return 0;
}
```



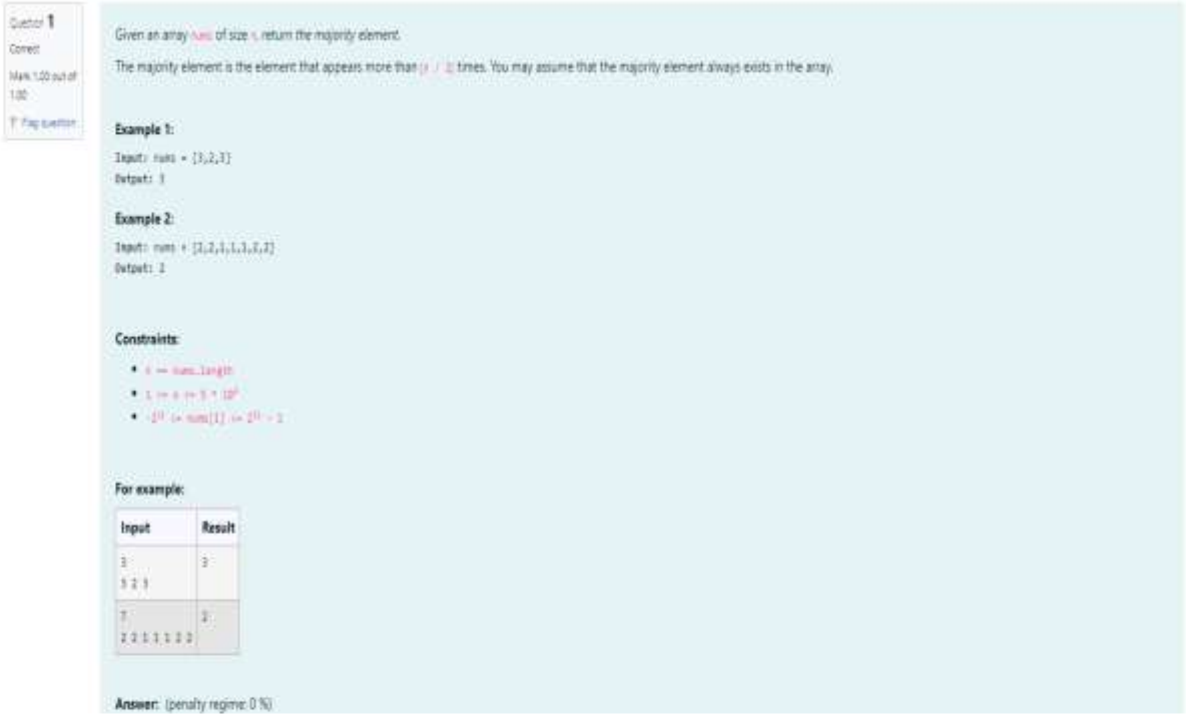
	Input	Expected	Got	
✓	5 1 1 1 0 0	2	2	✓
✓	10 1 1 1 1 1 1 1 1 1 1	0	0	✓
✓	8 0 0 0 0 0 0 0 0	8	8	✓
✓	17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0	2	2	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

2. DIVIDE AND CONQUER



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
```

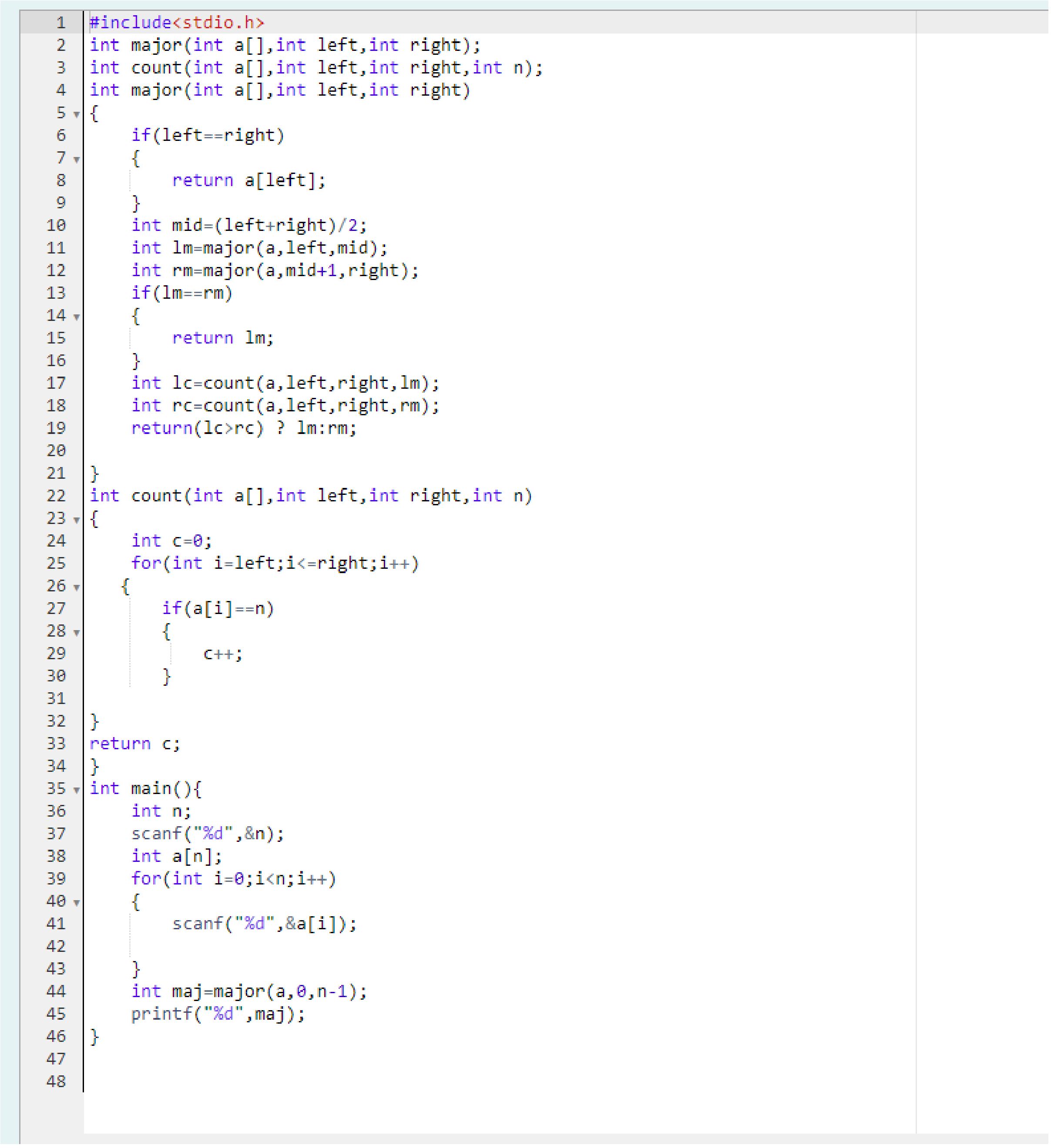
Constraints:

- `n == nums.length`
- `1 <= n <= 5 * 10`^{4}
- `-2`^{31}` <= nums[i] <= 2`^{31}` - 1`

For example:

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

Answer: (penalty regime: 0 %)



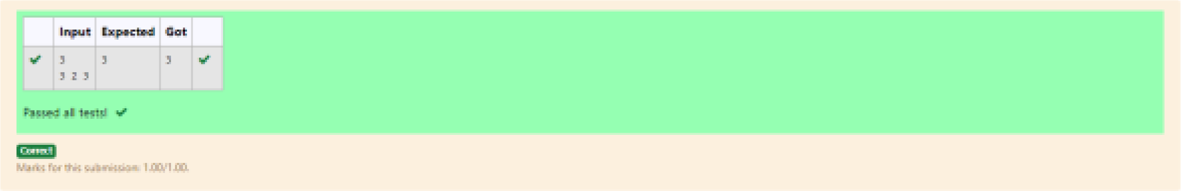
```c
#include<stdio.h>
int major(int a[],int left,int right);
int count(int a[],int left,int right,int n);
int major(int a[],int left,int right)
{
    if(left==right)
    {
        return a[left];
    }
    int mid=(left+right)/2;
    int lm=major(a,left,mid);
    int rm=major(a,mid+1,right);
    if(lm==rm)
    {
        return lm;
    }
    int lc=count(a,left,right,lm);
    int rc=count(a,left,right,rm);
    return(lc>rc) ? lm:rm;

}
int count(int a[],int left,int right,int n)
{
    int c=0;
    for(int i=left;i<=right;i++)
   {
       if(a[i]==n)
       {
           c++;
       }

}
return c;
}
int main(){
    int n;
    scanf("%d",&n);
    int a[n];
    for(int i=0;i<n;i++)
    {
        scanf("%d",&a[i]);

    }
    int maj=major(a,0,n-1);
    printf("%d",maj);
}


```



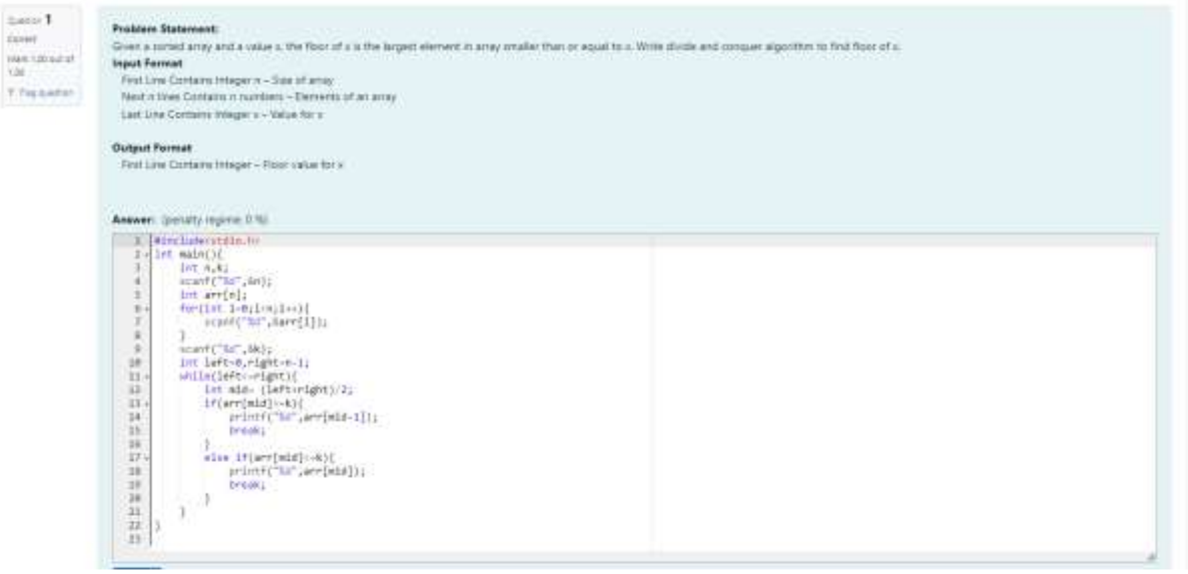
	Input	Expected	Got	
✓	3 3 2 3	3	3	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

3. DIVIDE AND CONQUER



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

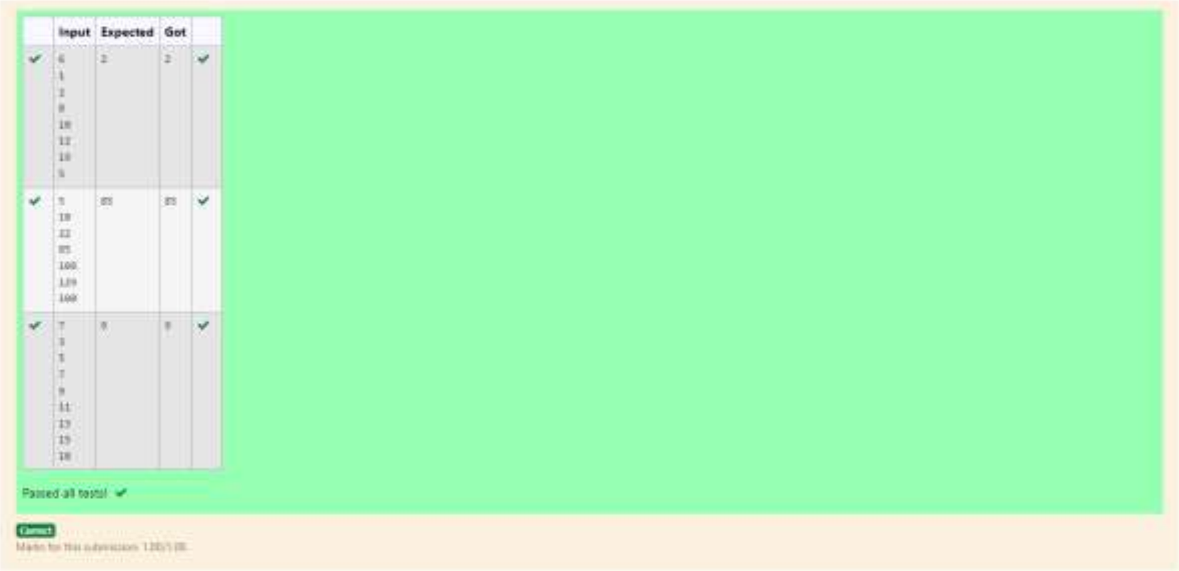
Last Line Contains Integer x – Value for x

Output Format

First Line Contains Integer – Floor value for x

Answer: (penalty regime: 0 %)

```c
#include<stdio.h>
int main(){
    int n,k;
    scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    scanf("%d",&k);
    int left=0,right=n-1;
    while(left<=right){
        int mid= (left+right)/2;
        if(arr[mid]>=k){
            printf("%d",arr[mid-1]);
            break;
        }
        else if(arr[mid]<=k){
            printf("%d",arr[mid]);
            break;
        }
    }
}
```



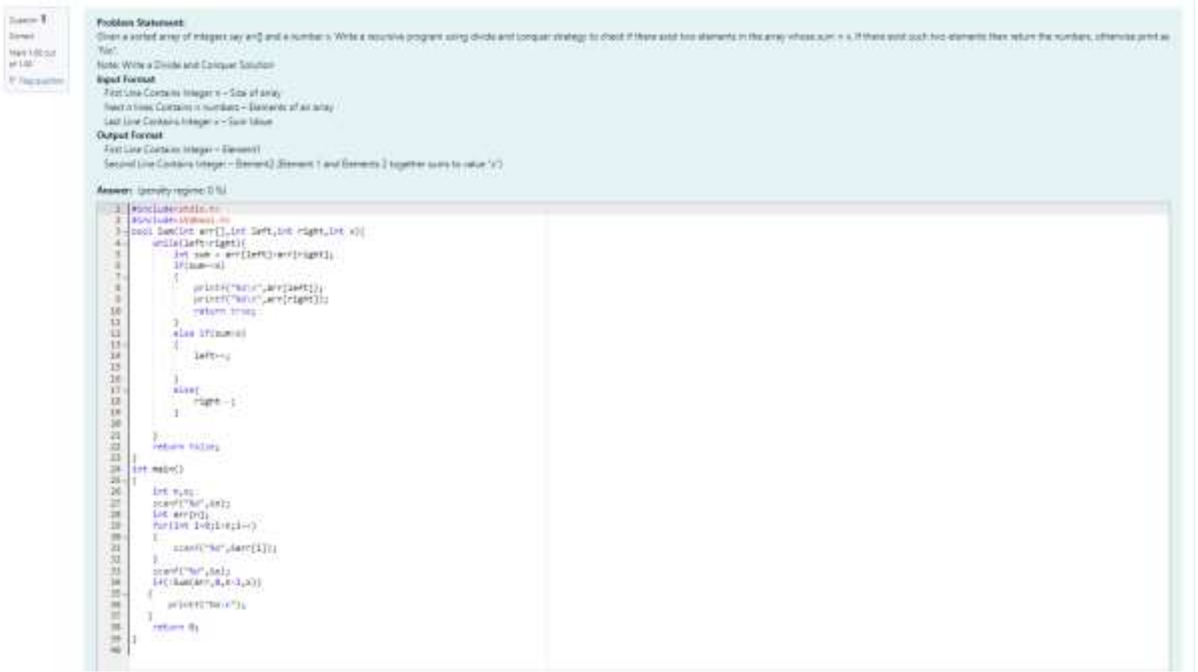
	Input	Expected	Got	
✓	6 1 2 8 10 12 19 5	2	2	✓
✓	5 10 22 85 108 129 100	85	85	✓
✓	7 3 5 7 9 11 13 15 10	9	9	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

4.DIVIDE AND CONQUER



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

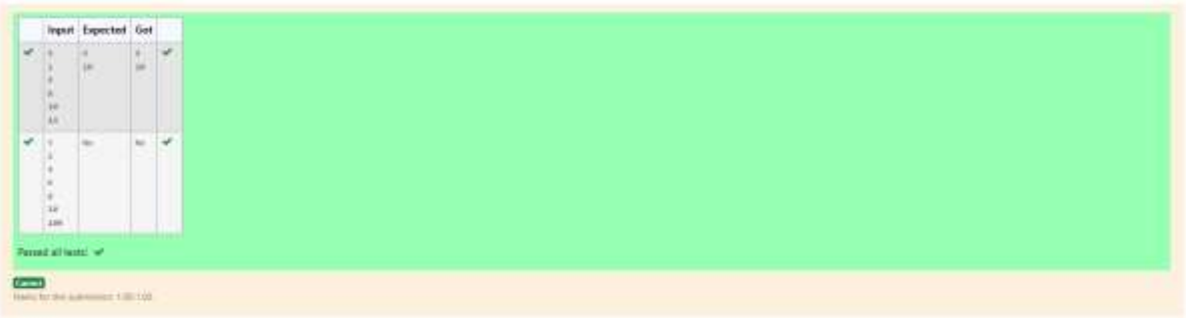
Second Line Contains Integer – Element2 (Element 1 and Elements 2 together sums to value "x")

Answer: (penalty regime: 0 %)

```
#include<stdio.h>
#include<stdbool.h>
bool Sum(int arr[],int left,int right,int x){
    while(left<right){
        int sum = arr[left]+arr[right];
        if(sum==x)
        {
            printf("%d\n",arr[left]);
            printf("%d\n",arr[right]);
            return true;
        }
        else if(sum<x)
        {
            left++;

        }
        else{
            right--;
        }

    }
    return false;
}
int main()
{
    int n,x;
    scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    scanf("%d",&x);
    if(!Sum(arr,0,n-1,x))
    {
        printf("No\n");
    }
    return 0;
}
```



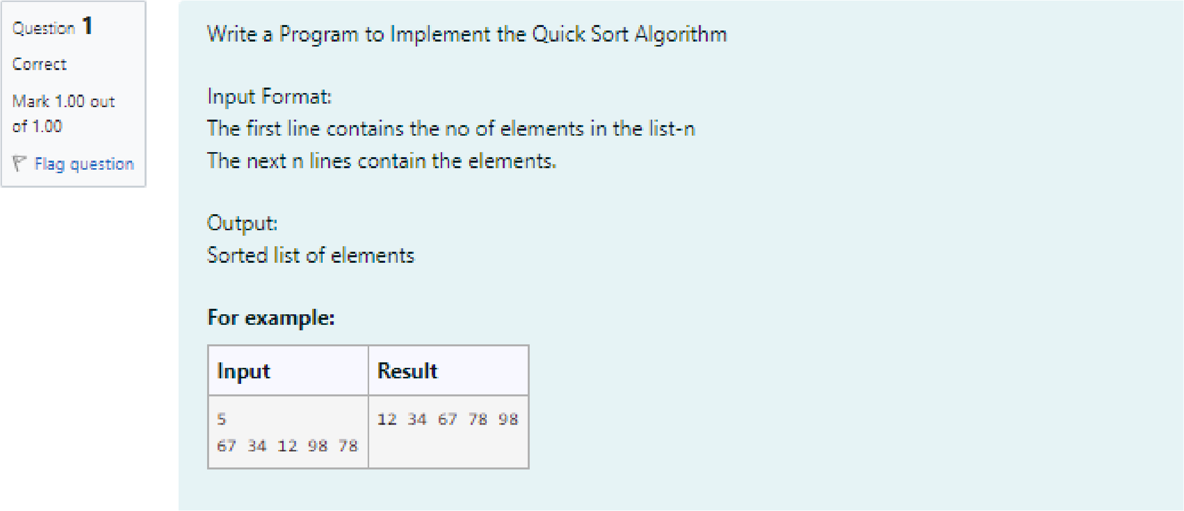
	Input	Expected	Got	
✓	4 2 4 8 10 14	4 10	4 10	✓
✓	5 2 4 6 8 10 100	No	No	✓

Passed all tests: ✓

Correct

Marks for this submission: 1.00/1.00

5. DIVIDE AND CONQUER



Question **1**

Correct

Mark 1.00 out of 1.00

Flag question

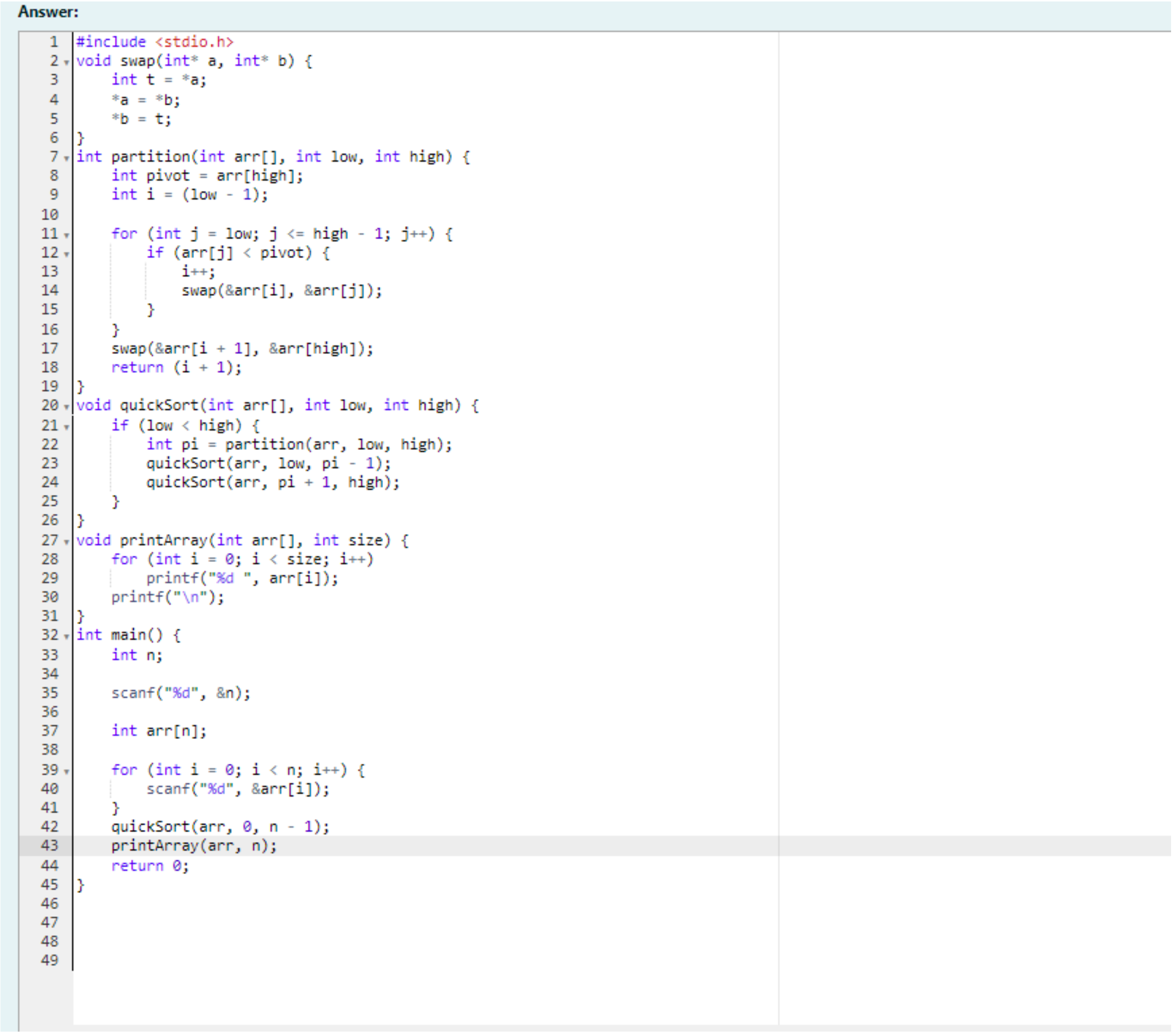
Write a Program to Implement the Quick Sort Algorithm

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 67 34 12 98 78	12 34 67 78 98



Answer:

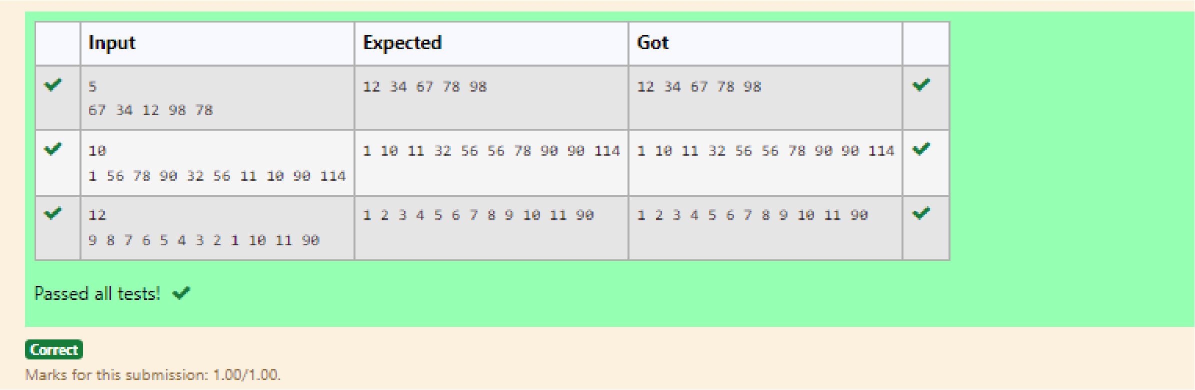
```c
#include <stdio.h>
void swap(int* a, int* b) {
    int t = *a;
    *a = *b;
    *b = t;
}
int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = (low - 1);

    for (int j = low; j <= high - 1; j++) {
        if (arr[j] < pivot) {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[i + 1], &arr[high]);
    return (i + 1);
}
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
void printArray(int arr[], int size) {
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    printf("\n");
}
int main() {
    int n;

    scanf("%d", &n);

    int arr[n];

    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    quickSort(arr, 0, n - 1);
    printArray(arr, n);
    return 0;
}
```



	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.