DIVIDE AND CONQUER PROBLEM PROGRAMS

1. **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.

Program:

#include <stdio.h>

int firstZero(int arr[],int low,int high){

if (high >= low){

int mid=low+(high-low)/2;

if((mid==0||arr[mid-1]==1)&&arr[mid]==0)return mid;

if(arr[mid]==1)return firstZero(arr,(mid+1),high);

else return firstZero(arr, low, (mid -1));}

return -1;}

int countZeroes(int arr[],int n){

int first = firstZero(arr, 0, n-1);

if (first == -1)return 0;

return(n-first);}

int main(){

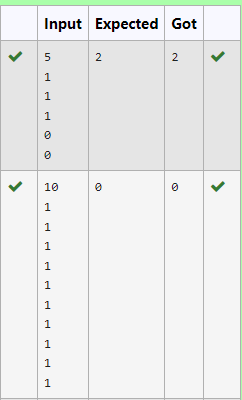
int n,i;

scanf("%d",&n);

int arr[n];

for(i=0;i<n;i++)scanf("%d",&arr[i]);

printf("%d",countZeroes(arr, n));}



2. 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

Program:

#include<stdio.h>

#include<stdlib.h>

int compare(const void\* a, const void\* b) {

return (\*(int\*)a - \*(int\*)b);}

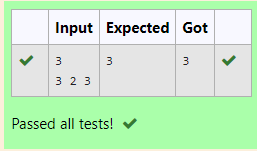
int main(){

int a;scanf("%d",&a);int arr[a];

for(int i=0;i<a;i++)scanf("%d",&arr[i]);

qsort(arr,a,sizeof(int),compare);

printf("%d",arr[a/2]);}



3. **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

Program:

#include<stdio.h>

int findFloor(int arr[],int left,int right,int x){

if(right<left){

return -1;

}

if(arr[right]<=x){

return arr[right];

}

if(arr[left]>x){

return -1;

}

int mid=(left+right)/2;

if(arr[mid]==x){

return arr[mid];

}

else if(arr[mid]<x){

int floorValue=findFloor(arr,mid+1,right,x);

return(floorValue!=-1)?floorValue:arr[mid];

}else{

return findFloor(arr,left,mid-1,x);

}

}

int main(){

int n;

scanf("%d",&n);

int arr[n];

for(int i=0;i<n;i++){

scanf("%d",&arr[i]);

}

int x;

scanf("%d",&x);

int result=findFloor(arr,0,n-1,x);

if(result==-1){

printf("%d",x);

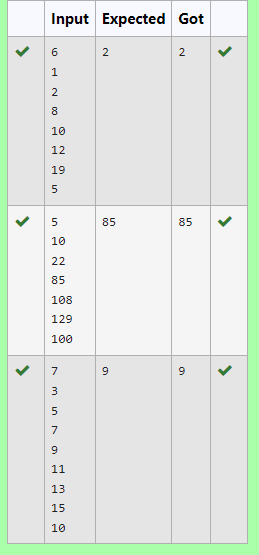
}

else{

printf("%d",result);

}

return 0;

}

4. **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”)

Program:

#include <stdio.h>

void findPair(int arr[],int i,int j,int x){

if (i>=j){

printf("No\n");

return;}

int sum=arr[i]+arr[j];

if (sum == x) {

printf("%d\n%d",arr[i],arr[j]);

return;}

else if(sum<x)findPair(arr,i+1,j,x);

else findPair(arr,i,j-1,x);}

int main(){

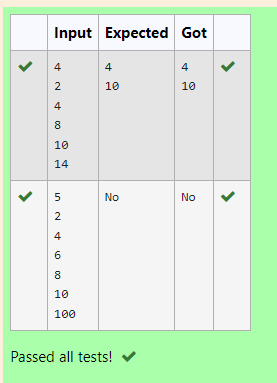
int n;scanf("%d",&n);int arr[n];

for (int i=0;i<n;i++)scanf("%d", &arr[i]);

int x;scanf("%d", &x);

findPair(arr,0,n-1,x);

return 0;}



5. 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 |

#include<stdio.h>

void quick(int a[],int left,int right){

if(left<right){

int i=left,j=right;

int pivot=a[left];

while(i<j){

while(a[j]>pivot)j--;

while(i<j&&a[i]<=pivot)i++;

if(i<j){

int temp=a[i];

a[i]=a[j];

a[j]=temp;}}

a[left]=a[j];a[j]=pivot;

quick(a,left,j-1);

quick(a,j+1,right);}}

int main(){

int a;scanf("%d",&a);int arr[a];

for(int i=0;i<a;i++)scanf("%d",&arr[i]);

quick(arr,0,a-1);

for(int i=0;i<a;i++)printf("%d ",arr[i]);

}

