DESIGN AND ANALYSIS OF ALGORITHMS

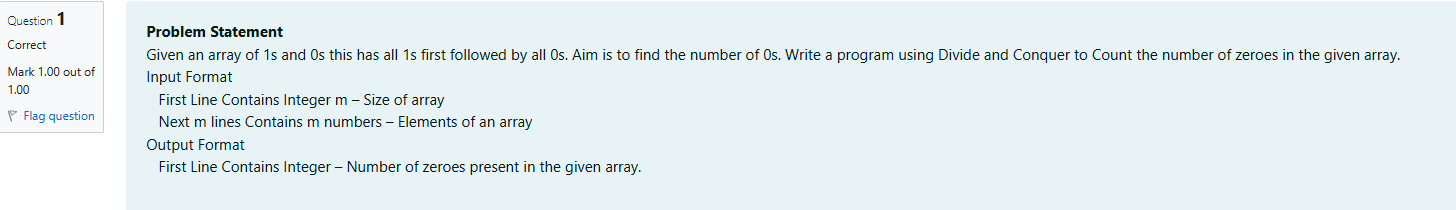
**NAME : P.R.DHANVIN**

**DEPT & SEC : CSE & ‘B’**

**ROLL NO : 230701071**

**WEEK : DIVIDE AND CONQUER**

1-NUMBER OF ZEROS IN A GIVEN ARRAY



#include <stdio.h>

int coun(int a[], int low, int high) {

if (low == high) {

return a[low] == 0 ? 1 : 0;

}

int mid = (low + high) / 2;

int leftCount = coun(a, low, mid);

int rightCount = coun(a, mid + 1, high);

return leftCount + rightCount;

}

int main() {

int n;

scanf("%d", &n);

int a[n];

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

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

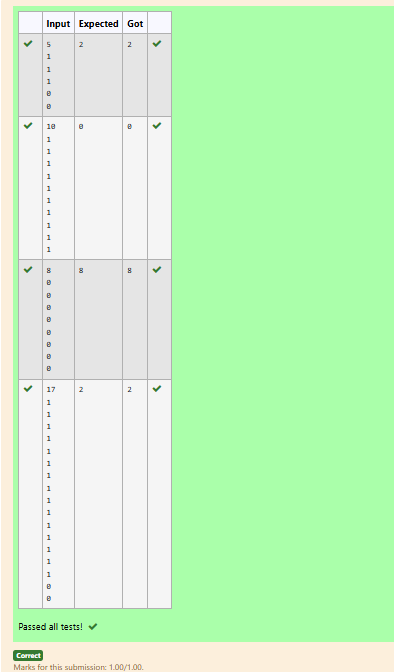
}

int result = coun(a, 0, n - 1);

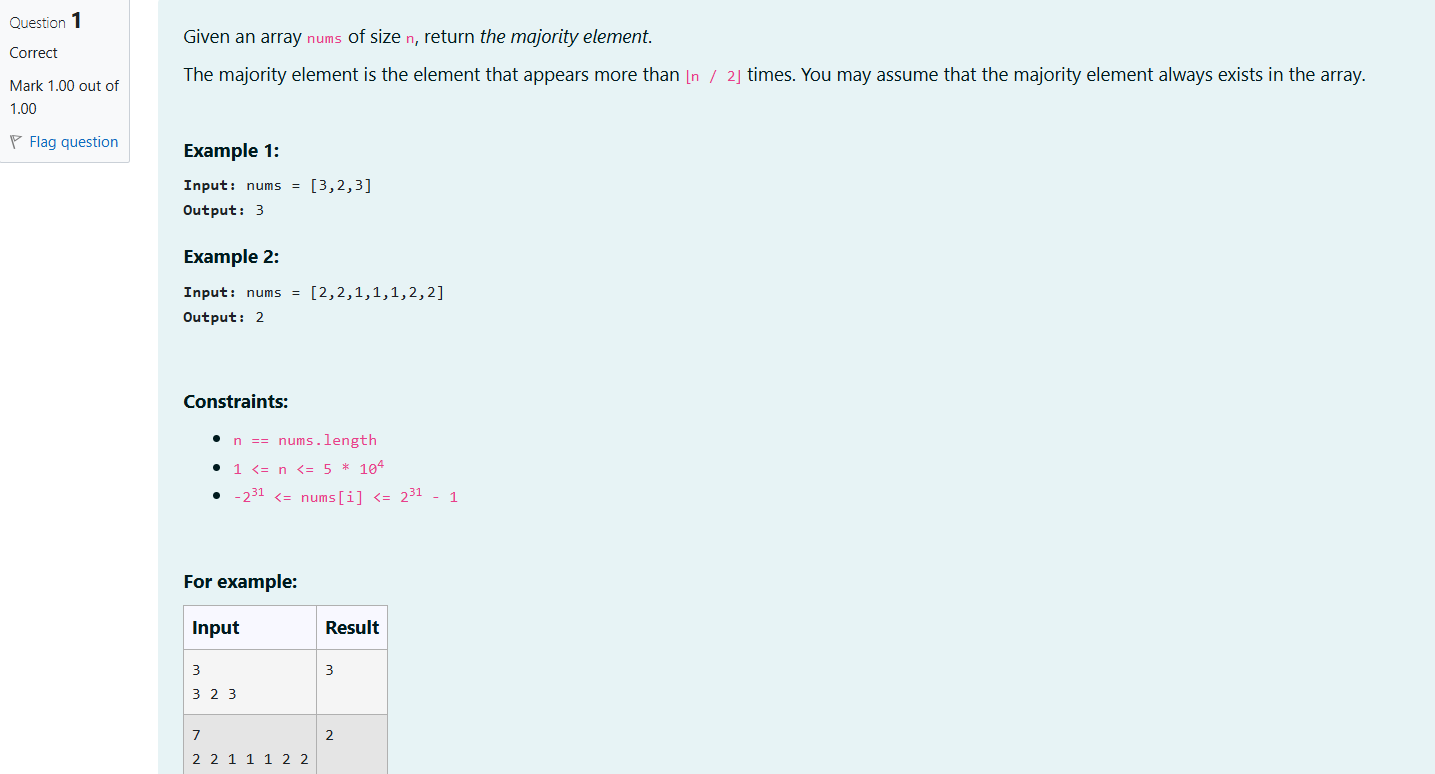
printf("%d\n", result);

return 0;

}



2-MAJORITY ELEMENT



#include <stdio.h>

#define MAX 1000

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

int freq[MAX] = {0};

int maxCount = 0;

int mostFrequent = arr[0];

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

freq[arr[i]]++;

if (freq[arr[i]] > maxCount) {

maxCount = freq[arr[i]];

mostFrequent = arr[i];

}

}

return mostFrequent;

}

int main() {

int n;

scanf("%d", &n);

int arr[n];

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

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

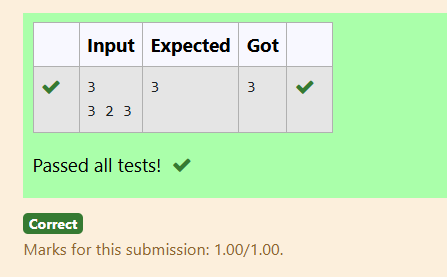
}

int result = findMostFrequent(arr, n);

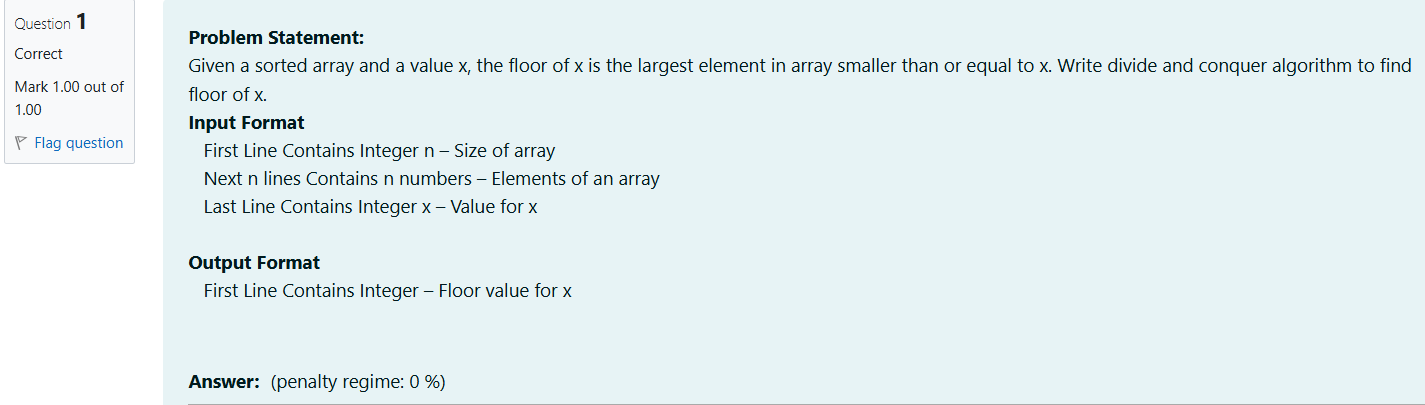
printf("%d\n", result);

return 0;

}



3-FINDING FLOOR VALUE



#include<stdio.h>

int find(int arr[],int high,int low,int x);

int main(){

int size;

scanf("%d",&size);

int arr[size];

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

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

}

int a;

scanf("%d",&a);

int mid=(0+(size-1)/2);

int left=find(arr,0,mid,a);

int right=find(arr,mid+1,size-1,a);

if(left>right)

{

printf("%d",left);

}

else

{

printf("%d",right);

}

}

int find(int arr[],int low,int high,int x){

int element=0;

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

if(arr[i]<=x){

if(arr[i]>element){

element=arr[i];

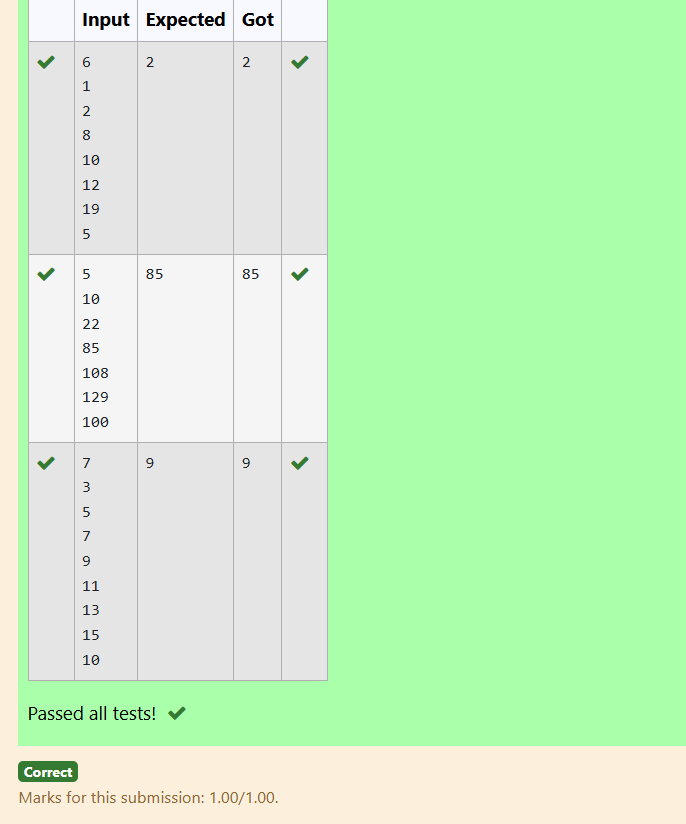
}

}

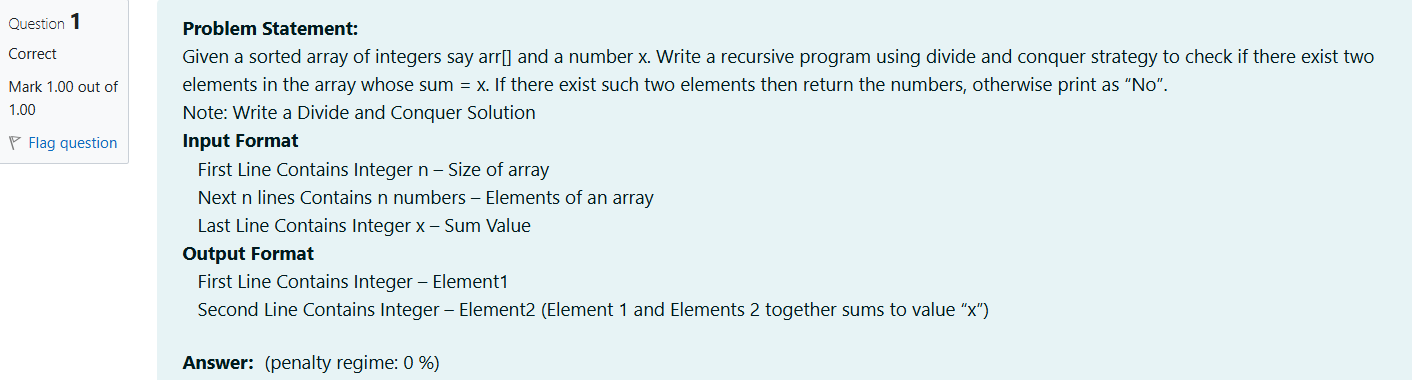
}

return element;

}



4-TWO ELEMENTS SUM TO X



#include <stdio.h>

int main()

{

int n;

scanf("%d",&n);

int arr[n];

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

{

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

}

int sum;

scanf("%d",&sum);

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

{

int m= arr[i]+arr[i+n];

if(m==sum)

{

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

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

break;

}

if (i==n-1 )

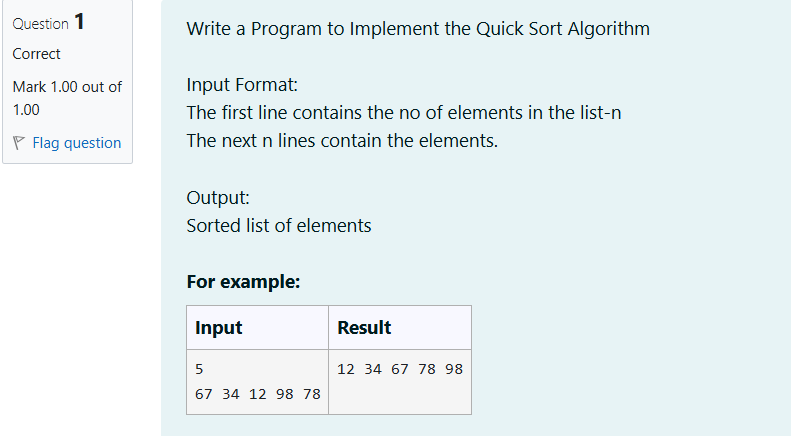
printf("No");

}

}



5-IMPLEMENTATION OF QUICK SORT



#include <stdio.h>

void quick\_sort(int arr[], int low, int high) {

if (low < high) {

int pivot = arr[low];

int i = low, j = high;

while (i < j) {

while (arr[i] <= pivot && i < high) i++;

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

if (i < j) {

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

arr[low] = arr[j];

arr[j] = pivot;

quick\_sort(arr, low, j - 1);

quick\_sort(arr, j + 1, high);

}

}

int main() {

int n;

scanf("%d", &n);

int arr[n];

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

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

}

quick\_sort(arr, 0, n - 1);

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

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

}

return 0;

}

