B.NIKITHA
230701211
CSE -'D'
III SEM

Divide and Conquer

1-Number of Zeros in a Given Array
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 %)

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

}

Feedback

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-Majority Element

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 * 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 %)

```
#include<stdio.h>
int main(){
    int n;
    int flag=0;
    int count;
    scanf("%d",&n);
    int a[n];
    for(int i=0;i<n;i++){
        scanf("%d",&a[i]);
    }
    for(int i=0;i<n-1;i++){
        count=0;
        for(int j=i;j<n;j++){
            if(a[i]==a[j])
            count++;
            if(count>n/2)
            {
                printf("%d",a[i]);
                flag=1;
              }

        }
```

```
        if(flag==1)
        break;
    }
}
```

Feedback

Input	Expected	Got
3 3 2 3	3	3

Passed all tests!

Correct

Marks for this submission: 1.00/1.00.

3-Finding Floor Value

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 %)

```
#include<stdio.h>
int largest(int a[],int i,int j,int x){
    int m=(i+j)/2;
    if(a[m]<x)
    return a[m];
    else if(a[m]>x)
    return largest(a,i,m-1,x);
    else return 0;

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

}
```

Feedback

Input	Expected	Got
6 1 2 8	2	2

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.

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 %)

```
#include<stdio.h>
void divide(int arr[],int low,int high,int s){
    int m;
    m=arr[low]+arr[high];
    if(m==s)
    {
        printf("%d\n%d",arr[low],arr[high]);}
    else if(low==high)
    printf("No");
    else if(m<s)
    divide(arr,low+1,high,s);
    else if(m>s)
    divide(arr,low,high-1,s);
```

```
}
int main(){
    int n;
    scanf("%d",&n);
    int a[n];
    for(int i=0;i<n;i++)
    scanf("%d",&a[i]);
    int s;
    scanf("%d",&s);
    divide(a,0,n-1,s);

}
```

Feedback

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-Implementation of Quick Sort

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:

```
#include <stdio.h>
void swap(int* a, int* b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

```
int partition(int arr[], int low, int high) {
    int p = arr[low];
    int i = low;
    int j = high;
    while (i < j) {
        while (arr[i] <= p && i <= high - 1) {
            i++;
        }
        while (arr[j] > p && j >= low + 1) {
            j--;
        }
        if (i < j) {
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[low], &arr[j]);
    return j;
}
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);
    }
}
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);
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    return 0;
}
```

Feedback

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.