**DIVIDE AND CONQUER**

**AIM:**

## 1-Number of Zeros in a Given Array

**ALGORITHM:**

* Read size of array (x).
* Create array (arr) of size x.
* Fill array with user input.
* Calculate halfway point (y = x / 2).
* Initialize zero counter (d = 0).
* Count zeros in both halves of the array.
* Print total count of zeros.

**PROBLEM:**

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 main()

{

int x,y,d=0;

scanf("%d",&x);

int arr[x];

y=x/2;

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

{

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

}

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

{

if(arr[i]==0)

{

d+=1;

}

}

for(int j=y;j<x;j++)

{

if(arr[j]==0)

{

d+=1;

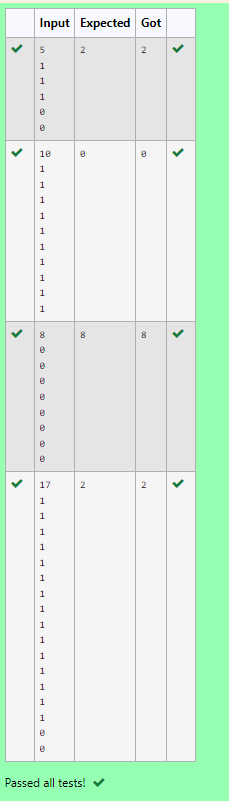
}

}

printf("%d",d);

}

**OUTPUT:**

****

**AIM:**

2-Majority Elements.

**ALGORITHM:**

1. Read integer a (size of the array).
2. Create array arr of size a.
3. Fill Array: Fill arr with user input.
4. Set candidate b = arr.
5. Count Occurrences
6. Check Majority

**PROBLEM:**

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 |

**PROGRAM:**

#include <stdio.h>

int c=0;

int C(int arr[],int x,int y,int k)

{

int m=x+(y-x)/2;

if (arr[m]==k)

c++;

else

{

C(arr,x,m,k);

C(arr,m+1,y,k);

}

return c;

}

int main()

{

int a;

scanf("%d",&a);

int arr[a];

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

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

int b=arr[0];

if (C(arr,0,a,b)>a/2)

printf("%d",b);

else

{

for (int i=0;i<a/2;i++)

if (arr[i]!=b)

{

printf("%d",b);

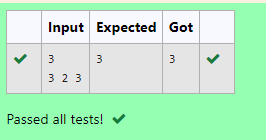
break;

}

}

}

**OUTPUT:**

****

**AIM:**

3-Finding Floor Value.

**ALGORITHM:**

1. Read integer a (size of the array).
2. Create array arr of size a.
3. Fill arr with user input.
4. Read integer b.
5. Define function f(arr, a, b):
6. Set V = f(arr, a, b).
7. Print the value of V.

**PROBLEM:**

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 f(int arr[],int a,int b)

{

int l=0, h=a-1;

int fv=-1;

while (l<=h)

{

int m=(l+h)/2;

if (arr[m]<=b)

{

fv=arr[m];

l=m+1;

}

else

{

h=m-1;

}

}

return fv;

}

int main()

{

int a,b;

scanf("%d",&a);

int arr[a];

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

{

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

}

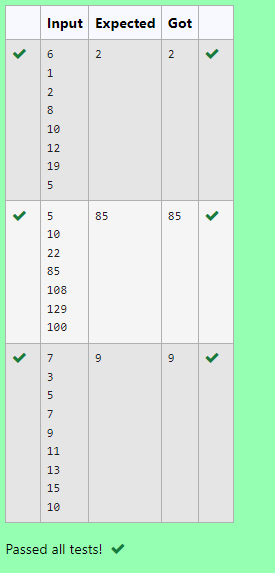
scanf("%d",&b);

int V=f(arr,a,b);

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

}

**OUTPUT:**

****

**AIM:**

4-Two Elements sum to x.

**ALGORITHM:**

1. Read integer n (size of the array).
2. Create array arr of size n.
3. Fill arr with user input.
4. Read integer sum.
5. Loop through each index i from 0 to n-1:
   * Calculate m = arr[i] + arr[i + n].
   * If m == sum:Print
   * If i == n - 1, print "No".

**PROBLEM:**

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>

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;

}

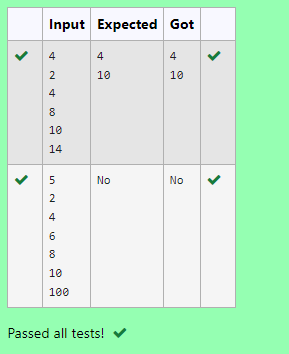
if (i==n-1 )

printf("No");

}

}

**OUTPUT:**

****

**AIM:**

5-Implementation of Quick Sort.

**ALGORITHM:**

1. Read integer n (size of the array).
2. Create array arr of size n.
3. Fill arr with user input.
4. Define function sort(arr, lt, rt)
5. Call sort(arr, 0, n - 1).
6. Print sorted array elements from arr.

**PROBLEM:**

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 |

**PROGRAM:**

#include<stdio.h>

void sort(int arr[],int lt,int rt)

{

if(lt<rt)

{

int p=(lt+rt)/2;

int i=lt;

int j=rt;

while(i<j)

{

while(arr[p]>=arr[i] )

{

i++;

}

while(arr[p]<arr[j] )

{

j--;

}

if(i<=j)

{

int temp=arr[i];

arr[i]=arr[j];

arr[j]=temp;

}

}

int temp=arr[j];

arr[j]=arr[p];

arr[p]=temp;

sort(arr,lt+1,rt);

}

}

int main()

{

int n;

scanf("%d",&n);

int arr[n];

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

{

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

}

sort(arr,0,n-1);

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

{

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

}

}

**OUTPUT:**

****