GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- Home
- Algorithms
- DS
- GATE
- Interview Corner
- Q&A
- C
- C++
- Java
- Books
- Contribute
- Ask a Q
- About

Array
Bit Magic
C/C++
Articles
GFacts
Linked List
MCQ
Misc
Output
String
Tree
Graph

Find the number of zeroes

Given an array of 1s and 0s which has all 1s first followed by all 0s. Find the number of 0s. Count the number of zeroes in the given array.

Examples:

```
Input: arr[] = {1, 1, 1, 1, 0, 0}
Output: 2

Input: arr[] = {1, 0, 0, 0, 0}
Output: 4

Input: arr[] = {0, 0, 0}
Output: 3
```

```
Input: arr[] = {1, 1, 1, 1}
Output: 0
```

We strongly recommend to minimize the browser and try this yourself in time complexity better than O(n).

A **simple solution** is to traverse the input array. As soon as we find a 0, we return n – index of first 0. Here n is number of elements in input array. Time complexity of this solution would be O(n).

Since the input array is sorted, we can use **Binary Search** to find the first occurrence of 0. Once we have index of first element, we can return count as n – index of first zero.

```
// A divide and conquer solution to find count of zeroes in an array
// where all 1s are present before all 0s
#include <stdio.h>

/* if 0 is present in arr[] then returns the index of FIRST occurrence
   of 0 in arr[low..high], otherwise returns -1 */
int firstZero(int arr[], int low, int high)
{
    if (high >= low)
    {
        // Check if mid element is first 0
        int mid = low + (high - low)/2;
        if (( mid == 0 || arr[mid-1] == 1) && arr[mid] == 0)
            return mid;

        if (arr[mid] == 1)  // If mid element is not 0
            return firstZero(arr, (mid + 1), high);
        else  // If mid element is 0, but not first 0
            return firstZero(arr, low, (mid -1));
    }
    return -1;
}

// A wrapper over recursive function firstZero()
int countOnes(int arr[], int n)
{
    // Find index of first zero in given array
    int first = firstZero(arr, 0, n-1);

    // If 0 is not present at all, return 0
    if (first == -1)
        return 0;

    return (n - first);
}

/* Driver program to check above functions */
int main()
{
    int arr[] =   {1, 1, 1, 0, 0, 0, 0, 0};
    int n = sizeof(arr)/sizeof(arr[0]);
```

```
    printf("Count of zeroes is %d", countOnes(arr, n));
    return 0;
}
```

Output:

```
Count of zeroes is 5
```

Time Complexity: O(Logn) where n is number of elements in arr[].

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Related Topics:

- Find Union and Intersection of two unsorted arrays
- Pythagorean Triplet in an array
- Maximum profit by buying and selling a share at most twice
- Design a data structure that supports insert, delete, search and getRandom in constant time
- Print missing elements that lie in range 0 – 99
- Iterative Merge Sort
- Group multiple occurrence of array elements ordered by first occurrence
- Given a sorted and rotated array, find if there is a pair with a given sum

Tags: Divide and Conquer

Writing code in comment? Please use **ideone.com** and share the link here.

low+high might cause overflow.

2 ^ | v • Reply • Share ›

surbhijain93 ➚ Aditya Goel • 3 months ago

Can you pls elaborate.

^ | v • Reply • Share ›

Aditya Goel ➚ surbhijain93 • 3 months ago

What's there to elaborate? Read about integer overflow, you will get an idea.

Also refer G2G post on this - http://www.geeksforgeeks.org/p...

^ | v • Reply • Share ›

nishant gupta • 3 months ago

Alternative :
Calculate the index of last '1' in the array :
http://ideone.com/dlB1Fu

Let me now your thoughts

1 ^ | v • Reply • Share ›

Guest • 3 months ago

Alternative :
Calculate the iindex of last 1 in the array :
http://ideone.com/dlB1Fu

Let me now your thoughts

^ | v • Reply • Share ›

lucy • 3 months ago

@geeksforgeeks we can also solve it by hash function in o(1) if i am not wrong?

^ | v • Reply • Share ›

Krishna ➚ lucy • 3 months ago

I think u can't do this with hash function. For hash u need to traverse whole array that is 0(n).

1 ^ | v • Reply • Share ›

lucy ➚ Krishna • 3 months ago

yes. thanks

^ | v • Reply • Share ›

Aditya Goel • 4 months ago

```
int countOnes(int arr[], int n)
{
        if(arr[n-1]==1)
                return 0;
        if(arr[0]==0)
                return n;
    return countOnes(arr, n/2)+countOnes(arr+n/2, n-n/2);
}
```

• Reply • Share ›

gamer • 4 months ago

int mid = low + (high - low)/2;. What is the added advantage of

this step instead of simply putting. int mid = (low + high)/2;

This might sound lame but I was curious.

2 • Reply • Share ›

vaishnavi • 8 months ago

count the string length (as l). count the number of 1's(i.e upto the character '0' as c).then substract l-c we get number of zeros.

• Reply • Share ›

Siya ➚ vaishnavi • 5 months ago

It is O(n) and we want solution better than this.

1 • Reply • Share ›

vaishnavi • 8 months ago

for(i=0;i<l;i++) {="" if(s[i]!="0")="" c="c+1;" if(s[i]="='0')" break;="" }="" printf("%d",l-c);="">

• Reply • Share ›

vaishnavi • 8 months ago

#include<conio.h>

#include<stdio.h>

void main()

{

char s[100];

int i,l,c=0;

clrscr();

clrscr();

printf("enter any binary number which has all zero's followed by all 1's");

scanf("%s",s);

l=strlen(s);

for(i=0;i<l;i++) {="" if(s[i]!="0")="" c="c+1;" if(s[i]="='0')" break;="" }="" printf("%d",l-c);="" getch();="" }="">

• Reply • Share ›

Pankaj Kumar • 8 months ago

#include<iostream>

using namespace std;

int main()

{

int i,count1=0;

int a[]= {1, 1, 1, 0, 0, 0, 0, 0};

int n = sizeof(a)/sizeof(a[0]);

for(int i=0;i<n;i++) {="" int="" ch="a[i];" switch(ch)="" {="" case="" 0:count1++;="" break;="" case="" 1:="" break;="" }="" }="" cout<<count1;="" return="" 0;="" }="">

2 • Reply • Share ›

Jun • 9 months ago

http://ideone.com/1ReSJX

1 • Reply • Share ›

instance • 10 months ago

```
int getPoint(int arr[], int start, int end)
{
int mid = (start + end)/2;
if(start >= end)
{
if(arr[start] == 1)
return start;
return -1;
}
if((arr[mid] == 1) && (arr[mid+1] == 0))
return mid+1;
```

```
else if((arr[mid] == 0) && (arr[mid-1] == 1))
return mid;
else if(arr[mid] == 1)
return getPoint(arr, mid+1, end);
else
return getPoint(arr, start, mid);
}
```

Reply • Share ›

awallace • 10 months ago

http://ideone.com/dBI8CG

several solutions in Java (Iterative not Recursive)

Reply • Share ›

Guest • a year ago

Simply if u use binary search.... it will easy than it

Reply • Share ›

Preethi • a year ago

We just search from back side of an array till find 1.
No of 0's=(index of first 1-n)
It Complexity also O(n).

1 Reply • Share ›

GOPI GOPINATH ➦ Preethi • a year ago

looks good, but complexity O(n) is not a better answer preethi , because we can still reduce it to O(log n) using Binary search

Reply • Share ›

Preethi ➦ GOPI GOPINATH • a year ago

Ya... thanx...

Reply • Share ›

arjomanD • a year ago

As simple as locating the middle of the array :D

Reply • Share ›

karthi • a year ago

import java.io.*;

class NumberOfZeros{

public static int getNumberOfZeros(int[] a){

```
if(a.length == 0){

return 0;

}

int count = 0;

for(int i=a.length-1;i!=-1&&a[i]!=1;i--){

count++;

}

return count;
```

see more

• Reply • Share ›

Chandu Mannam • a year ago

```
#include<stdio.h>

int getposition(int,int,int*);

int main()

{

int a[ ]={1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

int n,k;

n = sizeof(a)/sizeof(a[0]);

k = getposition(0,n,a);

printf("no of zeros=%d",n-k);

return 0;

}
```

see more

• Reply • Share ›

napender singh • a year ago

```
public class FindZero{

public static void main(String []args){
```

```
int[] arr = {1,1,1,0,0,0,1,0,1};

int one = findSumOfArr(arr);

int two = findLengthOfArr(arr);

int zero = two - one;

System.out.println(zero);

}

public static int findLengthOfArr(int[] a){

int j = 0;

for(int i=0;i<a.length;i++) j++;="" return="" j;="" }="" public="" static="" int="" findsumofarr(int[]="" a){="" int="" temp="0;" for(int="" i="0;i&amp;lt;a.length;i++){" temp="temp" +="" a[i];="" }="" return="" temp;="" }="" }="">
```

• Reply • Share ›

:/ • a year ago

In an interview I was asked this question with the additional constraint that the size of the array is not given. (Determining N, size, will take usually O(N)) Any ideas on how to approach this in O(log N)?

• Reply • Share ›

GOPI GOPINATH :/ • a year ago

If its an array.....sizeof(arr)/sizeof(arr[0]) will give the length of the array.
I dnt think interviewer will accept this. :P

• Reply • Share ›

GeeksforGeeks Mod :/ • a year ago

You can use the idea discussed in the below post.

http://www.geeksforgeeks.org/f...

2 • Reply • Share ›

laksbv :/ • a year ago

http://stackoverflow.com/quest...

• Reply • Share ›

coder • a year ago

i think this code will not work for an array which contains only zero. Need to add below check:

```
if(arr[low] == 0){
```

return low;
}
2 • Reply • Share ›

Kartik ➚ coder • a year ago
Please take a closer look. It works for all 0s also. Sample run http://ideone.com/nivtz3
• Reply • Share ›

Amit • a year ago

```
#include<stdio.h>

int main()

{

int findzeros(int arr[],int n);

int arr[9]={1,0,0,1,0,0,0,0,1};

printf("Number of zeros: %d",findzeros(arr,9));

return 0;

}

int findzeros(int arr[],int n)

{

if(n>=1)
```

see more

• Reply • Share ›

Siva Krishna • a year ago
Another possible way is ..first you check in the following way
1 2 4 8 16 32 ...2^k
stop when arr[i] == 0 and then do binary search on sub array i/2 to i for finding the exact position. But this has a worst case complexity of O(n) and i think average case complexity is less.
• Reply • Share ›

Kartik ➚ Siva Krishna • a year ago
Siva Krishna, Thanks for suggesting this. This solution can be combined with Binary Search to guarantee O(Logn). See http://www.geeksforgeeks.org/f...
It can in fact be a good solution if we don't know size of array.
• Reply • Share ›

zzer ➚ Siva Krishna • a year ago

it may overflow the boundary of the input array

▲ | ▼ • Reply • Share ›

Siva Krishna ➚ zzer • a year ago

that's an idea..not the exact implementation. You have to do some boundary checks while implementing that.

▲ | ▼ • Reply • Share ›

zzer ➚ Siva Krishna • a year ago

well, it's really a good idea, this idea has been used in some other problems, could u list some of them?

▲ | ▼ • Reply • Share ›

Siva Krishna ➚ zzer • a year ago

1.Find the point where a monotonically increasing function becomes positive first time.
2. Find the starting element in an rotated sorted array.

▲ | ▼ • Reply • Share ›

Ânon • a year ago

For thé simple solution, you return n - index + 1

▲ | ▼ • Reply • Share ›

AA ➚ Ânon • a year ago

SADSADSAD

▲ | ▼ • Reply • Share ›

- Interview Experiences
- Advanced Data Structures
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Mathematical Algorithms
- Recursion
- Geometric Algorithms

Popular Posts

- All permutations of a given string
- Memory Layout of C Programs
- Understanding “extern” keyword in C
- Median of two sorted arrays
- Tree traversal without recursion and without stack!
- Structure Member Alignment, Padding and Data Packing
- Intersection point of two Linked Lists
- Lowest Common Ancestor in a BST.
- Check if a binary tree is BST or not
- Sorted Linked List to Balanced BST

Recent Comments

- Nikhil kumar

  public class...

  Print missing elements that lie in range 0 – 99 · 5 minutes ago

- Ashish Aggarwal

  Try Data Structures and Algorithms Made Easy -...

  Algorithms · 27 minutes ago

- Vlad

  Thanks. Very interesting lectures.

  Expected Number of Trials until Success · 1 hour ago

- cfh

  My implementation which prints the index of the...

  Longest Even Length Substring such that Sum of First and Second Half is same · 1 hour ago

- Gaurav pruthi

  forgot to see that part ;)

  Bloomberg Interview | Set 1 (Phone Interview) · 2 hours ago

- saeid aslami

  thanks

  Greedy Algorithms | Set 7 (Dijkstra's shortest path algorithm) · 2 hours ago