Count smaller elements on right side

Write a function to count number of smaller elements on right of each element in an array. Given an unsorted array arr[] of distinct integers, construct another array countSmaller[] such that countSmaller[i] contains count of smaller elements on right side of each element arr[i] in array.

Examples:

```
Input:   arr[] =  {12, 1, 2, 3, 0, 11, 4}
Output:  countSmaller[]  =  {6, 1, 1, 1, 0, 1, 0}

(Corner Cases)
Input:   arr[] =  {5, 4, 3, 2, 1}
Output:  countSmaller[]  =  {4, 3, 2, 1, 0}

Input:   arr[] =  {1, 2, 3, 4, 5}
Output:  countSmaller[]  =  {0, 0, 0, 0, 0}
```

Method 1 (Simple)
Use two loops. The outer loop picks all elements from left to right. The inner loop iterates through all the elements on right side of the picked element and updates countSmaller[].

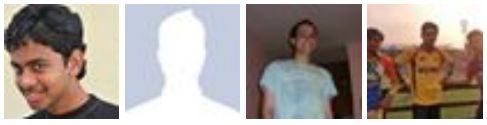
```
void constructLowerArray (int *arr[], int *countSmaller, int n)
{
  int i, j;

  // initialize all the counts in countSmaller array as 0
  for   (i = 0; i < n; i++)
     countSmaller[i] = 0;
```

Google™ Custom Search

GeeksforGeeks

53,521 people like GeeksforGeeks.



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

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
  for (i = 0; i < n; i++)
  {
    for (j = i+1; j < n; j++)
    {
       if (arr[j] < arr[i])
         countSmaller[i]++;
    }
  }
}

/* Utility function that prints out an array on a line */
void printArray(int arr[], int size)
{
  int i;
  for (i=0; i < size; i++)
    printf("%d ", arr[i]);

  printf("\n");
}

// Driver program to test above functions
int main()
{
  int arr[] = {12, 10, 5, 4, 2, 20, 6, 1, 0, 2};
  int n = sizeof(arr)/sizeof(arr[0]);
  int *low = (int *)malloc(sizeof(int)*n);
  constructLowerArray(arr, low, n);
  printArray(low, n);
  return 0;
}
```

Time Complexity: O(n^2)
Auxiliary Space: O(1)

Method 2 (Use Self Balancing BST)
A Self Balancing Binary Search Tree (AVL, Red Black,.. etc) can be used to get the solution in O(nLogn) time complexity. We can augment these trees so that every node N contains size the subtree rooted with N. We have used AVL tree in the following implementation.

We traverse the array from right to left and insert all elements one by one in an AVL tree. While inserting a new key in an AVL tree, we first compare the key with root. If key is greater than root, then it is greater than all the nodes in left subtree of root. So we add the size of left subtree to the count of smaller element for the key being inserted. We recursively follow the same approach for

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

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

all nodes down the root.

Following is C implementation.

```
#include<stdio.h>
#include<stdlib.h>

// An AVL tree node
struct node
{
    int key;
    struct node *left;
    struct node *right;
    int height;
    int size; // size of the tree rooted with this node
};

// A utility function to get maximum of two integers
int max(int a, int b);

// A utility function to get height of the tree rooted with N
int height(struct node *N)
{
    if (N == NULL)
        return 0;
    return N->height;
}

// A utility function to size of the tree of rooted with N
int size(struct node *N)
{
    if (N == NULL)
        return 0;
    return N->size;
}

// A utility function to get maximum of two integers
int max(int a, int b)
{
    return (a > b)? a : b;
}

/* Helper function that allocates a new node with the given key and
    NULL left and right pointers. */
struct node* newNode(int key)
{
```

Deploy Early.
Deploy Often.

DevOps from Rackspace:

Automation

FIND OUT HOW ▸

rackspace
the open cloud company

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    struct node* node = (struct node*)
                        malloc(sizeof(struct node));
    node->key   = key;
    node->left   = NULL;
    node->right  = NULL;
    node->height = 1;  // new node is initially added at leaf
    node->size = 1;
    return(node);
}

// A utility function to right rotate subtree rooted with y
struct node *rightRotate(struct node *y)
{
    struct node *x = y->left;
    struct node *T2 = x->right;

    // Perform rotation
    x->right = y;
    y->left = T2;

    // Update heights
    y->height = max(height(y->left), height(y->right))+1;
    x->height = max(height(x->left), height(x->right))+1;

    // Update sizes
    y->size = size(y->left) + size(y->right) + 1;
    x->size = size(x->left) + size(x->right) + 1;

    // Return new root
    return x;
}

// A utility function to left rotate subtree rooted with x
struct node *leftRotate(struct node *x)
{
    struct node *y = x->right;
    struct node *T2 = y->left;

    // Perform rotation
    y->left = x;
    x->right = T2;

    //  Update heights
    x->height = max(height(x->left), height(x->right))+1;
    y->height = max(height(y->left), height(y->right))+1;

    // Update sizes
```

705 Subscribe

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 15 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 19 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 44 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 45 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    x->size = size(x->left) + size(x->right) + 1;
    y->size = size(y->left) + size(y->right) + 1;

    // Return new root
    return y;
}

// Get Balance factor of node N
int getBalance(struct node *N)
{
    if (N == NULL)
        return 0;
    return height(N->left) - height(N->right);
}

// Inserts a new key to the tree rotted with node. Also, updates *coun
// to contain count of smaller elements for the new key
struct node* insert(struct node* node, int key, int *count)
{
    /* 1.  Perform the normal BST rotation */
    if (node == NULL)
        return(newNode(key));

    if (key < node->key)
        node->left  = insert(node->left, key, count);
    else
    {
        node->right = insert(node->right, key, count);

        // UPDATE COUNT OF SMALLER ELEMENTS FOR KEY
        *count = *count + size(node->left) + 1;
    }


    /* 2. Update height and size of this ancestor node */
    node->height = max(height(node->left), height(node->right)) + 1;
    node->size   = size(node->left) + size(node->right) + 1;

    /* 3. Get the balance factor of this ancestor node to check whethe
       this node became unbalanced */
    int balance = getBalance(node);

    // If this node becomes unbalanced, then there are 4 cases

    // Left Left Case
    if (balance > 1 && key < node->left->key)
        return rightRotate(node);
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
    // Right Right Case
    if (balance < -1 && key > node->right->key)
        return leftRotate(node);

    // Left Right Case
    if (balance > 1 && key > node->left->key)
    {
        node->left =  leftRotate(node->left);
        return rightRotate(node);
    }

    // Right Left Case
    if (balance < -1 && key < node->right->key)
    {
        node->right = rightRotate(node->right);
        return leftRotate(node);
    }

    /* return the (unchanged) node pointer */
    return node;
}

// The following function updates the countSmaller array to contain co
// smaller elements on right side.
void constructLowerArray (int arr[], int countSmaller[], int n)
{
  int i, j;
  struct node *root = NULL;

  // initialize all the counts in countSmaller array as 0
  for  (i = 0; i < n; i++)
     countSmaller[i] = 0;

  // Starting from rightmost element, insert all elements one by one i
  // an AVL tree and get the count of smaller elements
  for (i = n-1; i >= 0; i--)
  {
     root = insert(root, arr[i], &countSmaller[i]);
  }
}

/* Utility function that prints out an array on a line */
void printArray(int arr[], int size)
{
  int i;
  printf("\n");
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
  for (i=0; i < size; i++)
    printf("%d ", arr[i]);
}

// Driver program to test above functions
int main()
{
  int arr[] = {10, 6, 15, 20, 30, 5, 7};
  int n = sizeof(arr)/sizeof(arr[0]);

  int *low = (int *)malloc(sizeof(int)*n);

  constructLowerArray(arr, low, n);

  printf("Following is the constructed smaller count array");
  printArray(low, n);
  return 0;
}
```

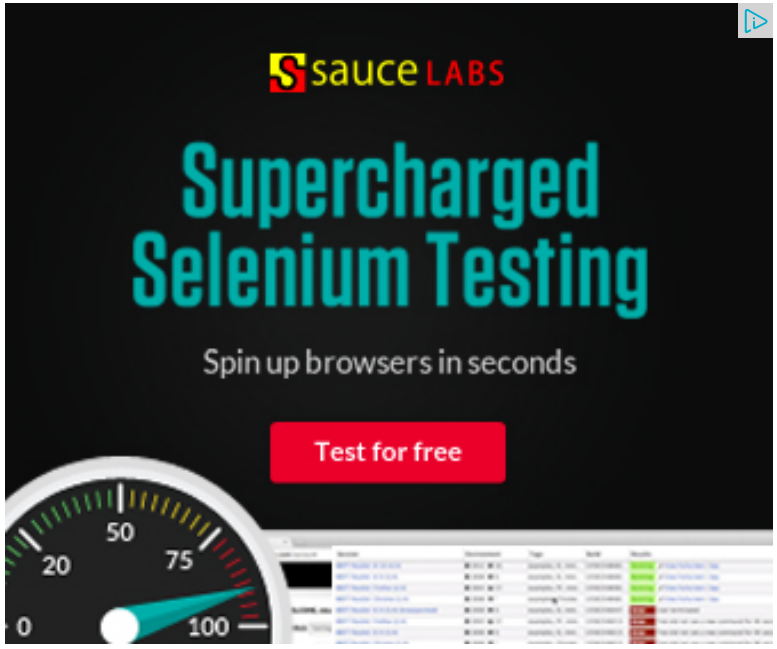
Output:

```
Following is the constructed smaller count array
3 1 2 2 2 0 0
```

Time Complexity: O(nLogn)
Auxiliary Space: O(n)

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

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com



Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

0 Tweet 0 0

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

42 Comments GeeksforGeeks

Sort by Newest

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Guest • 3 months ago

This is same as "Count Inversions in an array"

http://www.geeksforgeeks.org/c...

2 ▲ | ▼ • Reply • Share ›

Guest • 4 months ago

A simple O(nlogn) algorithm that works with repeated integers also but the dra
will be modified.

```
countSmaller[n-1] = 0;

for( int i = n -2; i >= 0 ;i--)
{
/* This return the index of the element just smaller than this element arr[i]
int position = find(arr, i+1, n-1);

countSmaller[i] = position - i;

// Insert the arr[i] element to its proper position so that i to n-1 is sorted
insert(arr, i, n -1);
}
```

1 ▲ | ▼ • Reply • Share ›

Guest ➦ Guest • 2 months ago

The insert method takes quadratic time overall. It's not O(nlogn)

▲ | ▼ • Reply • Share ›

Prakhar Jain • 9 months ago

Done in O(nlogn) time. Works for repeated integers. Used BIT.

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
//

//  countSmallerRight.cpp

//  geeksforgeeks

//

//  Created by Prakhar Jain on 25/08/13.

//  Copyright (c) 2013 Prakhar Jain. All rights reserved.

//

#include <iostream>
```

see more

1 ▲ | ▼ • Reply • Share ›

abhishek08aug • a year ago

Intelligent :D

▲ | ▼ • Reply • Share ›

Marsha Donna abhishek08aug • 7 months ago

pls help,...i m not able to copy paste c code from vc++ as comment here..after it gets uploaded as comment it is not displayed correctly

▲ | ▼ • Reply • Share ›

abhishek08aug • a year ago

You say: Given an unsorted array arr[] of "distinct" integers

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

and then you put two 1s in the input. :/

Input: arr[] = {12, 1, 2, 3, 0, 11, 1}

Correct it please.

```
/* Paste your code here (You may delete these lines if not writing c
```

Reply • Share ›

GeeksforGeeks abhishek08aug • a year ago

Thanks for pointing this out. We have corrected the example.

Reply • Share ›

Hanish • a year ago

There is a typo in Method 2 - 1st argument of this function.
void constructLowerArray (int *arr[], int countSmaller[], int n)

It should be int * arr . Please correct this

Reply • Share ›

GeeksforGeeks Hanish • a year ago

@Hanish: Thanks for pointing this out. We have fixed the typo error. K

Reply • Share ›

nitin gupta iitian • 2 years ago

//Count smaller elements on right side
/*

this logic use either array or stack
we use array .
Logic:
we travers the main array from right to left and keep inserting the current elem

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

always a sorted one
ex.
index 0 1 2 3 4 5
main array 12 1 3 0 11 1
for this from right to left
i = 5 ele= 1 since its right most so put zero in that index of Ans. array and pusl

index 0 1 2 3 4 5
main array 12 1 3 0 11 1
temp array 1
Ans. array 0

---

see more

• Reply • Share ›

nitin gupta iitian nitin gupta iitian • 2 years ago

sorry temp array operation in worst case is not less then O(n) its O(n)
here is my code....
don't mind its big ...but cover all cases and make temp_array operatior
worst case

```
 /* //Count smaller elements on right side
/*

#include <iostream>
#include <cstdlib>
using namespace std ;

//This function checks for Sorted Sequence if its find this sec
short checkForSorted (int *A , size_t size, int *Ans_array )
{
    int i =0,Ans_index=0 ;
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
short Sorted_order = -1 ; // -1 for in case all element ar
//checking for
```

see more

• Reply • Share ›

nitin gupta iitian ➚ nitin gupta iitian • 2 years ago

[sourcecode language="C++"]
[sourcecode]

```
/* #include <iostream>
#include <cstdlib>
using namespace std ;

//This function checks for Sorted Sequence if its find this seque
//then its calculate Ans_array

short checkForSorted (int *A , size_t size, int *Ans_array )
{
int i =0,Ans_index=0 ;
short Sorted_order = -1 ; // -1 for in case all element are equal ;
//0 for decressing and 1 for incressing sequence checking for
//

for ( i =1 ; i<size ; i++)
{
if ( A[i-1] < A[i])
```

see more

• Reply • Share ›

nitin gupta iitian ➚ nitin gupta iitian • 2 years ago

@geeksforgeeks
if any thing wrong , let me know

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
/* Paste your code here (You may delete these lines if
```

• Reply • Share ›

Arpit Gupta • 2 years ago

stack implementation in O(n)

#include
#include
using namespace std;

stacks;
stackt;

void count(int a[],int n)
{
int next,i,x,y,z,cnt;
for(i=n-1;i>0;i--)
s.push(a[i]);
for(i=0;iy)
cnt++;
s.pop();
t.push(y);
if(!s.empty())

see more

1 • Reply • Share ›

Arpit Gupta ➦ Arpit Gupta • 2 years ago

dont know my the complete code cannot b copied

• Reply • Share ›

geeksforgeeks ➦ Arpit Gupta • 2 years ago

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

@Arpit Gupta: Thanks for sharing the code. Could you please p
sourcecode tags.

• Reply • Share ›

Arpit Gupta • 2 years ago

Here is the stack implementation in O(n)

```
#include
#include
using namespace std;

stacks;
stackt;

void count(int a[],int n)
{
int next,i,x,y,z,cnt;
for(i=n-1;i>0;i--)
s.push(a[i]);
for(i=0;iy)
cnt++;
s.pop();
t.push(y);
if(!s.emptv())
```

see more

• Reply • Share ›

pankaj • 2 years ago

use of stack will give O(n) solution. this problem is same as finding first larger

```
/* Paste your code here (You may delete these lines if not writing c
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

• Reply • Share ›

kartik ➚ pankaj • 2 years ago

@pankaj: Use of stacks seems like a promising idea. Could you pleas

• Reply • Share ›

Ali • 2 years ago

Your program didn't give right answer to the following input:
12 1 2 3 0 11 1
Your program produced:
6 2 2 2 0 1 0
whereas the right output is:
6 1 2 2 0 1 0
could you please notify me by email with your answer , this would help me a lo
thank you

• Reply • Share ›

kartik ➚ Ali • 2 years ago

@Ali: Take a closer look at the problem statement. It says distinct integ
valid one.

• Reply • Share ›

Mad Coder • 2 years ago

This problem can also be solved by using Binary Indexed Tree.

```
/* Paste your code here (You may delete these lines if not writing co
```

• Reply • Share ›

ramesh • 2 years ago

how can we modify the above method 2 to get the sum of smaller elements or
I think in the insert function , instead of doing coun+=size(node->left) , we nee

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

the node values. Is there any better way?
Thanks.

• Reply • Share ›

adithya • 2 years ago

```
#include <stdio.h>

int main()
{
        int a[20],i,n,j,count[20];
        for(i=0;i<n;i++)
        {
                count[i]=0;
        }
        printf("Enter the no of no's");
        scanf("%d",&n);
        printf("Enter the no's");
        for(i=0;i<n;i++)
        {
                scanf("%d",&a[i]);
        }
        for(i=0;i<n;i++)
        {
```

see more

• Reply • Share ›

pranay • 2 years ago

for the second method, can it be done by some in-build data structure in cpp a
Thanks.

• Reply • Share ›

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

GeeksforGeeks • 2 years ago

@Vinothkumar and @PsychoCoder:

There were problems in the original version of the second method. We have u have also added code for the same.

^ | v • Reply • Share ›

Vinothkumar • 2 years ago

can anyone explain the second method with example?

^ | v • Reply • Share ›

PsychoCoder • 2 years ago

In the 1st example for Method 2.

12 -> 2 -> 1
makes a rearrangement. Before rearrangement the left subtree count of 12 is ??

^ | v • Reply • Share ›

Rahul Sharma • 2 years ago

#include

```
void rightsmaller(int a[])
{
int count[20]={0},i=0,j=0;
count[0]=0;
for( i=0;i0)
{
for(j=0;j<i;j++)
{
if(a[j]<a[i])
{
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
}
}
}
}
```

see more

Reply • Share ›

kartik ➦ Rahul Sharma • 2 years ago

@Rahul Sharm: Could you please post the code again within the sourc

Reply • Share ›

murali529 • 2 years ago

we can even do this with a simple recursive method right

```
 public int numSmaller(int[] elements, int position){

if(position == (elements.length-1)){
return 0;
}

return numSmaller(int[] elements, position+1) + isMin(int[] elements,

}

public int isMin(int elements[], int presentPosition){

if(elements[presentPosition] > elements[presentPosition+1]){
return 1;
}
return 0;
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
}
```

Time Complexity will be O(n)

 • Reply • Share ›

Sanju • 2 years ago

Your second algorithm is wrong as per the question.

Questions says smaller elements to the right, not all smaller elements.

```
/* Paste your code here (You may delete these lines if not writing c
```

 • Reply • Share ›

GeeksforGeeks ➚ Sanju • 2 years ago

@Sanju: Please take a closer look at the algorithm. After finding smalle
the element from tree. By Deletion, we make sure that this node is not
elements to its right.

 • Reply • Share ›

kenny • 2 years ago

can someone explain the approach with an example ?

 • Reply • Share ›

Dheeraj • 2 years ago

Second case source code
pass the following parameters
arr=input arr;
min=array where the value to be stored
n=size of array arr

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

```
 typedef struct node
{
        int data;
        node *left,*right;
        int small;
        };
void fun(int arr[],int mins[],int n)
{
     node *root=NULL;
     for(int i=n-1;i>=0;i--)
     {
             int num=arr[i];
```

see more

• Reply • Share ›

nick ➚ Dheeraj • 2 years ago

i guess your tree isn't balance ..... so at the worst case scenario co
however it can be improved using AVL tree .... i had tried and its workir

```
/* Paste your code here (You may delete these lines if not wri
```

• Reply • Share ›

nick ➚ nick • 2 years ago

i mean to say complexity will be O(n2) not O(nlogn)

• Reply • Share ›

Dheeraj ➚ nick • 2 years ago

yup..for skewed trees it would be O(n^2)

```
/* Paste your code here (You may delete these li
```

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

• Reply • Share ›

lomash goyal • 2 years ago

you can first sort the array which can be done in O(nlogn) and after that fill the index of element in count smaller array.
one thing that need to be kept in mind that if the array is containing the duplicat also required to account that.but that also can be done in O(n).
so overall complexity would be O(nlogn).

• Reply • Share ›

sidh ➦ lomash goyal • 2 years ago

Looks like you didn't get the question. It is not about all smaller element

• Reply • Share ›

sidh • 2 years ago

A good example that shows use of self balancing BSTs.

Can somebody please share the code for method 2?

• Reply • Share ›

Subscribe Add Disqus to your site

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

@geeksforgeeks, **Some rights reserved** **Contact Us!** Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com