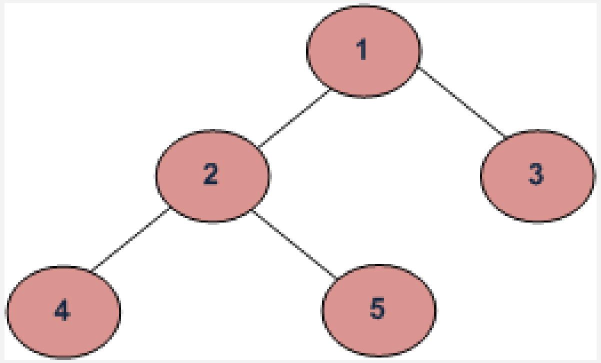
# Write a C program to Calculate Size of a tree

Size of a tree is the number of elements present in the tree. Size of the below tree is 5.



Example Tree

Size() function recursively calculates the size of a tree. It works as follows:

Size of a tree = Size of left subtree + 1 + Size of right subtree

**Algorithm:**

```
size(tree)
1. If tree is empty then return 0
2. Else
     (a) Get the size of left subtree recursively  i.e., call
          size( tree->left-subtree)
     (a) Get the size of right subtree recursively  i.e., call
          size( tree->right-subtree)
     (c) Calculate size of the tree as following:
```

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

```
          tree_size  =  size(left-subtree) + size(right-
                              subtree) + 1
    (d) Return tree_size
```

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

/* A binary tree node has data, pointer to left child
   and a pointer to right child */
struct node
{
    int data;
    struct node* left;
    struct node* right;
};

/* Helper function that allocates a new node with the
   given data and NULL left and right pointers. */
struct node* newNode(int data)
{
  struct node* node = (struct node*)
                       malloc(sizeof(struct node));
  node->data = data;
  node->left = NULL;
  node->right = NULL;

  return(node);
}

/* Computes the number of nodes in a tree. */
int size(struct node* node)
{
  if (node==NULL)
    return 0;
  else
    return(size(node->left) + 1 + size(node->right));
}

/* Driver program to test size function*/
int main()
{
  struct node *root = newNode(1);
  root->left        = newNode(2);
  root->right       = newNode(3);
  root->left->left  = newNode(4);
  root->left->right = newNode(5);
```

## 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

```
    printf("Size of the tree is %d", size(root));
    getchar();
    return 0;
}
```

**Time & Space Complexities:** Since this program is similar to traversal of tree, time and space complexities will be same as Tree traversal (Please see our Tree Traversal post for details)

## Related Tpoics:

- Print a Binary Tree in Vertical Order | Set 2 (Hashmap based Method)
- Print Right View of a Binary Tree
- Red-Black Tree | Set 3 (Delete)
- Construct a tree from Inorder and Level order traversals
- Print all nodes at distance k from a given node
- Print a Binary Tree in Vertical Order | Set 1
- Interval Tree
- Check if a given Binary Tree is height balanced like a Red-Black Tree

1 Tweet 0 0

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

17 Comments GeeksforGeeks

Sort by Newest ▾

Join the discussion…

san • 20 days ago

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication5

{

class Program

{

static void Main(string[] args)

{

Node<int> root = new Node<int>(1);
```

see more

^ | ˅ • Reply • Share ›

695 Subscribe

## Recent Comments

karthik it should have been max_wrap= max_wrap -...

Maximum circular subarray sum · 1 minute ago

affiszerv Your example has two 4s on row 3, that's why it...

Backtracking | Set 7 (Sudoku) · 45 minutes ago

**RVM** Can someone please elaborate this Qs from above...

Flipkart Interview | Set 6 · 1 hour ago

**Vishal Gupta** I talked about as an Interviewer in general,...

Software Engineering Lab, Samsung Interview | Set 2 · 1 hour ago

**@meya** Working solution for question 2 of 4f2f round....

Amazon Interview | Set 53 (For SDE-1) · 1 hour ago

sandeep void rearrange(struct node *head) {...

Given a linked list, reverse alternate nodes and append at the end · 3 hours ago

**rikitic** • 10 months ago

inserting count variable in pre-order traversal will do it..i suppose . any one cor

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

Reply • Share ›

**Vibhu Tiwari** rikitic • 10 months ago

I think any order traversal will do it because in the traversal of the trees increment the count variable as in these traversals every time we are e number of these elements will hold in the variable count.

1 • Reply • Share ›

**abhishek08aug** • a year ago

Below is the C++ code for getting the size of the tree. I have just extended my http://www.geeksforgeeks.org/6...

Have made all the _* functions in tree class private as they are not called from member in tree class to keep track of the number of inserts i.e. size of the BS

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

class tree_node {
  private:
    int data;
    tree_node * left;
    tree_node * right;
  public:
    tree_node() {
      left=NULL;
      right=NULL;
```

see more

^ | ⌄ • Reply • Share ›

**abhishek08aug** ➦ abhishek08aug • a year ago

Preorder Traversal

5 3 1 7 6 50

Inorder Traversal

1 3 5 6 7 50

Postorder Traversal

1 3 6 50 7 5

Size of the tree (as stored in tree object) is: 6

Size of the tree is: 6

^ | ⌄ • Reply • Share ›

**Nikin** • a year ago

```
int size(node *sr)
{
if(sr) return size(sr->left) + size(sr->right) + 1;
return 0;
}
```

1 ^ | ⌄ • Reply • Share ›

**Sun** • 2 years ago

Why doesn't this code work ?

```
        private int size(Node node,int count)
```

```
{

        if(node == null)
        {
                return 0;
        }


        size(node.left,count);
        count++;
        size(node.right,count);

        return count;






}
```

⌃ | ⌄ • Reply • Share ›

**varahi** ➦ Sun • 25 days ago

both left and right sub tree's count needs to be added

⌃ | ⌄ • Reply • Share ›

**Srikar** ➦ Sun • 2 months ago

count value is not changing....

⌃ | ⌄ • Reply • Share ›

**Gaurav pruthi** ➦ Sun • 4 months ago

Each count variable is having its local scope.. either take it global or us

1 ^ | ⌄ • Reply • Share ›

**Karun** ➦ Sun • 5 months ago

You are not providing a pointer to the parameters you are passing. The

1 ^ | ⌄ • Reply • Share ›

**Uma Trika** ➦ Sun • 5 months ago

```
void sizeUtil(struct node *node, int *size)
{
if(node == NULL)
return ;
*size = *size +1;
sizeUtil(node->left, size);
sizeUtil(node->right, size);
}
```

2 ^ | ⌄ • Reply • Share ›

**prakash** ➦ Sun • a year ago

hi dude,

this wont work. since ur count variable is not a pointer so,count of left s

1 ^ | ⌄ • Reply • Share ›

**enchantress** ➦ Sun • 2 years ago

count is local here to every recursive call..

either make count global or do count+=size(node.left,count);

count++;

count+=size(node.right,count);

^ | ⌄ • Reply • Share ›

**Marsha Donna** ➦ enchantress • 2 months ago

@enchantress your code wil not work ...for the tree shwn in ab

Yo..uma's approach is correct

^ | ⌄ • Reply • Share ›

**Sunil** • 3 years ago

The function to find the size of the tree can also be written like this.

```
int size(struct node* root)
{
        static int count=0;
        if(root!=NULL)
        {
                size(root->llink);
                count++;
                size(root->rlink);
        }
        return count;
}
```

^ | ⌄ • Reply • Share ›

**Pandhari** ➦ Sunil • a year ago

Not needed  static int count=0;

It requires extra memory and that exists throughout life of process as it

^ | ⌄ • Reply • Share ›

✉ Subscribe    Add Disqus to your site