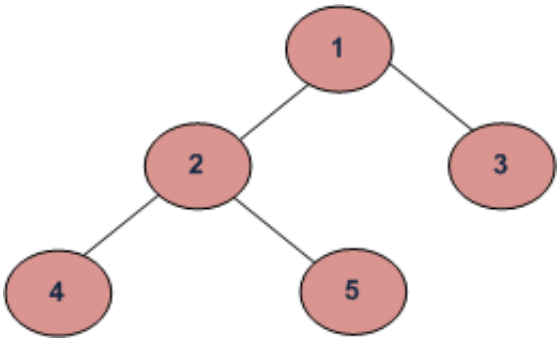
Write a C Program to Find the Maximum Depth or Height of a Tree

Maximum depth or height of the below tree is 3.



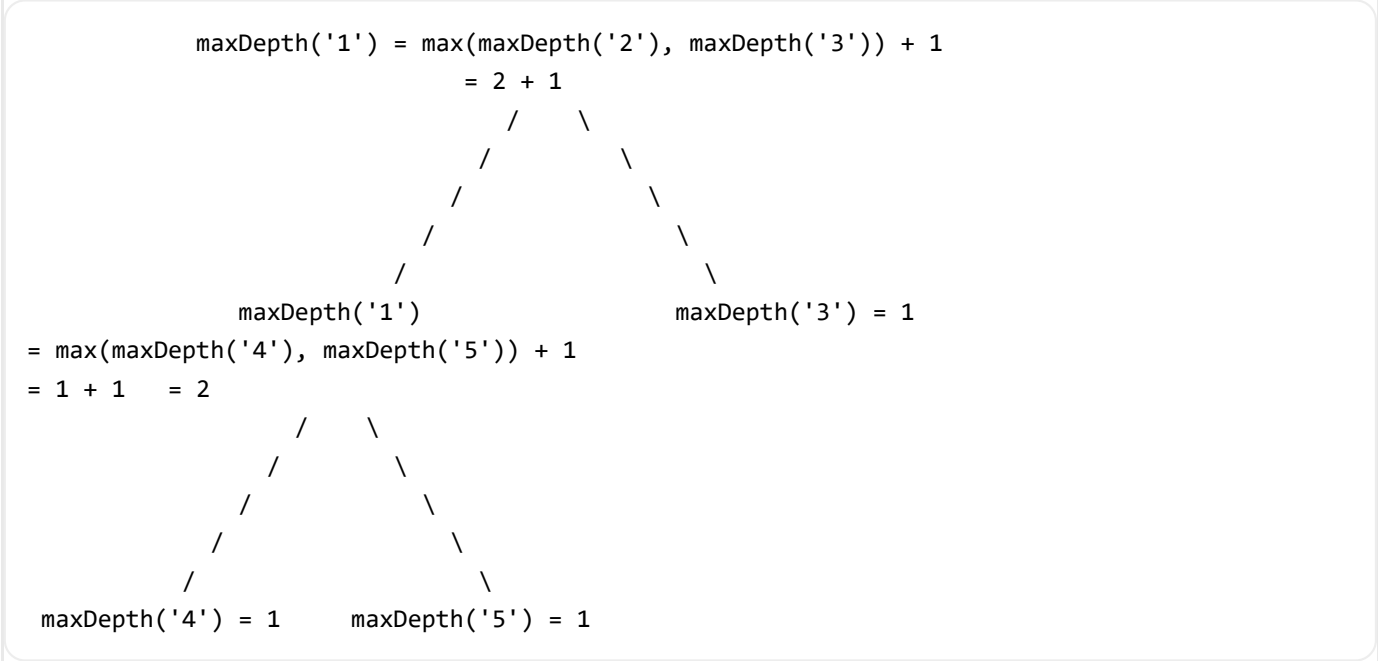
Example Tree

Recursively calculate height of left and right subtrees of a node and assign height to the node as max of the heights of two children plus 1. See below pseudo code and program for details.

Algorithm:

```
 maxDepth()
1. If tree is empty then return 0
2. Else
     (a) Get the max depth of left subtree recursively  i.e.,
          call maxDepth( tree->left-subtree)
     (a) Get the max depth of right subtree recursively  i.e.,
          call maxDepth( tree->right-subtree)
     (c) Get the max of max depths of left and right
          subtrees and add 1 to it for the current node.
         max_depth = max(max dept of left subtree,
                         max depth of right subtree)
                         + 1
     (d) Return max_depth
```

See the below diagram for more clarity about execution of the recursive function maxDepth() for above example tree.



Implementation:

```
#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;
};

/* Compute the "maxDepth" of a tree -- the number of
    nodes along the longest path from the root node
    down to the farthest leaf node.*/
int maxDepth(struct node* node)
{
   if (node==NULL)
       return 0;
   else
   {
       /* compute the depth of each subtree */
       int lDepth = maxDepth(node->left);
       int rDepth = maxDepth(node->right);

       /* use the larger one */
       if (lDepth > rDepth)
           return(lDepth+1);
       else return(rDepth+1);
   }
}

/* 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);
}

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);

    printf("Hight of tree is %d", maxDepth(root));

    getchar();
    return 0;
}
```

Run on IDE

Time Complexity: O(n) (Please see our post Tree Traversal for details)

References:

http://cslibrary.stanford.edu/110/BinaryTrees.html

152 Comments Category: Trees Tags: Height of a Tree, Tree Traveral, Trees

Related Posts:

- Find Count of Single Valued Subtrees
- Check if a given array can represent Preorder Traversal of Binary Search Tree
- Mirror of n-ary Tree
- Succinct Encoding of Binary Tree
- Construct Binary Tree from given Parent Array representation
- Symmetric Tree (Mirror Image of itself)
- Find Minimum Depth of a Binary Tree
- Maximum Path Sum in a Binary Tree

1 Average Difficulty : **1/5.0**
Based on **2** vote(s)

(Login to Rate)

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.

152 Comments GeeksforGeeks 1 Login

Recommend 3 Share Sort by Newest

Join the discussion…

Aditi Dubey • 11 days ago

I think the rectification in the code is return -1 instead of retun 0.The code is as follows:

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

struct node{

int data;

struct node*left;

struct node*right;

};

struct node*newnode(int data)

{

see more

• Reply • Share ›

Amanshu Kataria • 14 days ago

@GeeksforGeeks The height of a tree is number of edges on longest path from root node to leaf node. So according to this definition the output of the code should be 2 not 3. Please check the code and this link too http://stackoverflow.com/quest....

• Reply • Share ›

Pratik • 2 months ago

Where we are incrementing ??

if (node==NULL)

return 0;

```
else

{

/* compute the depth of each subtree */

int lDepth = maxDepth(node->left);

int rDepth = maxDepth(node->right);

/* use the larger one */

if (lDepth > rDepth)

return(lDepth+1);

else return(rDepth+1);

}
```

• Reply • Share ›

Tushar Dwivedi ➦ Pratik • a month ago

Incrementing what? It's recursion.

The function will keep calling itself.

```
int lDepth = maxDepth(node->left);

int rDepth = maxDepth(node->right);
.
Unless (node==NULL)
.
```

And then it will take the larger out of the two previous recursive calls.

• Reply • Share ›

Sabarigirish V • 2 months ago

heh i think it gives 1 + height as answer, basically we see the no of edges from the root to the leaf(longest path)ie edges, but here we get no of nodes in the path . i think we need to return -1 to correct it .

1 • Reply • Share ›

Tonmoy Rakshit • 3 months ago

Cant understand how the increment operation is happening!

• Reply • Share ›

Akshay Tata • 4 months ago

Can we calculate the size of the tree by doing a traversal and do a log2(size + 1)? In this

case the size of the tree is 5. So Height becomes log2 (5+1) = log2(6) = 2.44 and then we round it to the next integer which is 3? Am I right. Its in order traversal so would take O(n) time, where n is the number of nodes.

1 • Reply • Share ›

RAJAT ➚ Akshay Tata • 2 months ago

I think complexity will increase significantly you have to traverse all nodes then you calculate O(n) and in this case it will be 0(logn)

• Reply • Share ›

shilpi • 4 months ago

@geeksforgeeks

there should be 1 more condition after this if(node==NULL) return 0;

this one -> if(node->left==NULL && node->right==NULL)
return 0;

1 • Reply • Share ›

Varun Sagar ➚ shilpi • 3 months ago

not adding that condition is equivalent to adding that condition and returning 0, think about it (adding nothing = adding a zero)

• Reply • Share ›

Dhrtzzz • 4 months ago

```
int height(node * root)
{
int l,r;
if(root==NULL)
{ return 0 ; }

l = height(root->left)+1;
r = height(root->right)+1;
return max(l,r) ;
}
```

1 • Reply • Share ›

Sarthak Garg ➚ Dhrtzzz • 4 months ago

or simply use: return 1 + max(height(root->left), height(root->right));

1 • Reply • Share ›

Pranjal Dubey ➚ Sarthak Garg • 18 days ago

wouldn't it return height = 1 even in the case when there is only one node in the tree??

• Reply • Share ›

sasha • 4 months ago

How in
int lDepth=maxDepth(node->left);
Int ldepth is incrementing,there is no such incrementation?

1 • Reply • Share ›

Sarthak Garg sasha • 4 months ago

You're returning lDepth + 1. That's the increment.

• Reply • Share ›

Avatar • 4 months ago

{
if(!tree)
return 0;
return 1+max(tree->left,tree->right);
}

1 • Reply • Share ›

typing.. • 4 months ago

@geeksforgeeks Solution is not correct!!
http://www.cs.cmu.edu/~adamchi..., look at here,
point number two-
"The height of a node is the number of edges from the node to the deepest leaf",
and point number three-
"The height of a tree is a height of the root", according to this the height of a rooted tree(tree with only one node) should be 0, but this code outputs 1, so in function maxDepth() it should be return -1, correct me if m wrong..

1 • Reply • Share ›

JavaCoder typing.. • 4 months ago

Some consider a tree with one node to be of height 1 while some consider it to be of height 0. I think you can check in the book CLRS and follow what it tells.

• Reply • Share ›

typing.. JavaCoder • 4 months ago

i didnt read CLRS completely, but my point is if you are asked for height, let say of a building, then you will say 'n' floors, but you will never say 'n+1'(includes 'n' ceilings + a ground), in similar way height of a tree should be number of edges, not number of nodes..

• Reply • Share ›

Tushar Dwivedi typing... • a month ago

Your analogy of "floors" itself explains the scenario.

In some countries, they start counting floors from 0 (ground floor), and in some countries, they count from 1.
So, for the same building they will give different numbers based on their understanding (n & n+1).
.
Same with tree data structure.

• Reply • Share ›

Anandh Perumal typing.. • 4 months ago

NO,tree with one node has a height of 1,while its level will be 0

• Reply • Share ›

typing.. Anandh Perumal • 4 months ago

My point is if you are asked for height, let say of a building, then you will say 'n' floors, but you will never say 'n+1'(includes 'n' ceilings + a ground), in similar way height of a tree should be number of edges, not number of nodes..
If you are sure enough then plz provide me some justification..

• Reply • Share ›

Pingu typing.. • 4 months ago

please rectify the code and post it here.
PLEASE

• Reply • Share ›

typing.. Pingu • 4 months ago

not much rectification needed, only replace return 0 with return 1, here is code:
http://ideone.com/dLLkXU

1 • Reply • Share ›

Sandeep Kumar • 4 months ago

```
int depth_of_tree(node *root){
if(!root)
return 0;
else
return 1+(depth_of_tree(root->left)>depth_of_tree(root->right)?depth_of_tree(root->left):depth_of_tree(root->right));
```

• Reply • Share ›

Shantanu • 4 months ago

```
int maxDepth(struct node *node)

{
if(node==NULL)
return 0;
else
{
return(1+max(maxDepth(node->left),maxDepth(node->right)));
}
}
```

1 • Reply • Share ›

Divjot Singh • 4 months ago

```
public static int maxDepth(Node node) {
return node == null ? 0 : (Math.max(maxDepth(node.left), maxDepth(node.right)) + 1);
}
```

• Reply • Share ›

wangjam ➚ Divjot Singh • 4 months ago

shouldn't be it Math.max(maxDepth(node.left) , maxDepth(node.right))+ 1 ; ?

• Reply • Share ›

Divjot Singh ➚ wangjam • 4 months ago

My bad! I intended to write that.

• Reply • Share ›

Dipankar Bhardwaj • 4 months ago

http://code.geeksforgeeks.org/...

• Reply • Share ›

XyronyX • 5 months ago

can anyone tell me how exactly is the value of ldepth and rdepth increasing even if there is no such sign of any increment ?

1 • Reply • Share ›

JavaCoder ➚ XyronyX • 4 months ago

```
if (lDepth > rDepth)
return(lDepth+1);
else return(rDepth+1);
```

The "+1" in these lines increases the value.

1 • Reply • Share ›

Hitesh Saini ➚ XyronyX • 4 months ago

take example of a skew tree

maintain the stack for recursion you'll definitely get your answer. if not please reply i'll explain you an example for the same..:)

• Reply • Share ›

hermione ➚ Hitesh Saini • 3 months ago

can you explain with an example?

• Reply • Share ›

Hitesh Saini ➚ hermione • 3 months ago

see the video https://www.youtube.com/watch?...

if still you have any doubt please reply..:)

• Reply • Share ›

V_CODER ➚ Hitesh Saini • 4 months ago

@Hitesh,can u pls give an example or any link explaining such concept?

• Reply • Share ›

Hitesh Saini ➚ V_CODER • 4 months ago

https://www.youtube.com/watch?...
see this for more clarification.

• Reply • Share ›

#InnerPeace ➚ XyronyX • 4 months ago

Once you try to understand recursion will get to know how ldepth and rdepth is increasing :)

• Reply • Share ›

Ananda kumar N • 5 months ago

```
int getHeight(struct node *root){
int l=getHeight(root->left);
int r=getHeight(root->right);
if(root==NULL)
return 0;
else
return( l>r?i:r )+1;

}
```

• Reply • Share ›

Anonymous • 5 months ago

What if there are million number of nodes in a tree ??

• Reply • Share ›

Avata This comment was deleted.

Killing Idiots Guest • 5 months ago
Abey sale, jab less efficient hai toh jhak marane k liye post kar raha hai.
• Reply • Share ›

pk • 5 months ago
Just include #include<algorithm> The main logic is as follows
int maxDepth(struct node* node)
{
if (node == NULL)
return 0;
return 1 + max(maxDepth(node->left),maxDepth(node->right));
}
1 • Reply • Share ›

LNR • 5 months ago
Height of a tree:
Number of edges from the root node to its longest leaf node.

Admin please note that instead of counting the number of edges, the program is counting the number of nodes. Please correct the code.
1 • Reply • Share ›

Holden • 6 months ago
Java implementation: http://ideone.com/uo9FYw
• Reply • Share ›

Abhimanyu • 6 months ago
IN size function if i use variable
a=size(root->left);
b=size(root->right);
return (a+b);
Why it prints zero ?same was working in calculating max_Depth code where we were doing ldepth=maxdepth(root->left),rdepth=maxdepth(root->right) it was printing maxdepth but in case of size output shows zero?
• Reply • Share ›

Joey • 6 months ago
Isnt this right ?

```
/* Compute the "maxDepth" of a tree -- the number of
nodes along the longest path from the root node
down to the farthest leaf node.*/
int maxDepth(struct node* node)
{
if (node==NULL)
return 0;
else
{
return 1+ max(maxDepth(node->left),maxDepth(node->right));
}
}
```

1 • Reply • Share ›

aa1992 Joey • 6 months ago

yup its correct.
check this http://ideone.com/8p7VY6

• Reply • Share ›

gizmogaurav • 7 months ago

Correct me If am wrong.
According to my understanding , The height of above example should be 2.

As The height of a tree is the number of edges on the longest path from the node to a leaf.A leaf node will have a height of 0.

• Reply • Share ›

Prince Bharti gizmogaurav • 5 months ago

yes it should be !! if u are assuming the root node at level 0. here they are assuming the root node at level 1.

• Reply • Share ›

Load more comments