

Write C Code to Determine if Two Trees are Identical

Two trees are identical when they have same data and arrangement of data is also same.

To identify if two trees are identical, we need to traverse both trees simultaneously, and while traversing we need to compare data and children of the trees.

Algorithm:

```
sameTree(tree1, tree2)
1. If both trees are empty then return 1.
2. Else If both trees are non -empty
    (a) Check data of the root nodes (tree1->data == tree2->data)
    (b) Check left subtrees recursively i.e., call sameTree(
        tree1->left_subtree, tree2->left_subtree)
    (c) Check right subtrees recursively i.e., call sameTree(
        tree1->right_subtree, tree2->right_subtree)
    (d) If a,b and c are true then return 1.
3 Else return 0 (one is empty and other is not)
```

```
#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);
}
```

```
/* Given two trees, return true if they are
structurally identical */
int identicalTrees(struct node* a, struct node* b)
{
    /*1. both empty */
    if (a==NULL && b==NULL)
        return 1;

    /* 2. both non-empty -> compare them */
    if (a!=NULL && b!=NULL)
    {
        return
        (
            a->data == b->data &&
            identicalTrees(a->left, b->left) &&
            identicalTrees(a->right, b->right)
        );
    }

    /* 3. one empty, one not -> false */
    return 0;
}

/* Driver program to test identicalTrees function*/
int main()
{
    struct node *root1 = newNode(1);
    struct node *root2 = newNode(1);
    root1->left = newNode(2);
    root1->right = newNode(3);
    root1->left->left = newNode(4);
    root1->left->right = newNode(5);

    root2->left = newNode(2);
    root2->right = newNode(3);
    root2->left->left = newNode(4);
    root2->left->right = newNode(5);

    if(identicalTrees(root1, root2))
        printf("Both tree are identical.");
    else
        printf("Trees are not identical.");

    getchar();
    return 0;
}
```

[Run on IDE](#)

Time Complexity:

Complexity of the identicalTree() will be according to the tree with lesser number of nodes. Let number of nodes in two trees be m and n then complexity of sameTree() is $O(m)$ where $m < n$.

95 Comments Category: [Trees](#) Tags: [Tree Traversal](#), [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

2

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

(Login to Rate)

Like Share 8 people like this. Be the first of your friends.

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

95 Comments

GeeksforGeeks

1 Login▼

♥ Recommend 1

🔗 Share

Sort by Newest▼



Join the discussion...

**Tonmoy Rakshit** • 3 months ago

Can any1 explain the recursive operation in detail?

^ | v • Reply • Share ›

**Jayesh** • 3 months ago

Java Implementation

<http://javabypatel.blogspot.in...>

^ | v • Reply • Share ›

**crazyforstudy it** • 4 months ago

So. It is a top down approach.....correct me if I am wrong

^ | v • Reply • Share ›

**Ashish** • 4 months ago

There is no reference to sameTree() used in Time Complexity analysis.It should be identicalTrees().

^ | v • Reply • Share ›

**Pingu** • 4 months ago

```
return  
(  
a->data == b->data &&  
identicalTrees(a->left, b->left) &&  
identicalTrees(a->right, b->right)  
);
```

if all these conditions evaluate to true then it returns 1

is this the logic ?

^ | v • Reply • Share ›

**Hitesh Saini** → Pingu • 4 months ago

yes

^ | v • Reply • Share ›

**Pingu** • 4 months ago

if inorder traversal of 2 trees is same does that mean they both are identical?

^ | v • Reply • Share ›

**Aayush** → Pingu • 4 months ago

Nope, they might not be identical Pingu

^ | v • Reply • Share ›

**Waqas Hamid** • 4 months ago

If one of trees contains only the root and the other has multiple nodes run time error occurs.....Checked on Ideone...

^ | v • Reply • Share ›

**Dipankar Bhardwaj** • 4 months ago

Simple and clean <http://code.geeksforgeeks.org/...>

1 ^ | v • Reply • Share ›

**Lokesh** • 4 months ago

<http://code.geeksforgeeks.org/...>

^ | v • Reply • Share ›

**sunil** • 5 months ago

if i do pre/post/inorder traversal of the two tree and stored them into two different array.Is that still work. I know that going to take more space, but is it true to do so.

^ | v • Reply • Share ›

**Anand Barnwal** → sunil • 5 months ago

**Hitesh Saini** • 5 months ago

Suppose the two trees are:

B <- A -> NULL (here A is root, left child is B and right child is NULL)

And

NULL <- A -> B (here A is root, left child is NULL and right child is B)

The preorder traversal of both trees will give "AB". But the trees are not identical.

Hope, it clears your doubt.

4 ^ | v • Reply • Share ›

**pk** • 5 months ago

```
bool compare(struct node* node1, struct node* node2)
```

{

```
if (node1 == NULL && node2 == NULL)
```

```
return true;
```

```
if (node1 == NULL || node2 == NULL)
```

```
return false;
```

```
if (node1->data == node2->data)
```

```
return compare(node1->left, node2->left) && compare(node1->right, node2->right);
```

```
return false;
```

}

3 ^ | v • Reply • Share ›

**dhiru** • 6 months ago

can we check the identity of the tree by comparing the preorder of two trees?

^

| v • Reply • Share ›

**Hitesh Saini** → dhiru • 5 months ago

yes we can but only if both the trees have same number of nodes.

if the number of nodes are different then it will not work.

take a example make two trees of different length and in larger tree preserve the state of nodes of small tree (means both the tree are identical) but when you right preorder of both tree you will find yhem different .

Hope it clears your doubt .

^ | v • Reply • Share ›

**dhiru** → Hitesh Saini • 4 months ago

but for two trees to be identical they should have same number of nodes

^

| v • Reply • Share ›

**Hitesh Saini** → dhiru • 4 months ago

consider two trees

A
/\n
B C
and

A
/
B
/
C

preorder traversal of both the tree are same but they are not identical.

in my previous comment same number of nodes means same arrangement of nodes.

^ | v • Reply • Share ›



Mission Peace • 7 months ago

Checkout my channel on youtube on binary trees

<https://www.youtube.com/channe...>

^ | v • Reply • Share ›



Anonymous • 7 months ago

Code in Java :

```
int compare(Node rt1 , Node rt2){
    if(rt1 == null && rt2 == null)
        return 1;
    if(rt1 == null || rt2 == null)
        return 0;
    if(rt1.data == rt2.data){
        return Math.min(compare(rt1.left,rt2.left),compare(rt1.right,rt2.right));
    }else{
        return 0;
    }
}
```

1 ^ | v • Reply • Share ›



Holden ➔ Anonymous • 5 months ago

You can define it 'boolean' with 'true' and 'false' :)

^ | v • Reply • Share ›



natasha • 8 months ago

int TreeFuncLib::IdenticalTree(struct node* tree1. struct node* tree2)

```

{

if(tree1 == NULL && tree2 == NULL)

return 1;

else if(tree1 == NULL && tree2 != NULL)

return(0);

else if(tree2 == NULL && tree1 != NULL)

return(0);

else

{

return((tree1->data == tree2->data)&&(IdenticalTree(tree1->left,tree2->left))&&
(IdenticalTree(tree1->right,tree2->right)));

}

}

```

1 ^ | v • Reply • Share ›



Jerry Goyal • 8 months ago

```

int it(struct node* r1, struct node* r2)

{

    if(!r1&&!r2) return 1;

    if(!r1||!r2) return 0;

    return (r1->data==r2->data && it(r1->left,r2->left)
        && it(r1->right,r2->right) );

}

```

1 ^ | v • Reply • Share ›



Jerry Goyal • 8 months ago

instead, use a global variable flag to check

```

int flag=0;

identicalTrees(struct node* a, struct node* b)

{

```

```

    if(a==NULL&&b==NULL){
        return;
    }
    if(a==NULL || b==NULL){
        flag=1;
        return;
    }
    if(a->data!=b->data){
        flag=1;
        return;
    }
    identicalTrees(a->left,b->left);
    identicalTrees(a->right,b->right);
}

if flag=1 i.e.not identical

```

^ | v • Reply • Share ›



LNR → Jerry Goyal • 5 months ago

What if either of the trees is NULL?

Won't

if(a->data!=b->data)

statement lead to an error?

^ | v • Reply • Share ›



Jerry Goyal → LNR • 5 months ago

thanks for pointing out. updated the code.

^ | v • Reply • Share ›



surbhijain93 • 8 months ago

```
int i(struct node* n1,struct node* n2)
```

```
{
```

```
if(n1==NULL && n2==NULL)
```

```
return 1;
```

```
if((n1==NULL)||(n2==NULL))
```

```
return 0;
```

```
return
```

```
(n1->data==n2->data &&
```



```
i(n1->left,n2->left) &&
```

```
i(n1->right,n2->right));
```

```
}
```

^ | v • Reply • Share ›



trend_setter • 9 months ago

what is the difference between return 0, return 1 and return -1, How does it differs from exit?

^ | v • Reply • Share ›



Aditya Goel → trend_setter • 8 months ago

exit from anywhere on your code will terminate execution immediately.

return from main() is equivalent to exit() function. The program terminates immediately execution with exit status set as the value passed to return. Status 0 means the program succeeded. Status different from 0 means the program exited due to error or abnormally.

return in an inner function (not main) will terminate immediately the execution of the specific function returning the given result to the calling function.

1 ^ | v • Reply • Share ›



Rahul → trend_setter • 9 months ago

return is used to return program control to the calling function. Exit is used to exit the code

^ | v • Reply • Share ›



Anand Barnwal → Rahul • 5 months ago

exit(STATUS) is a system call i.e. used to terminate the current process.

^ | v • Reply • Share ›



trend_setter → Rahul • 9 months ago

Thanks for the reply. What does return 0 and return 1 means?

^ | v • Reply • Share ›



Holden → trend_setter • 5 months ago

true and false :)

^ | v • Reply • Share ›



trend_setter • 9 months ago

sometimes we put struct node* node and other times struct node *node, what is the difference ?

^ | v • Reply • Share ›



Hari → trend_setter • 9 months ago

no difference.. it will always be interpreted as a pointer of type (struct node)

1 ^ | v • Reply • Share ›



Ajitesh Mandal • 9 months ago

<http://ideone.com/WsJylq>

^ | v • Reply • Share ›



surbhijain93 • 9 months ago

```
bool compare(nodeptr node1,nodeptr node2)
{
    if(node1==NULL&& node2==NULL)
        return 1;
    else if(node1!=NULL&& node2==NULL||node1==NULL&&node2!=NULL)
        return 0;
    else if(node1->data==node2->data)
    {
        printf("%d \n",node1->data);
        return(compare(node1->left,node2->left)&&compare(node1->right,node2->right));
    }
    else
        return 0;
}
```

^ | v • Reply • Share ›



Rajeev → surbhijain93 • 6 months ago

in your second if condition you neednt check for all four conditions. Just an node1 == NULL || node2 == NULL would do as you are already checking the case where both are null before it

1 ^ | v • Reply • Share ›



Saurabh Gupta • a year ago

Inorder traversal of both the trees simultaneously can also be an approach to the problem. Here is my code, do comment in case of any flaws or improvements.

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
    int num;
    node *left, *right;
};
```

```

node *root1 = NULL;
node *root2 = NULL;
node *newnode(int n)
{
node *ptr = (node *)malloc(sizeof(node));
ptr ->num = n;
ptr ->left = NULL;
ptr ->right = NULL;

```

[see more](#)

^ | v • [Reply](#) • [Share](#) ›



neelabhsingh → [Saurabh Gupta](#) • a year ago

I think your following code will not run, Suppose root1==NULL, root1!=NULL, so your first condition fails. Now in your second condition root1->num will give you runtime error, please carefully check your code. Add if(root!=NULL && root2!=NULL) then your code is fine.

```

condition1:
if(root1 == NULL && root2 == NULL)
return;
condition2:
if(root1 ->num == root2 ->num)
{
identical(root1 ->left, root2 ->left);
identical(root1 ->right, root2 ->right);
}

```

4 ^ | v • [Reply](#) • [Share](#) ›



Sumit Thakur • a year ago

Calculate in-order\pre-order\post-order for both trees and store them in two arrays. Now compare both arrays. Is it a good idea ?

4 ^ | v • [Reply](#) • [Share](#) ›



Pranav Kumar Jha → [Sumit Thakur](#) • a year ago

See for yourself!

tree1: 1

/ \

2 3

tree2: 2

\

1

\

-

3

Both trees have same inorder traversals: 2 1 3

And they are definitely not identical!

11 ^ | v • Reply • Share ›



Holden → Pranav Kumar Jha • 5 months ago

Nice example :)

^ | v • Reply • Share ›



himan → Sumit Thakur • a year ago

Two different trees can have same in-order/pre-order/post-order.

You can do with both in-order and preorder but this will increase time and space complexity.

8 ^ | v • Reply • Share ›



PRime → Sumit Thakur • a year ago

You can do like this but this will increase the time complexity.

^ | v • Reply • Share ›



ryan • a year ago

```
bool compare(nodeptr node1,nodeptr node2)
{
if(node1==NULL&& node2==NULL)
return 1;
else if(node1!=NULL&& node2==NULL||node1==NULL&&node2!=NULL)
return 0;
else if(node1->data==node2->data)
{
printf("%d \n",node1->data);
return(compare(node1->left,node2->left)&&compare(node1->right,node2->right));
}
else
return 0;
}
```

2 ^ | v • Reply • Share ›



RK → ryan • 9 months ago

You can reduce the size of your code:

```
bool compare(nodeptr node1,nodeptr node2)
{
if ( node1==NULL && node2==NULL ) return 1;
if ( node1==NULL || node2==NULL ) return 0;
```

```
return ( node1->data == node2->data && compare(node1->left,node2->left) &&
compare(node1->right,node2->right));
}
```

^ | v • Reply • Share ›



Amit Agrawal → ryan • a year ago

it's correct...

^ | v • Reply • Share ›



Guest → ryan • a year ago

Not correct

^ | v • Reply • Share ›



ryan → Guest • a year ago

tell the case when it fail

^ | v • Reply • Share ›

Load more comments



Subscribe



Add Disqus to your site



Privacy

@geeksforgeeks, Some rights reserved

[Contact Us!](#)

[About Us!](#)

[Advertise with us!](#)