

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;
};
```

Google™ Custom Search



GeeksforGeeks



52,731 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

```

/* 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);
}

```



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

```

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;
}

```

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 $m < n$.



Better Than Hadoop.

HPCC Systems is Big Data Processing and Analytics
Open Source. Proven. Trusted.

LexisNexis® Learn More

Related Topics:

- [Print a Binary Tree in Vertical Order | Set 2 \(Hashmap based Method\)](#)
- [Print Right View of a Binary Tree](#)
- [Red-Black Tree | Set 3 \(Delete\)](#)



ORACLE® 12^c
DATABASE

Oracle Database 12c Training and Certification

Explore new features, maximize
your skills, and achieve your
potential.

View Courses

ORACLE®

- 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



4

Tweet 0

2

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

24 Comments

GeeksforGeeks

Sort by Newest ▾



Join the discussion...



neelabhsingh · 4 months ago

Please noted it down following condition is essential when you are comparing nonidentical trees. It will compares nodes only if they are non null. So following in C and NULL Pointer Exception in Java.

If you are comparing the Identical you will not required the following conditions.

/* 2. both non-empty -> compare them */

if (a!=NULL && b!=NULL)

^ | ▾ · Reply · Share ▸



Uma Trika · 5 months ago

```
int identicalTrees(struct node* a, struct node* b)
{
    if(a== NULL && b == NULL)
        return 1;
    if ((a==NULL && b!=NULL) || (a!=NULL && b==NULL))
        return 0;
```

695



Subscribe

Recent Comments

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

[Maximum circular subarray sum](#) · 2 minutes ago

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

[Backtracking | Set 7 \(Sudoku\)](#) · 46 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

AdChoices

[▶ Binary Tree](#)

[▶ Java Tree](#)

[▶ Java to C++](#)

```

if(a->data != b->data)
return 0;
identicalTrees(a->left, b->left);
identicalTrees(a->right, b->right);

}

```

^ | v • Reply • Share ›



Marsha Donna → Uma Trika • 2 months ago

Your code doesnt work in case the 2 trees dont have identical data bec
call to the left and right subtree has not been stored anywhere and not

^ | v • Reply • Share ›



Marsha Donna → Marsha Donna • 2 months ago

i think the corection to be made is

```

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

{

if(a== NULL && b == NULL)

return 1;

if ((a==NULL && b!=NULL) || (a!=NULL && b==NULL))

return 0;

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

return 0;

return identicalTrees(a->left, b->left)&&identicalTrees(a->right,

```

AdChoices ▶

▶ [Tree Trees](#)

▶ [Tree Root](#)

▶ [Tree View](#)

AdChoices ▶

▶ [In Memory Tree](#)

▶ [Java C Code](#)

▶ [Compare C++ Code](#)

```
}
```

^ | v • Reply • Share ›



gaurav • 8 months ago

Won't a level order traversal be sufficient to check the identical arrangement a wrong.

^ | v • Reply • Share ›



pavansrinivas → gaurav • 7 months ago

Using LevelOrder Traversal in JAVA

```
boolean areIdentical(Node r1,Node r2){
    Node temp = r1;
    Node temp2 = r2;
    Queue<node> q = new LinkedList<node>();
    Queue<node> q2 = new LinkedList<node>();
    q.add(temp);
    q2.add(r2);
    while (!q.isEmpty()&&!q2.isEmpty()) {
        temp = q.remove();
        temp2 = q2.remove();

        if((temp==null&&temp2!=null) || (temp!=null&&temp2==null))
            return false;
    }
    if(temp!=null&&temp2!=null){
```

see more

^ | v • Reply • Share ›



Anonymous → pavansrinivas • 5 months ago



i think , level order traversal will not work because two different traversed in Level order.

For example: - below tree will process node in same order so y both trees are same..

a)

1

/

2

/

3

b)

1

\

2

\

3

please correct me if i misunderstood your approach.

1 ^ | v • Reply • Share ›



pavansrinivas → Anonymous • 5 months ago

My approach gives the correct result.I used 2 queues or tree..I simultaneously travel the 2 trees and if I find null in vice-verse I return false..

In your condition , first root is entered in first queue and both the left child are added..when popped,node in first queue is null, so it returns false....

^ | v • Reply • Share ›



Kartik → gaurav • 8 months ago



There can be many ways to solve a problem. You can do it using level turn out to be more complex than simple recursive code. Time Comple $O(n)$ extra space.

1 ^ | v • Reply • Share ›



abhishek08aug • a year ago

C++ code: extended from my earlier code in: <http://www.geeksforgeeks.org/w>

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

class tree_node {
private:
    int data;
    tree_node * left;
    tree_node * right;
public:
    tree_node() {
        left=NULL;
        right=NULL;
    }
    void set_data(int data) {
        this->data=data;
    }
}
```

[see more](#)

^ | v • Reply • Share ›



Gaurav Verma • a year ago

can anyone explain return (node)?

^ | v • Reply • Share ›



Marsha Donna → Gaurav Verma · 2 months ago

u hav to go through basics of pointers to understand that

^ | v · Reply · Share ›



Mokone Mokemane · a year ago

gr8 code

^ | v · Reply · Share ›



Nikin · a year ago

```
bool areIdentical(node *sr1, node *sr2)
{
    if(sr1 == NULL && sr2 == NULL) return true;
    if(sr1 != NULL && sr2 != NULL)
        return (sr1->data == sr2->data &&
                areIdentical(sr1->left, sr2->left) &&
                areIdentical(sr1->right, sr2->right));
    return false;
}
```

^ | v · Reply · Share ›



Ankush · 2 years ago

In identicaltrees() ,can't we break as soon as third condition happens..

^ | v · Reply · Share ›



Ankit Sablok · 2 years ago

A much simpler solution would be to find the inorder traversals of the trees as a simple strcmp() function.

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

^ | v · Reply · Share ›



Gaurav Ramesh → Ankit Sablok · 5 months ago

this might work with just in-order if you add/append a dummy value to y
.. like 0 or something..

^ | v · Reply · Share ›



paramjeet parlokiya → Ankit Sablok · 9 months ago

smart n very tricky
#intelligent;)

^ | v · Reply · Share ›



Karanpreet → Ankit Sablok · a year ago

just the inorder traversal won't be sufficient.

If inorder traversal as well as preorder or postorder traversal strings are
are identical.

So inorder+preorder

OR

inorder+postorder

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

^ | v · Reply · Share ›



annonimus → Ankit Sablok · 2 years ago

No, There may be trees whose inorder are same but they are not ident
e.g:

1. a

b

c

2. c

b

a

Both are having inorder same but not identical.

^ | v • Reply • Share ›



Ankit Sablok → [anonimus](#) • a year ago

@anonimus and @Ramakrishna : Thanks for the clarification, the inorder and the preorder strings and compare if both of them that would suffice.

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

^ | v • Reply • Share ›



Krish → [Ankit Sablok](#) • 10 months ago

@Ankit : But the complexity may be more than the above

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

^ | v • Reply • Share ›



Ramakrishna → [anonimus](#) • a year ago

I agree with Anonymous...

I think trees with the same inorder traversal string need not be the tree just with inorder string as an input. You would also need to form the tree.

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

^ | v • Reply • Share ›



Tushar Roy • 3 years ago

This is much sleek code for identicalTrees method.

```
int identicalTrees(Node *node1,Node *node2)
{
    return ((!node1 && !node2) || (node1 && node2) &&
    node1->data == node2->data && sameTree(node1->left,node2->left)
    && sameTree(node1->right,node2->right));
}
```

^ | v • Reply • Share ›



Subscribe



Add Disqus to your site

