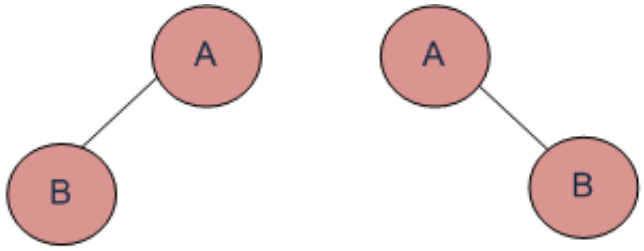
If you are given two traversal sequences, can you construct the binary tree?

It depends on what traversals are given. If one of the traversal methods is Inorder then the tree can be constructed, otherwise not.



Trees having Preorder, Postorder and Level-Order and traversals

Therefore, following combination can uniquely identify a tree.

Inorder and Preorder.
Inorder and Postorder.
Inorder and Level-order.

And following do not.
Postorder and Preorder.
Preorder and Level-order.
Postorder and Level-order.

For example, Preorder, Level-order and Postorder traversals are same for the trees given in above diagram.

Preorder Traversal = AB
Postorder Traversal = BA
Level-Order Traversal = AB

So, even if three of them (Pre, Post and Level) are given, the tree can not be constructed.

44 Comments Category: Trees Tags: Binary Tree, Tree Traveral

Related Posts:

- Check sum of Covered and Uncovered nodes of Binary Tree
- Lowest Common Ancestor in a Binary Tree | Set 2 (Using Parent Pointer)
- Construct a Binary Search Tree from given postorder
- BFS vs DFS for Binary Tree
- Maximum difference between node and its ancestor in Binary Tree
- Inorder Non-threaded Binary Tree Traversal without Recursion or Stack
- Check if leaf traversal of two Binary Trees is same?
- Closest leaf to a given node in Binary Tree

(Login to Rate and Mark)

1.7 Average Difficulty : **1.7/5.0**
Based on **24** vote(s)

Add to TODO List
Mark as DONE

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

44 Comments GeeksforGeeks 1 Login

Recommend 5 Share Sort by Newest

Join the discussion…

Satish Kumar • 2 months ago

tree construction when inoder and preorder are given

using recursion

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

• Reply • Share ›

Praveen Yadav • 6 months ago

Answer to all three types:

```
#include <iostream>

#include <string>

#include <queue>

using namespace std;

typedef struct node{

int data;

node *left;

node *right;

}node;

node *newNode(int new_data)
```

see more

1 • Reply • Share ›

Arun Mittal • 8 months ago

// when inorder and preorder are given
// please correct me if any mistake

```
#include<stdio.h>

#include<stdlib.h>

typedef struct node

{

int data;
```

int data;

struct node *left;

struct node *right;

}node;

int he,n;

---

see more

• Reply • Share ›

Rishu Agrawal • 10 months ago

Point to be noted -
given a Preorder & Postorder sequence we always cannot uniquely identify the tree.

Example -
consider two different trees
TREE 1
root=a;
root->left=b;
root->left->right=c;

Tree 2
root=a;
root->right=b;
root->right->left=c;

both the trees are different, but have same Preorder and Postorder seq.
Pre - a b c
Post - c b a

4 • Reply • Share ›

Arun Mittal ➚ Rishu Agrawal • 8 months ago

there must be inorder traversal
// make me correct if any mistake

• Reply • Share ›

Rishu Agrawal • 10 months ago

A SPECIAL CASE :
>> If each internal nodes has two children then the tree can be determined from Preorder and Postorder.
Example -
Preorder : a b c d f g e

Postorder : c f g b d e a

explanation -
From preorder we can determine 'a' is the root. In postorder seq. 'a' should be at the end and 'e' is before 'a', hence 'e' has to be right child of a (as each node has two child).
now, remove 'a' and 'e' from both the sequence and follow Recursively.

6 • Reply • Share ›

Guest • a year ago

```
void printpath(int *path,int len)
{
for(int i=0;i<len;i++) {="" printf("%d="" ",path[i]);="" }="" printf("\n");="" }="" void="" move(struct="" node*="" node,="" int="" index)="" {="" static="" int="" path[100];="" if(node="=NULL)" {="" return;="" }="" path[index++]="node->data;
move(node->left,index);
if(node->left==NULL && node->right==NULL)
printpath(path,index);
move(node->right,index);
}
```

• Reply • Share ›

sureshd.naik • 2 years ago

this sentence --> " For example, Preorder, Level-order and Postorder traversals are same for the trees given in above diagram " is wrong it seems, because post-order can differ from pre and level order traversals for the both trees given above

1 • Reply • Share ›

rohit ➦ sureshd.naik • a year ago

above statements talking about Preorder, Level-order and Postorder traversals of 2 given trees.
both have same preorder , postorder, & level-order traversals.

• Reply • Share ›

Pranav Kumar Jha ➦ sureshd.naik • a year ago

I see nothing wrong with the example. Moreover, what do you mean "....can differ..." ? Either it does differ or doesn't differ at all. :/

7 • Reply • Share ›

Jaguar • 2 years ago

If we are given preorder and postorder traversal and we are also given that the tree is a complete binary tree (In which every leaf is at same level) then we can determine the tree.

21 • Reply • Share ›

thephenom ➦ Jaguar • a year ago

Why do you even need both in that case? You can construct the tree with just preorder or postorder if you know that it is a complete binary tree.

• Reply • Share ›

Heta Vaishnani ➚ thephenom • a year ago

hoe can we construct complete binary tree using only pre/post?

• Reply • Share ›

Jaguar ➚ Heta Vaishnani • a year ago

let the pre-order traversal is 'abc', then definitely 'a' is the root, 'b' is the left child and 'c' is the right child. and we can construct a tree. As the tree is full and we know that total number of nodes in full binary search tree are (2^n - 1), where 'n' are number of levels. then if the pre-order traversal is 'a bcd efg', then 'a' is the node and 'bcd' corresponds to it's left child and 'efg' corresponds to it's right child and using the base case (for 3 nodes), we can construct the tree recursively.
Hope I am not wrong and explained it well. :)

2 • Reply • Share ›

Heta Vaishnani ➚ Jaguar • a year ago

i guess we are talking about complete binary tree not full binary tree..
thanx

1 • Reply • Share ›

Jaguar ➚ Heta Vaishnani • a year ago

well, this might be ambiguous because I am talking about a tree in which 'every leaf is at same level'. I have edit the main thread.

• Reply • Share ›

Jaguar ➚ thephenom • a year ago

And thanks for pointing this out. :)

• Reply • Share ›

Jaguar ➚ thephenom • a year ago

well, I did not give it a thought when I wrote this comment. Actually, YES, we can find a tree from either of it's pre-order or post-order traversal, if it is a full binary search tree.

• Reply • Share ›

sudhir miglani ➚ Jaguar • a year ago

Only if its binary "Search" tree. If it is only binary tree or even full

binary tree, we need two traversals(including inorder, as explained above).

Correct me if i am wrong :)

• Reply • Share ›

Jaguar sudhir miglani • a year ago

I guess we can determine the tree, even if we are given a "binary tree". We just need to know the nodes. I explained it in earlier comments of Heta Vaishnani comment. Someone correct me if I am wrong. :)

• Reply • Share ›

sudhir miglani Jaguar • a year ago

Oh Yes ! I missed it.. So let me clarify this... Your solution works only for "Full Binary Tree" and not just general binary tree.
Sounds fine?

1 • Reply • Share ›

Jaguar sudhir miglani • a year ago

Yeah, I guess so. :P

• Reply • Share ›

Amit Baghel • 2 years ago

visited!

2 • Reply • Share ›

smith • 2 years ago

when preorder and inorder is given:

code is

```
#include<stdio.h>

#include<malloc.h>

struct node

{

int data;

struct node *left;

struct node *right;
```

```
};

struct node *construct(int * int * struct node * int, int int int);
```

see more

2 • Reply • Share ›

Rahul Singh • 3 years ago

@geeksforgeeks team if we know any one of the traversal except inorder we can construct the unique BST . because we can ourself find the inorder traversal by sort the given traversal sequence.

7 • Reply • Share ›

Rahul Rahul Singh • 3 years ago

@Rahul Singh

We are talking about Binary tree not BST

28 • Reply • Share ›

trilok sharma • 3 years ago

```
#include
#include
#include
using namespace std;

struct node
{
int data;
node *left;
node *right;
};

node* Newnode(int data)
{
node * curr;
curr = (node *)malloc(sizeof(node));
curr->data = data;
curr->left = curr->right = NULL;
```

see more

4 • Reply • Share ›

Himanshu • 3 years ago

Here is a an algorithm from the URL http://stackoverflow.com/quest... that mentions how to construct a BST given inorder and level order.

f(inorder, levelorder):
if length(levelorder) == 0:
return None
root = levelorder[0]#set root to first element in levelorder
subIn1, subIn2 = partition(inorder, levelorder[0]) #partition inorder based on root
subLevel1 = extract(levelOrder, subIn1)#remove elements in level order not in subIn1
subLevel2 = extract(levelOrder, subIn2)#remove elements in level order not in subIn2
root->left = f(subIn1, subLevel1)
root->right = f(subIn2, subLevel2)
return root

1 • Reply • Share ›

Avinash • 4 years ago

```
 /* Paste your code here (You may delete these lines if not
writing code) */
Construct Tree from given Inorder and Preorder traversals
April 16, 2010
Let us consider the below traversals:

Inorder sequence: D B E A F C
Preorder sequence: A B D E C F

BuildTree(inorder[],preorder[],start,end)
{
        static int preindex=0;
        If start>end return NULL;

        struct node *newnode=new(preorder(preindex));
        preindex=preindex+1;

        If start==end return node;
```

see more

• Reply • Share ›

Avinash • 4 years ago

Let us consider the below traversals:

Inorder sequence: D B E A F C
Preorder sequence: A B D E C F

BuildTree(inorder[],preorder[],start,end)
{
static int preindex=0;

static int preindex=0;
If start>end return NULL;

struct node *newnode=new(preorder(preindex));
preindex=preindex+1;

If start==end return node;

int searchind=search(inorder,start,end,node->data);

node->left=BuildTree(inorder,preorder,start,searchind-1);
node->right=BuildTree(inorder,preorder,searchind+1,end);

---

see more

1 ▲ ▼ • Reply • Share ›

Devansh • 4 years ago
Inorder of a tree is must as from other traversal we are getting the root node of that tree and from inorder we get the child nodes which are in left subtree and right subtree as nodes which are in left subtree appears before root node in inorder traversal and the ones which are in right subtree appears after root.
3 ▲ ▼ • Reply • Share ›

An • 5 years ago
Hey can u xplain how to create a tree from inorder and level order !! I tried but can't figure out how we will get the knowledge of which child to attach to which root.
▲ ▼ • Reply • Share ›

Anand • 5 years ago
Given a post order and pre order traversal you can still construct a unique tree provide each internal nodes has two childern's
anandtechblog.blogspot.com/201...
▲ ▼ • Reply • Share ›

Anand • 5 years ago
anandtechblog.blogspot.com/201...
▲ ▼ • Reply • Share ›

manishj • 5 years ago
Let I(n) = i0,i1,i2,i3....in be elements of a inorder traversal of a binary tree.
Simillarly let Pre(n) = p0,p1,p2.....pn be the elements of a preorder traversal of a binary tree.

Now If we know that ik is root of binary tree , we can be sure that elements i0..ik-1 are in left subtree of tree and elements from 1k+1..in are in right-subtree rooted at ik(we can

prove this by contradiction).

Now , if we fix ik to be root. We inturn fix its left subtree (i0..ik-1), and its right right-subtree(ik+1...n), thus in essence we fix the tree (we can apply induction on n to mathematically prove this).

Inorder to fix the root , we can use either pre-order traversal.In any preorder traversal Pre(n) ,p0 is root of the the binarytree. We can also use Postorder traversal Post(n) say q0..qn in which qn is always the root of the tree.

So this proves that inorder combined with either post-order or preorder uniquely determine a tree.

1 ▲ | ▼ • Reply • Share ›

Himanshu Aggarwal • 5 years ago

Similarly, for trees like :

```
    A                  A
   /        and          \
  B                      B
```

both have preorder(and level-order) AB and postorder BA

▲ | ▼ • Reply • Share ›

Himanshu Aggarwal • 5 years ago

consider Two Binary Trees

For tree1 :
Root = A
Left Chid = B

Preorder: A,B
Postorder: B,A

and for tree 2:
Root = A
Right Child = B

Preorder: A,B
Postorder: B,A

For given preorder and postorder two different binary trees can be formed

▲ | ▼ • Reply • Share ›

Karthick ➦ Himanshu Aggarwal • 4 years ago

Forget about binary tree. What about a BST with just pre-order or a post-order?

Forget about binary tree. What about a BST with just pre-order or a post-order?

• Reply • Share ›

wgpshashank • 5 years ago

It is not very clear why InOrder is a must to recreate the tree.
Can you please provide more details regarding the same?

• Reply • Share ›

tech.login.id2 • 6 years ago

It is not very clear why InOrder is a must to recreate the tree.
Can you please provide more details regarding the same?

2 • Reply • Share ›

Rohini • 6 years ago

//preIndex is global

```
 node* BST::buildTree(int in[],int inStrt,int inEnd,int len,int pre[])
{
    if(preIndex >= len || inStrt > inEnd)
        return NULL;

    node *retNode = makeNode(pre[preIndex++]);

    if(inStrt == inEnd)
        return retNode;

    int inIndex = findNodeIn(in,inStrt, inEnd, retNode->data);
    retNode->left = buildTree(in, inStrt,inIndex-1,len,pre);
    retNode->right = buildTree(in,inIndex+1,inEnd,len,pre);

    return retNode;
```

see more

• Reply • Share ›

GeeksforGeeks ➚ Rohini • 6 years ago

@Rohini: Thanks for providing the code. We have published it here.

1 • Reply • Share ›

abhi ➚ GeeksforGeeks • 3 years ago

What about the case when we have duplicates in the Binary Tree ?

We can't identify the tree, right ?
suppose for the case : when all node values are 1's only

Suppose for the case : when all node values are 1's only.

• Reply • Share ›

vinni abhi • 2 years ago

We can create a new member in the node structure which is unique for each node in the tree. And use this unique representation of node to traverse the tree. It has got nothing to do with the "data" member, it 's just a representation.

• Reply • Share ›