Construct Tree from given Inorder and Preorder traversals

Let us consider the below traversals:

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

In a Preorder sequence, leftmost element is the root of the tree. So we know 'A' is root for given sequences. By searching 'A' in Inorder sequence, we can find out all elements on left side of 'A' are in left subtree and elements on right are in right subtree. So we know below structure now.

```
                 A
               /   \
             /       \
           D B E      F C
```

We recursively follow above steps and get the following tree.

```
         A
       /   \
     /       \
    B         C
   / \        /
 /     \    /
D       E  F
```

Algorithm: buildTree()
1) Pick an element from Preorder. Increment a Preorder Index Variable (preIndex in below code) to pick next element in next recursive call.
2) Create a new tree node tNode with the data as picked element.
3) Find the picked element's index in Inorder. Let the index be inIndex.
4) Call buildTree for elements before inIndex and make the built tree as left subtree of tNode.
5) Call buildTree for elements after inIndex and make the built tree as right subtree of tNode.
6) return tNode.

Thanks to Rohini and Tushar for suggesting the code.

```
/* program to construct tree using inorder and preorder traversals */
#include<stdio.h>
```

```
#include<stdlib.h>

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

/* Prototypes for utility functions */
int search(char arr[], int strt, int end, char value);
struct node* newNode(char data);

/* Recursive function to construct binary of size len from
   Inorder traversal in[] and Preorder traversal pre[].  Initial values
   of inStrt and inEnd should be 0 and len -1.  The function doesn't
   do any error checking for cases where inorder and preorder
   do not form a tree */
struct node* buildTree(char in[], char pre[], int inStrt, int inEnd)
{
  static int preIndex = 0;

  if(inStrt > inEnd)
     return NULL;

  /* Pick current node from Preorder traversal using preIndex
    and increment preIndex */
  struct node *tNode = newNode(pre[preIndex++]);

  /* If this node has no children then return */
  if(inStrt == inEnd)
    return tNode;

  /* Else find the index of this node in Inorder traversal */
  int inIndex = search(in, inStrt, inEnd, tNode->data);

  /* Using index in Inorder traversal, construct left and
     right subtress */
  tNode->left = buildTree(in, pre, inStrt, inIndex-1);
  tNode->right = buildTree(in, pre, inIndex+1, inEnd);

  return tNode;
}

/* UTILITY FUNCTIONS */
/* Function to find index of value in arr[start...end]
   The function assumes that value is present in in[] */
int search(char arr[], int strt, int end, char value)
{
  int i;
  for(i = strt; i <= end; i++)
  {
    if(arr[i] == value)
      return i;
  }
}

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

  return(node);
}

/* This funtcion is here just to test buildTree() */
void printInorder(struct node* node)
{
  if (node == NULL)
     return;

  /* first recur on left child */
  printInorder(node->left);

  /* then print the data of node */
  printf("%c ", node->data);

  /* now recur on right child */
  printInorder(node->right);
}

/* Driver program to test above functions */
int main()
{
  char in[] = {'D', 'B', 'E', 'A', 'F', 'C'};
  char pre[] = {'A', 'B', 'D', 'E', 'C', 'F'};
  int len = sizeof(in)/sizeof(in[0]);
  struct node *root = buildTree(in, pre, 0, len - 1);

  /* Let us test the built tree by printing Insorder traversal */
  printf("\n Inorder traversal of the constructed tree is \n");
  printInorder(root);
  getchar();
}
```

Run on IDE

Time Complexity: O(n^2). Worst case occurs when tree is left skewed. Example Preorder and Inorder traversals for worst case are {A, B, C, D} and {D, C, B, A}.

Please write comments if you find any bug in above codes/algorithms, or find other ways to solve the same problem.

118 Comments Category: Trees Tags: Inorder Traversal, Preorder Traversal, Tree Traveral

Related Posts:

- Find all possible binary trees with given Inorder Traversal
- Find LCA in Binary Tree using RMQ
- Find multiplication of sums of data of leaves at same levels
- 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

(Login to Rate and Mark)

3 Average Difficulty : **3/5.0**
Based on **4** vote(s)

Add to TODO List

Mark as DONE

Like Share 51 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.

118 Comments GeeksforGeeks 1 Login

Recommend 5 Share Sort by Newest

Join the discussion…

Stunner runner • 2 months ago

can someone implement this in python code plz

Reply • Share ›

ravinder • 3 months ago

Thanks a lot ot helped me a lot.

Reply • Share ›

Jayesh • 4 months ago

Java Implementation:

http://javabypatel.blogspot.in...

1 • Reply • Share ›

Rudolf Eremyan Jayesh • 3 months ago

Thnx a lot!!))))))))))))))))))

Reply • Share ›

priya • 5 months ago

how to do it with given inorder and postorder traversals?

Reply • Share ›

Jayesh priya • 4 months ago

Refer: http://javabypatel.blogspot.in...

• Reply • Share ›

Sagar Kulkarni priya • 5 months ago

It is very easy and solution is similar.
Instead of preIndex, you use postIndex starting from last element of Post-Order.
And it should be decreasing in each recursive call.
Also order of recursion is slightly changed :
For Pre-Order it was :

```
tNode->left = buildTree(in, pre, inStrt, inIndex-1);
tNode->right = buildTree(in, pre, inIndex+1, inEnd);
```

For Post-Order it will be :

```
tNode->right = buildTree(in, post, inIndex+1, inEnd);

tNode->left = buildTree(in, post, inStrt, inIndex-1);
```

I would suggest you try doing the program.

3 • Reply • Share ›

Shitiz Bhutani • 5 months ago

The Clean Solution-->

```
#include<bits stdc++.h="">

using namespace std;

struct Node

{

int data;

struct Node *left;

struct Node *right;

};

struct Node *newNode(int data)

{
```

see more

• Reply • Share ›

allahu akbar Shitiz Bhutani • 5 months ago

There might be a reason that 'shit' is in your name...

• Reply • Share ›

Shitiz Bhutani ↪ allahu akbar • 4 months ago

You are acting like a shit now. But who cares u asshole.

• Reply • Share ›

Abhilash • 6 months ago

Anyone know iterative solution?

• Reply • Share ›

Karan Kapoor ↪ Abhilash • 6 months ago

You can maintain a struct of Node and do it via iteration.
Iteration is a little non intuitive though but since all recursive paradigms can be represented via stacks(as they are implemented in compiler that way) you can do it..
Just maintain a stack of states and push into it the structure node every time you recurse and you ll be done..
I would like to stick with recursion.. Its much more beautiful :)

• Reply • Share ›

geek_13 • 6 months ago

Searching is an O(logn) operation as the inorder array is sorted(binary search) and for n nodes it should be O(nlogn) and not O(n^2). Please correct if i am wrong.

1 • Reply • Share ›

Keshav Sharma ↪ geek_13 • 6 months ago

Inorder traversal is sorted for BST only. So, searching is not O(log n) in every case.

3 • Reply • Share ›

geek_13 ↪ Keshav Sharma • 6 months ago

Thanks Keshav. I forgot that its a binary tree

• Reply • Share ›

geek_13 • 6 months ago

Searching is a logn operation as the inorder arrray is sorted(binary search) and for n nodes the complexity should be nlogn and not n^2. Please correct if I am wrong.

• Reply • Share ›

dev21 • 6 months ago

Even if there are repeated elements. Can inorder and postorder uniquely identify a binary tree?

1 ^ | v • Reply • Share ›

Sreekar ➦ dev21 • 6 months ago

I don't think so. I have implemented an "iterative version" for the same. It has same issue.

^ | v • Reply • Share ›

sick_master • 6 months ago

A very simple implementation.->

```
struct node * fun( int s , int l , int in ) // s ->start index in arr[0]
{ // l -> last index in arr[0]
for ( int i = s ; i <= l ; ++i ) // in -> index of ele in arr[1] to
{ // search in arr[0].
// arr[0] ->stores inorder .
if ( arr[1][in] == arr[0][i] ) // arr[1] -> stores preorder
{
nd * tmp = newNode ( arr[1][in] ) ;
if( i > s )
tmp -> left = fun( s , i-1 , in+1 ) ;
if ( i < l )
tmp -> right = fun ( i+1 , l , in+i+1 ) ;
return tmp ;
}
}
}
```

call fun(0 , n-1 , 0) from main() // n ->no of ele..

fun returns pointer to the subtree whose root is arr[1][in] and all its element lie in inorder arr[0][s] to arr[0][l].

^ | v • Reply • Share ›

techopinio • 6 months ago

It's not working for repeated elements

2 ^ | v • Reply • Share ›

Prince Bharti • 6 months ago

java solution (please go through the sol and let me know if there any problem)...

```
class sol{

int search(int[] arr,int key){
int i;
```

for(i=0;i<=arr.length;i++){
if(arr[i]==key)

break;
}
return i;
}

static int track=0;

node ctree(int[]inorder,int[] preorder, int instart,int inend){

int pos=search(inorder,preorder[track]);

see more

• Reply • Share ›

Holden • 7 months ago

what we can use instead of "static" in Java? It won't allow me to put 'static', and even final, to make "preIndex" unchangeable during each recursive call?
I wrote this in Java in O(nlgn):
http://ideone.com/dm9pgV

any comments?

• Reply • Share ›

Prince Bharti → Holden • 6 months ago

refer to my code above..

1 • Reply • Share ›

Balasaheb Dabhade → Holden • 7 months ago

Check now :
public class Method2 {

public static Node buildTree(int[] inorder, int[] preorder) {

return buildTree(inorder, 0, inorder.length - 1, preorder, 0,
preorder.length - 1);
}

static class Node {
int data;

public Node(int data) {
this.data = data;
left = null;

```
right = null;
}

Node right;
```

see more

• Reply • Share ›

ramesh kanth • 7 months ago

How It will work Repeated Element...

• Reply • Share ›

Saurabh • 7 months ago

Can anybody please check my code and see whats the problem I mean it is working for some half of the prorder array and not for the rest.

Thanks in advance.
Below is the code.

https://ideone.com/oLZkUU.

• Reply • Share ›

HariIITR • 8 months ago

@GeeksforGeeks, Since there is no proper article on "Construction of tree using in order and post order traversal" like we have for others, therefore I've put a nominal effort to write the code for this.
Here we go.........

http://ideone.com/nz9kMw

Plz comment in case of any error................

4 • Reply • Share ›

Baggy • 8 months ago

I printed the nodes in postorder and also preorder after building the tree. Both the ans are wrong.

Inorder traversal of the constructed tree is
D B E A F C
Postorder traversal of the constructed tree is
D B E F C A (wrong)
Preorder traversal of the constructed tree is
A D B E F C (wrong)

• Reply • Share ›

Varun Sagar ➦ Baggy • 7 months ago

fixed

• Reply • Share ›

Aryan Parker • 9 months ago

In the above code if we change the order of the recursive calls to construct the tree for left and right subtrees, the output tree changes. why is it so.
I mean changing the order from
tNode->left = buildTree(in, pre, inStrt, inIndex-1);
tNode->right = buildTree(in, pre, inIndex+1, inEnd);

to
tNode->right = buildTree(in, pre, inIndex+1, inEnd);
tNode->left = buildTree(in, pre, inStrt, inIndex-1);

• Reply • Share ›

puneet ➦ Aryan Parker • 9 months ago

because preindex is a static variable, if you change the order.... it'll construct right subtree from left subtree nodes given in preorder traversal array

1 • Reply • Share ›

Born Actor • a year ago

http://ideone.com/fvWfri

• Reply • Share ›

meelogan • a year ago

blah:

what if there are duplicates in the binary tree

• Reply • Share ›

thodde390 • a year ago

What if you wanted to do it using in-order and post-order instead of pre-order? How would that change the buildTree function?

1 • Reply • Share ›

blah • a year ago

what if there are duplicates in the binary tree

1 • Reply • Share ›

Abhinav Bhardwaj • a year ago

http://ideone.com/i2nFdK

• Reply • Share ›

zxcve • a year ago

Using HashMultiMap for fast search to reduce time to 0(n) in average.
Worst case will be when entire tree is of same value.
node * inorder_preorder(int *cur, unordered_multimap<int, int=""> *umap, int *pre, int start, int end)

{

node *temp;

int center = 0;

if (start > end)

return NULL;

temp = new node;

temp->data = pre[*cur];

auto it = umap->equal_range(pre[*cur]);

see more

• Reply • Share ›

kaushik Lele • a year ago

My java implementation shared at
http://ideone.com/KMLKHJ
http://kaushik-lele-algos-data...

2 • Reply • Share ›

info12345678 • a year ago

thanks

• Reply • Share ›

Vikas • a year ago

Consider this:
Inorder: D B E A C F
PreOrder: A B D E C F

The result for the Inorder traversal is: D B E A F C

4 • Reply • Share ›

helper • a year ago

my code for this question :D http://ideone.com/5acaGU

• Reply • Share ›

sree_ec • a year ago

O(n) - Going through each node only once.

tree_int* convertArrayToTree(int in[],int pre[],int instrt,int inlen,int *prestrt)
{

int temp;
int i,k;
tree_int* mytree=NULL;
if(inlen == 0 || (instrt >=inlen))
return NULL;

temp = pre[*prestrt];

(*prestrt)++;

for(i=instrt;i<inlen;i++) {="" if(temp="=" in[i])="" {="" mytree="(tree_int*)newNode(temp);"
mytree-="">left = convertArrayToTree(in,pre,instrt,i,prestrt);
mytree->right = convertArrayToTree(in,pre,i+1,inlen,prestrt);

return mytree;

}
}
return NULL;

}

• Reply • Share ›

sree_ec • a year ago

tree_int* convertArrayToTree(int in[],int pre[],int instrt,int inlen,int *prestrt)
{

int temp;
int i,k;
tree_int* mytree=NULL;
if(inlen == 0 || (instrt >=inlen))
return NULL;

temp = pre[*prestrt];

(*prestrt)++;

for(i=instrt;i<inlen;i++) {="" if(temp="=" in[i])="" {="" mytree="(tree_int*)newNode(temp);"
mytree-="">left = convertArrayToTree(in,pre,instrt,i,prestrt);

```
mytree->right = convertArrayToTree(in,pre,i+1,inlen,prestrt);

return mytree;

}
}
return NULL;

}
```

• Reply • Share ›

ryan • a year ago

```
nodeptr tree(char* inorder,char*preorder,int strt,int end)
{
if(end-strt==-1)
return NULL;
static int no;
nodeptr node=getnode();
node->data=preorder[no];
int i=0;
for(i=strt;i<=end;i++)
{
if(preorder[no]==inorder[i])
break;
}
no++;
node->left=tree(inorder,preorder,strt,i-1);
node->right=tree(inorder,preorder,i+1,end);
return node;
}
```

1 • Reply • Share ›

Ekta Goel • a year ago

Cannot we use binary search for finding the position of key in array storing the inorder traversal? Will not that reduce the complexity to O(nlogn)?

• Reply • Share ›

DS+Algo → Ekta Goel • a year ago

Remember, binary search always works with sorted array

1 • Reply • Share ›

Ekta Goel → DS+Algo • a year ago

Inorder traversal of the tree is always sorted and hence binary search can be used.

• Reply • Share ›

DS+Algo Ekta Goel • a year ago

that is for binary search tree

2 • Reply • Share ›

Ekta Goel DS+Algo • a year ago

Oh.. i mistakenly took the tree to be BST but actually it is Binary. Thanks for pointing this out.

• Reply • Share ›

Saurabh • a year ago

A simple approach:

http://ideone.com/9VO1ai

• Reply • Share ›

Load more comments