

GeeksforGeeks

A computer science portal for geeks

Placements

Practice

GATE CS

IDE

Q&A

GeeksQuiz

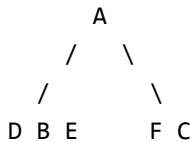
Construct Tree from given Inorder and Preorder traversals

Let us consider the below traversals:

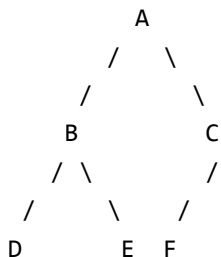
Inorder sequence: D B E A F C

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.



We recursively follow above steps and get the following tree.



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.

C

```
/* 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 subtree */
    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 function 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 Inorder traversal */
    printf("Inorder traversal of the constructed tree is \n");
    printInorder(root);
    getchar();
}

```

[Run on IDE](#)

Java

```

// Java program to construct a tree using inorder and preorder traversal

// A binary tree node
class Node {

    char data;
    Node left, right;

    Node(char item) {
        data = item;
        left = right = null;
    }
}

```

```

    }
}

class BinaryTree {

    static Node root;
    static int preIndex = 0;

    /* 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 */
    Node buildTree(char in[], char pre[], int inStrt, int inEnd) {

        if (inStrt > inEnd) {
            return null;
        }

        /* Pick current node from Preorder traversal using preIndex
        and increment preIndex */
        Node tNode = new Node(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;
            }
        }
        return i;
    }

    /* This function is here just to test buildTree() */
    void printInorder(Node node) {
        if (node == null) {
            return;
        }

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

        /* then print the data of node */
        System.out.print(node.data + " ");
    }
}

```

```

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

    // driver program to test above functions
    public static void main(String args[]) {
        BinaryTree tree = new BinaryTree();
        char in[] = new char[]{'D', 'B', 'E', 'A', 'F', 'C'};
        char pre[] = new char[]{'A', 'B', 'D', 'E', 'C', 'F'};
        int len = in.length;
        Node mynode = tree.buildTree(in, pre, 0, len - 1);

        // building the tree by printing inorder traversal
        System.out.println("Inorder traversal of constructed tree is : ");
        tree.printInorder(mynode);
    }
}

```

Run on IDE

Python

```

# Python program to construct tree using inorder and
# preorder traversals

# A binary tree node
class Node:

    # Constructor to create a new node
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

    """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 """
    def buildTree(inOrder, preOrder, inStrt, inEnd):

        if (inStrt > inEnd):
            return None

        # Pick current node from Preorder traversal using
        # preIndex and increment preIndex
        tNode = Node(preOrder[buildTree.preIndex])
        buildTree.preIndex += 1

        # If this node has no children then return
        if inStrt == inEnd :
            return tNode

        # Else find the index of this node in Inorder traversal
        inIndex = search(inOrder, inStrt, inEnd, tNode.data)

        # Using index in Inorder Traversal, construct left
        # and right subtrees
        tNode.left = buildTree(inOrder, preOrder, inStrt, inIndex-1)
        tNode.right = buildTree(inOrder, preOrder, 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 inOrder[]

def search(arr, start, end, value):
    for i in range(start, end+1):
        if arr[i] == value:
            return i

def printInorder(node):
    if node is None:
        return

    # first recur on left child
    printInorder(node.left)

    # then print the data of node
    print node.data,

    # now recur on right child
    printInorder(node.right)

# Driver program to test above function
inOrder = ['D', 'B', 'E', 'A', 'F', 'C']
preOrder = ['A', 'B', 'D', 'E', 'C', 'F']
# Static variable preIndex
buildTree.preIndex = 0
root = buildTree(inOrder, preOrder, 0, len(inOrder)-1)

# Let us test the build tree by printing Inorder traversal
print "Inorder traversal of the constructed tree is"
printInorder(root)

# This code is contributed by Nikhil Kumar Singh(nickzuck_007)
```

[Run on IDE](#)

Output :

```
Inorder traversal of constructed tree is :
D B E A F C
```

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.



126 Comments Category: Trees Tags: Inorder Traversal , Preorder Traversal , Tree Traversal

Related Posts:

- [Check if removing an edge can divide a Binary Tree in two halves](#)
- [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?](#)

([Login](#) to Rate and Mark)

3.1

Average Difficulty : 3.1/5.0
Based on 19 vote(s)

☐

Add to TODO List

☐

Mark as DONE

Like Share 52 people like this.

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

@geeksforgeeks, Some rights reserved

[Contact Us!](#)

[About Us!](#)

[Advertise with us!](#)