

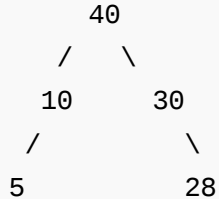
## Construct Special Binary Tree from given Inorder traversal

Given Inorder Traversal of a Special Binary Tree in which key of every node is greater than keys in left and right children, construct the Binary Tree and return root.

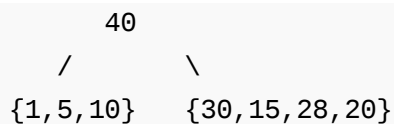
Examples:

Input: inorder[] = {5, 10, 40, 30, 28}

Output: root of following tree



The idea used in [Construction of Tree from given Inorder and Preorder traversals](#) can be used here. Let the given array is {1, 5, 10, 40, 30, 15, 28, 20}. The maximum element in given array must be root. The elements on left side of the maximum element are in left subtree and elements on right side are in right subtree.



We recursively follow above step for left and right subtrees, and finally get the following tree.



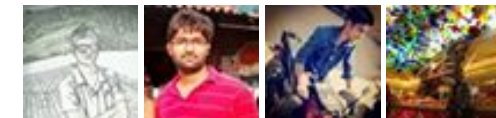
Google™ Custom Search



GeeksforGeeks



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



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

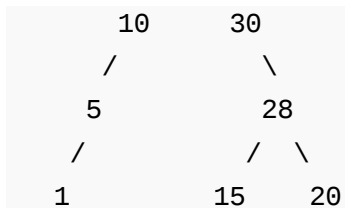
[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)



#### Algorithm: buildTree()

- 1) Find index of the maximum element in array. The maximum element must be root of Binary Tree.
- 2) Create a new tree node 'root' with the data as the maximum value found in step 1.
- 3) Call buildTree for elements before the maximum element and make the built tree as left subtree of 'root'.
- 5) Call buildTree for elements after the maximum element and make the built tree as right subtree of 'root'.
- 6) return 'root'.

**Implementation:** Following is C/C++ implementation of the above algorithm.

```

/* program to construct tree from inorder traversal */
#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;
};

/* Prototypes of a utility function to get the maximum
value in inorder[start..end] */
int max(int inorder[], int strt, int end);

/* A utility function to allocate memory for a node */
struct node* newNode(int data);

/* Recursive function to construct binary of size len from
Inorder traversal inorder[]. Initial values of start and end
should be 0 and len -1. */
struct node* buildTree (int inorder[], int start, int end)
  
```



## 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

```

{
    if (start > end)
        return NULL;

    /* Find index of the maximum element from Binary Tree */
    int i = max (inorder, start, end);

    /* Pick the maximum value and make it root */
    struct node *root = newNode(inorder[i]);

    /* If this is the only element in inorder[start..end],
       then return it */
    if (start == end)
        return root;

    /* Using index in Inorder traversal, construct left and
       right subtress */
    root->left = buildTree (inorder, start, i-1);
    root->right = buildTree (inorder, i+1, end);

    return root;
}

/* UTILITY FUNCTIONS */
/* Function to find index of the maximum value in arr[start...end] */
int max (int arr[], int strt, int end)
{
    int i, max = arr[strt], maxind = strt;
    for(i = strt+1; i <= end; i++)
    {
        if(arr[i] > max)
        {
            max = arr[i];
            maxind = i;
        }
    }
    return maxind;
}

/* 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;
}

/* 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("%d ", node->data);

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

/* Driver program to test above functions */
int main()
{
    /* Assume that inorder traversal of following tree is given
        40
       /  \
      10   30
     /      \
    5         28 */

    int inorder[] = {5, 10, 40, 30, 28};
    int len = sizeof(inorder)/sizeof(inorder[0]);
    struct node *root = buildTree(inorder, 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);
    return 0;
}

```

Output:

```

Inorder traversal of the constructed tree is
5 10 40 30 28

```

Time Complexity:  $O(n^2)$

## Recent Comments

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

[Backtracking | Set 7 \(Sudoku\)](#) · 37 minutes ago

**RVM** Can someone please elaborate this Qs from above...

[Flipkart Interview | Set 6](#) · 57 minutes ago

**Vishal Gupta** I talked about as an Interviewer in general,...

[Software Engineering Lab, Samsung Interview | Set 2](#) · 57 minutes 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

Neha I think that is what it should return as, in...

Find depth of the deepest odd level leaf node · 3 hours ago

AdChoices 

[► Binary Tree](#)

[► Java Tree](#)

[► Java Source Code](#)

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above



AdChoices

[▶ Tree Root](#)

[▶ Java Array](#)

[▶ Tree View](#)

AdChoices

[▶ Node](#)

[▶ In Memory Tree](#)

[▶ Root Size](#)

## 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 \(Delete\)](#)
- [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](#)



0

Tweet

0



0

Writing code in comment? Please use [ideone.com](#) and share the link here.

38 Comments

GeeksforGeeks



Join the discussion...



**prashant** • 10 hours ago

searching max which requires  $O(n)$  can be optimized with RMQ algorithm  
construct segment tree for the given traversal and then find max index in  $O(\log t(N)=2t(n/2)+O(\log n))$

^ | v • Reply • Share ›



**prashant** • 10 hours ago

$T(N)=2T(N/2)+O(N)$

its complexity will be  $O(n \log n)$  using masters theorem  
but for the skewed tree worst case analysis comes with  $O(n^2)$   
note it is not max heap as it is not complete binary tree

^ | v • Reply • Share ›



**prashant** • 10 hours ago

```
#include<iostream>
```

```
using namespace std;
```

```
struct tnode
```

```
{
```

```
tnode* lchild;
```

```
int data;
```

```
tnode* rchild;
```

```
tnode(int d)
```

```
{
```

```
lchild=NULL;
```

```
data=d;
```

```
rchild=NULL;
```

```
,  
};  
int search(int in[],int low,int high)  
{  
int max=0,loc;  
for(int i=low;i<=high;i++)
```

[see more](#)

^ | v • Reply • Share ›



**prashant** • a day ago

search for maximum key and recur into left and right subtree based on that inc

^ | v • Reply • Share ›



**sunil** • a month ago

```
/* If this is the only element in inorder[start..end],  
then return it */  
if (start == end)  
return root;
```

is not required, the condition at the start of the function is sufficient.

^ | v • Reply • Share ›



**alveko** • 11 months ago

The proper name for this "Special Binary Tree" is Cartesian tree which can be

^ | v • Reply • Share ›



**AMIT** → alveko • 10 months ago

I already posted a cmnt with an algo in o(n)  
time and o(1) extra space  
see my comment with name ammy

/\* Paste your code here (You may **delete** these lines **if not** wr

I

^ | v • Reply • Share ›



**alexander.korobeynikov** → AMIT • 10 months ago

It does not seem to be  $O(1)$  extra space

^ | v • Reply • Share ›



**AMIT** → alexander.korobeynikov • 10 months ago

only recursion stack is used..if you consider it too, its  $O(1)$

^ | v • Reply • Share ›



**uva** • 11 months ago

RMQ can make it  $O(n \lg n)$

^ | v • Reply • Share ›



**alveko** → uva • 11 months ago

Theoretically, RMQ can make it  $O(n)$

^ | v • Reply • Share ›



**Upendra** → alveko • 11 months ago

@alvenko how you can say that RMQ can make this  $O(n)$  because finding max element in a range and finding max using Range Maximum Query is  $O(n \lg n)$ ;

If you construct RMQ such a way that finding max can be done in  $O(1)$

Confirm me if you are thinking to implement RMQ such that finding max can be done in  $O(1)$

Happy Coding!!!

^ | v • Reply • Share ›



**alveko** → Upendra • 11 months ago



RMQ can be solved with sparse tables in  $O(1)$  with  $O(N)$

Have a look at this paper:

<http://www.ics.uci.edu/~eppste...>

And this page:

<http://www.topcoder.com/tc?d1=...>

Very briefly: a generic RMQ can be linear-time-reduced  
be linear-time-reduced back to a specific case of RMQ  
< $o(n)$ ,  $o(1)$ ="">.

So, yes, the coefficient in  $O(N)$  is probably high, bad in theory it works :)

^ | v • Reply • Share ›



**Upendra** → alveko • 11 months ago

Thanks men I got it.

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

^ | v • Reply • Share ›



**alveko** → alveko • 11 months ago

... which can be solved in  $O(1)$  with  $O(N)$  preprocessing

^ | v • Reply • Share ›



**ammy** • a year ago

this can be done in  $o(n)$

no need to search for largest value

let  $a[1..n]$  be the given array

if  $a[i] > a[i-1]$

then node with value  $a[i]$  will be the parent of node with value  $a[i-1]$  and  $a[i-1]$  its  
else  
node with value  $a[i]$  will be in the right subtree of node with value  $a[i-1]$   
so, by maintaining a max value a node can attend, this can be done in  $O(n)$   
code-- this function returns the head of BST

```
typedef struct node2 *bst;
typedef struct node2{
    int data;
    bst left, right;
}bintree;

bst special(int a[], int *start, int end, int min, int max)
{
```

[see more](#)

^ | v • Reply • Share ›



**xiaoc10** → ammy • a year ago

What're 'min' and 'max' used for?

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

^ | v • Reply • Share ›



**ammy** → xiaoc10 • a year ago

@xiaoc10--to find where to stop building a subtree. suppose we

50

/

10

and then,,we see a value 20 which is greater than

10.so,,20 cant be inserted as a child of 10,i.e max value is the \

max value can be 9.  
min has no use. ignore it  
^ | v • Reply • Share ›



**ammy** → ammy • a year ago

bst head=NULL;

^ | v • Reply • Share ›



**Ashish** • 2 years ago

Can we do this divide and conquer:  
think about three nodes 1,2,3 in inorder traversal.  
They can be put into 4 combinations LIKE:  
[2[1,3]]  
[1R[2R[3]]]  
[3L[2L[1]]]  
[3L[1R[2]]]

RETURN THE ROOT FROM THIS FUNTION.

SO DIVIDE THE ARRAY IN TWO PARTS: 1 AND 3 AND 2 IS THE PART THA

^ | v • Reply • Share ›



**pb** • 2 years ago

even a heap satisfy this property...this kind of tree can be think of as a subset  
will take  $O(n \log n)$

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

^ | v • Reply • Share ›



**kartik** → pb • 2 years ago

It's other way. Heap can be said as a subset of this special tree. Also, |

^ | v • Reply • Share ›



Palash → kartik • a year ago

Check out the implementation of NlogN version below.

```
/*
int getNext(anode *sort, int index, int com, int n, int
{
    int i=index+1;
    if(com)
    {
        while(sort[index].index>sort[i].index &&
            if(i>=n || sort[i].index<start || sort[i]
            return i;
    }
    if(!com)
    {
        while(sort[index].index<sort[i].index &&
            if(i>=n || sort[i].index<start || sort[i]
            return i;
    }
}
```

see more

^ | v • Reply • Share ›



Palash → kartik • a year ago

It can be done in NlogN, by sorting the inorder array initially, also original inorder and then using it.

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

^ | v • Reply • Share ›



**Robin** · 2 years ago

Good

^ | v · Reply · Share ›



**Praveen** · 2 years ago

Excellent Post. It's an amazon interview question.

^ | v · Reply · Share ›



**Mukesh kumar Agrawal** · 2 years ago

I think its complexity should be  $O(n \cdot \log n)$  as follow:

$$T(n) = n + 2T(n/2)$$

On expanding it, u will get  $T(n) = O(n \cdot \log n)$ .

Please point me out if I am wrong.

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

^ | v · Reply · Share ›



**kartik** → Mukesh kumar Agrawal · 2 years ago

Consider the case of a skewed tree to see worst case complexity.

100

/

90

/

80

/

70

In worst case, time complexity  $T(n) = T(n-1) + cn$

^ | v · Reply · Share ›



**Mukesh** → kartik · 2 years ago

Thanks

It is not mentioned that it is a proper binary tree.

^ | v · Reply · Share ›



**hemant** · 2 years ago

Max heap

1 ^ | v · Reply · Share ›



**kartik** → hemant · 2 years ago

It is not max heap. It follows only one property of Max Heap. The other |  
except the last level and the last level should be filled only from left side

^ | v · Reply · Share ›



**rakesh** · 2 years ago

we can find the max element in array with  $\log(n)$  right?  
since the array is always increasing and then decreasing.  
so the overall complexity would be decreased.

^ | v · Reply · Share ›



**kartik** → rakesh · 2 years ago

It is not always first increasing and then decreasing.

^ | v · Reply · Share ›



**rakesh** · 2 years ago

we can find the max element in an array with  $\log(n)$  since the array and subarr  
then decreasing.that way we can decrease the total complexity.

^ | v · Reply · Share ›



**pkm** · 2 years ago



good to see this problem here. it's an Amazon interview question.

^ | v • Reply • Share ›



**akash bansal** • 2 years ago

why u r taking the largest element to be the root ? i mean there is no compulsi largest element."the maximum elememnt in the array must be root"?????

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

^ | v • Reply • Share ›



**GeeksforGeeks** → akash bansal • 2 years ago

@akash bansal: Please take a closer look at the problem statement. It key of every node is greater than keys in left and right children

^ | v • Reply • Share ›



**akash bansal** → GeeksforGeeks • 2 years ago

okk thanks my mistake i must have read the question thoroughl

^ | v • Reply • Share ›



Subscribe



Add Disqus to your site

@geeksforgeeks, **Some rights reserved**

**Contact Us!**

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team