



Introduction to Trees



Medium

Accuracy: 91.08%

Submissions: 3K+

Points: 4

Unlock your coding potential - Join our Hiring Coding Contest and land your dream job!



Given an integer i . Print the **maximum number of nodes** on level i of a binary tree.

Example 1:

Input: 5

Output: 16

Example 2:

Input: 1

Output: 1

Your Task:

```
1 // } Driver Code Ends
11 // User function Template for C++
12
13 class Solution {
14 public:
15     int countNodes(int i) {
16         // your code here
17         if(i==1) return 1;
18         return pow(2,i-1);
19     }
20 };
21 // } Driver Code Ends
```

[Custom Input](#)

Compile & Run

Submit



Binary Tree Representation

Easy

Accuracy: 91.09%

Submissions: 2K+

Points: 2

Unlock your coding potential - Join our Hiring Coding Contest and land your dream job!

You are given an array nodes. It contains 7 integers, which represents the value of nodes of the binary tree in level order traversal. You are also given a root of the tree which has a value equal to nodes[0].

Your task to construct a binary tree by creating nodes for the remaining 6 nodes.

Example:

Input:

nodes = [1 2 3 4 5 6 7]

Output:

```
      1
     / \
    2   3
   / \ / \
  4 5 6 7
```

```
1 // } Driver Code Ends
35 //User function Template for C++
36
37
38 class Solution{
39 public:
40
41 void create_tree(node* root0, vector<int> &arr){
42     //Your code goes here
43     queue<node*> q;
44     q.push(root0);
45
46     int i = 1;
47     int n = arr.size();
48
49     while(i < n){
50         node* root = q.front();
51         q.pop();
52
53         node* left = newNode(arr[i]);
54         root->left = left;
55         i++;
56         q.push(left);
57
58         node* right = newNode(arr[i]);
59         root->right = right;
60         i++;
61         q.push(right);
62
63     }
64 }
65
66 };
67 // } Driver Code Ends
```



Preorder Traversal



Basic

Accuracy: 62.73%

Submissions: 121K+

Points: 1

Unlock your coding potential - Join our Hiring Coding Contest and land your dream job!

Given a binary tree, find its preorder traversal.

Example 1:

Input:

```
    1
   /
  4
 / \
4   2
```

Output: 1 4 4 2

Example 2:

Input:

6

```
1 // } Driver Code Ends
112
113
114 //Function to return a list containing the preorder traversal of the tree.
115
116 vector<int> preorder(Node* root)
117 {
118     // Your code here
119     vector<int> res;
120     if(root != NULL){
121         cout << (root->data) << " ";
122         preorder(root->left);
123         preorder(root->right);
124         return res;
125     }
126 }
```



[Custom Input](#)

Compile & Run

Submit

Inorder Traversal

Basic Accuracy: 67.15% Submissions: 123K+ Points: 1

Unlock your coding potential - Join our Hiring Coding Contest and land your dream job!

Given a Binary Tree, find the In-Order Traversal of it.

Example 1:

Input:

```
    1
   / \
  3   2
```

Output: 3 1 2

Example 2:

Input:

```
    10
   /  \
```

C++ (g++ 5.4)

Average Time: 15m

Start Timer

```
1 // } Driver Code Ends
19
20
21
22 class Solution {
23 public:
24     // Function to return a list containing the inorder traversal of the tree.
25
26
27     vector<int> inOrder(Node* root) {
28         // Your code here
29         vector<int> res;
30         if(root != NULL){
31
32             inOrder(root->left);
33             cout << (root->data) << " ";
34             inOrder(root->right);
35         }
36         return res;
37     }
38 }
39 };
40 // } Driver Code Ends
```



Custom Input

Compile & Run

Submit



Postorder Traversal



Basic

Accuracy: 74.96%

Submissions: 89K+

Points: 1

Unlock your coding potential - Join our Hiring Coding Contest and land your dream job!



Given a binary tree, find the Postorder Traversal of it.

For Example, the postorder traversal of the following tree is:

5 10 39 1

```
  1
 /  \
10   39
/
5
```

Example 1:

Input:

```
  19
 /  \
```

```
1 // } Driver Code Ends
113
114
115
116 //Function to return a list containing the postorder traversal of the tree.
117 vector<int> postOrder(Node* root)
118 {
119     // Your code here
120     vector<int> res;
121     if(root != NULL){
122
123         postOrder(root->left);
124         postOrder(root->right);
125         cout << (root->data) << " ";
126
127         return res;
128
129     }
130 }
131
```

