Domains Contests Rank Leaderboard Jobs nikban

All Contests > 30 Days of Code > Day 23: Review+Binary Trees!

Day 23: Review+Binary Trees!

by vatsalchanana

Problem Submissions Leaderboard Discussions

Welcome to Day 23! Review everything we've learned so far by making Tic Tac Toe from scratch with this video, or just jump right into the problem.

You are given a pointer *root* pointing to the root of a binary tree. You need to print the *level order* traversal of this tree. In level order traversal, we visit the nodes level by level from left to right.
For example:

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

For the above tree, the level order traversal is 3 -> 5 -> 2 -> 1 -> 4 -> 6.

HINT: A queue could be helpful.

The code for input/output is already handled in the editor. You only have to complete the function *levelOrder* given in the editor.

Good luck!

Input Format

First line contains *T*, the number of test cases. Next *T* lines contain an integer *data* to be added to the binary search tree.

Output Format

Print the values of the level order traversal separated by spaces.

Sample Input

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

Sample Output

```
3 2 5 1 4 7
```

Explanation

```
Level 1:        3
              /   \
Level 2:     2     5
            /     /\
Level 3:   1     4  7
```

We need to print the nodes level by level. We process each level from left to right.

Level Order Traversal: 3 -> 2 -> 5 -> 1 -> 4 -> 7

Submissions: 3106

Max Score: 100

Difficulty: Easy

More

Current Buffer (saved locally, editable)

C++

```
#include <iostream>
#include <cstddef>
#include <queue>
#include <string>
#include <cstdlib>

using namespace std;
class Node{
    public:
        int data;
        Node *left,*right;
        Node(int d){
            data=d;
            left=right=NULL;
        }
};
class Solution{
    public:
        Node* insert(Node* root, int data){
            if(root==NULL){
                return new Node(data);
            }
            else{
                Node* cur;
                if(data<=root->data){
                    cur=insert(root->left,data);
                    root->left=cur;
                }
                else{
                   cur=insert(root->right,data);
                   root->right=cur;
                 }
           return root;
           }
        }
```

```
    void levelOrder(Node * root){
      //Write your code here


    }
```

```
};//End of Solution
int main(){
    Solution myTree;
    Node* root=NULL;
    int T,data;
    cin>>T;
    while(T-->0){
        cin>>data;
        root= myTree.insert(root,data);
    }
    myTree.levelOrder(root);
    return 0;
}
```

Line: 7 Col: 1

Upload Code as File

□ Test against custom input

Run Code

Submit Code