Level Order Traversal of Binary Tree

Try to solve the Level Order Traversal of Binary Tree problem.

We'll cover the following

Statement

Given the root of a binary tree, display the values of its nodes while performing a level order traversal. Return the node values for all levels in a string separated by the character `:`. If the tree is empty, i.e., the number of nodes is 0, then return “None” as the output.

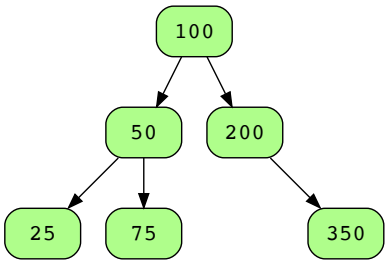
Constraints:

- The number of nodes in the tree is in the range $[0, 2000]$.
- $-1000 \leq$ `Node.data` ≤ 1000

Examples

Sample example 1

Input



Output

```
100 : 50, 200 : 25, 75, 350
```

Same-level nodes, i.e., 50 and 200, are separated by a comma.

1 of 2

Understand the problem

Let’s take a moment to make sure you’ve correctly understood the problem. The quiz below helps you check if you’re solving the correct problem:

Level Order Traversal of Binary Tree

1 What should be the output if the following tree is given as input?

A) 50, -50 : 200, 32, 12, -1

B) 50 : -50, 200 : 32 : 12 : -1

C) -1 : 12 : 32 : -50, 200 : 50

Submit Answer

Question 1 of 2
0 attempted

Reset Quiz

Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Drag and drop the cards to rearrange them in the correct sequence.

Declare two queues, `currentQueue` and `nextQueue`.

Push the root node to `currentQueue` and set the level to zero.

Dequeue the first element from `currentQueue` and push its children in `nextQueue`.

If `currentQueue` is empty, increase the level number and swap the two queues.

Repeat until `currentQueue` is empty.

Reset | Show Solution | Submit

Try it yourself

Implement your solution in the following coding playground.

> **Note:** The binary tree node's class has members `left` and `right` to store references to other nodes, along with the member `data` to hold the node's value.

Java

usercode > main.java

```java
// Definiton of a binary tree node class
// class TreeNode<T> {
//     T data;
//     TreeNode<T> left;
//     TreeNode<T> right;
//         this.data = data;
//         this.left = null;
//         this.right = null;
//     }
// }

import java.util.*;
import ds_v1.BinaryTree.TreeNode;

public class Main{
    public static String levelOrderTraversal(TreeNode<Integer> root) {

      // Write your code here
      return "";
  }
}
```

Powered by AI | Submit

Test Cases | Results

Case 1 | Case 2 | Case 3

Input #1

[100,50,200,25,75,300,10,350,15]

Level Order Traversal of Binary Tree