

Grokking the Coding Interview: Patterns for Coding Questions

57% completed



- Minimum Depth of a Binary Tree (easy)
- Level Order Successor (easy)
- Connect Level Order Siblings (medium)
- Problem Challenge 1
- Solution Review: Problem Challenge 1**
- Problem Challenge 2
- Solution Review: Problem Challenge 2

Pattern: Tree Depth First Search

- Introduction
- Binary Tree Path Sum (easy)
- All Paths for a Sum (medium)
- Sum of Path Numbers (medium)
- Path With Given Sequence (medium)
- Count Paths for a Sum (medium)
- Problem Challenge 1
- Solution Review: Problem Challenge 1
- Problem Challenge 2
- Solution Review: Problem Challenge 2

Pattern: Two Heaps

- Introduction
- Find the Median of a Number Stream (medium)
- Sliding Window Median (hard)
- Maximize Capital (hard)
- Problem Challenge 1
- Solution Review: Problem Challenge 1

Pattern: Subsets

- Introduction
- Subsets (easy)
- Subsets With Duplicates (easy)
- Permutations (medium)
- String Permutations by changing case (medium)
- Balanced Parentheses (hard)
- Unique Generalized Abbreviations (hard)
- Problem Challenge 1
- Solution Review: Problem Challenge 1
- Problem Challenge 2
- Solution Review: Problem Challenge 2
- Problem Challenge 3
- Solution Review: Problem Challenge 3

Pattern: Modified Binary Search

- Introduction
- Order-agnostic Binary Search

Solution Review: Problem Challenge 1

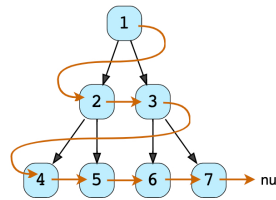
We'll cover the following

- Connect All Level Order Siblings (medium)
- Solution
- Code
 - Time complexity
 - Space complexity

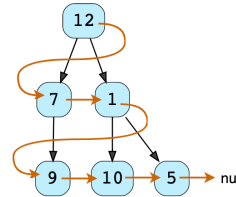
Connect All Level Order Siblings (medium)

Given a binary tree, connect each node with its level order successor. The last node of each level should point to the first node of the next level.

Example 1:



Example 2:



Solution

This problem follows the [Binary Tree Level Order Traversal](#) pattern. We can follow the same **BFS** approach. The only difference will be that while traversing we will remember (irrespective of the level) the previous node to connect it with the current node.

Code

Here is what our algorithm will look like; only the highlighted lines have changed:

```
1  const Deque = require('./collections/deque'); //http://www.collectionsjs.com
2
3  class TreeNode {
4    constructor(val) {
5      this.val = val;
6      this.left = null;
7      this.right = null;
8      this.next = null;
9    }
10
11
12    // tree traversal using 'next' pointer
13    print_tree() {
14      process.stdout.write("Traversal using 'next' pointer: ");
15      let current = this;
16      while (current !== null) {
17        process.stdout.write(`${current.val} `);
18        current = current.next;
19      }
20    }
21  }
22
23  function connect_all_siblings(root) {
24    if (root === null) {
25      return;
26    }
27
28    const queue = new Deque();
```

Explore

Tracks

My Courses

Edpresso

Refer a Friend

Create

(easy)

Ceiling of a Number (medium)

Next Letter (medium)

Number Range (medium)

Search in a Sorted Infinite Array (medium)

Minimum Difference Element (medium)

Bitonic Array Maximum (easy)

Problem Challenge 1

Solution Review: Problem Challenge 1

Problem Challenge 2

Solution Review: Problem Challenge 2

Problem Challenge 3

Solution Review: Problem Challenge 3

Pattern: Bitwise XOR

Introduction

Single Number (easy)

Two Single Numbers (medium)

Complement of Base 10 Number (medium)

Problem Challenge 1

Solution Review: Problem Challenge 1

Pattern: Top 'K' Elements

Close

Output

4.116s

Traversal using 'next' pointer: 12 7 1 9 10 5

Time complexity ⚡

The time complexity of the above algorithm is $O(N)$, where 'N' is the total number of nodes in the tree. This is due to the fact that we traverse each node once.

Space complexity ⚡

The space complexity of the above algorithm will be $O(N)$ which is required for the queue. Since we can have a maximum of $N/2$ nodes at any level (this could happen only at the lowest level), therefore we will need $O(N)$ space to store them in the queue.

Interviewing soon? We've partnered with [Hired](#) so that companies apply to you instead of you applying to them. [See how](#) 🗨

← Back

Next →

Problem Challenge 1

Problem Challenge 2

✓ Mark as Completed

🗨 Report an Issue

🗨 Ask a Question