

## Grokking the Coding Interview: Patterns for Coding Questions

58% 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

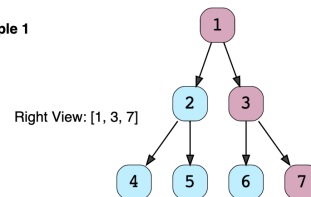
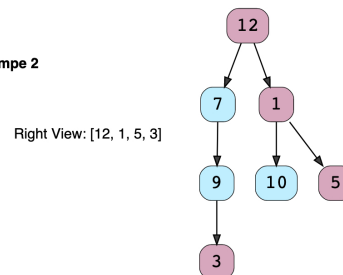
## Solution Review: Problem Challenge 2

### We'll cover the following

- Right View of a Binary Tree (easy)
- Solution
- Code
  - Time complexity
  - Space complexity
- Similar Questions

### Right View of a Binary Tree (easy)

Given a binary tree, return an array containing nodes in its right view. The right view of a binary tree is the set of **nodes visible when the tree is seen from the right side**.

**Example 1****Exampe 2**

### Solution

This problem follows the [Binary Tree Level Order Traversal](#) pattern. We can follow the same **BFS** approach. The only additional thing we will be do is to append the last node of each level to the result array.

### 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
4  class TreeNode {
5    constructor(val) {
6      this.val = val;
7      this.left = null;
8      this.right = null;
9    }
10 }
11
12
13 function tree_right_view(root) {
14   result = [];
15   if (root === null) {
16     return result;
17   }
18
19   const queue = new Deque();
20   queue.push(root);
21   while (queue.length > 0) {
22     const levelSize = queue.length;
23     for (i = 0; i < levelSize; i++) {
24       currentNode = queue.shift();
25       // if it is the last node of this level, add it to the result
26       if (i === levelSize - 1) {
27         result.push(currentNode);
28       }
29     }
30   }
31 }
```

Explore

Tracks

My Courses

Edpresso

Refer a Friend

Create

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

Introduction

Top 'K' Numbers (easy)

Kth Smallest Number (easy)

'K' Closest Points to the Origin (easy)

Connect Ropes (easy)

Top 'K' Frequent Numbers (medium)

Frequency Sort (medium)

Kth Largest Number in a Stream

Tree right view: 12 1 5 3

6.539s

**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  $O(N)$  as we need to return a list containing the level order traversal. We will also need  $O(N)$  space 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.

**Similar Questions**

**Problem 1:** Given a binary tree, return an array containing nodes in its left view. The left view of a binary tree is the set of nodes visible when the tree is seen from the left side.

**Solution:** We will be following a similar approach, but instead of appending the last element of each level we will be appending the first element of each level to the output array.

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 2

Introduction

Mark as Completed

Report an Issue

Ask a Question