

## Grokking the Coding Interview: Patterns for Coding Questions

55% completed



### Pattern: Tree Breadth

#### First Search

- Introduction
- Binary Tree Level Order Traversal (easy)
- Reverse Level Order Traversal (easy)
- Zigzag Traversal (medium)
- Level Averages in a Binary Tree (easy)
- 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

## Level Order Successor (easy)

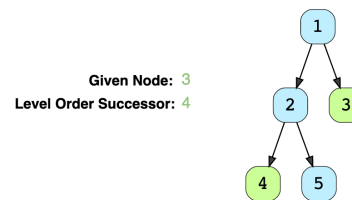
### We'll cover the following

- Problem Statement
- Try it yourself
- Solution
- Code
  - Time complexity
  - Space complexity

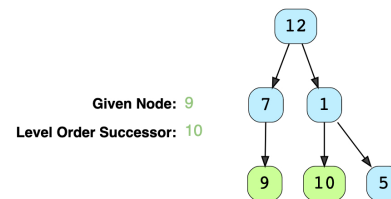
### Problem Statement

Given a binary tree and a node, find the level order successor of the given node in the tree. The level order successor is the node that appears right after the given node in the level order traversal.

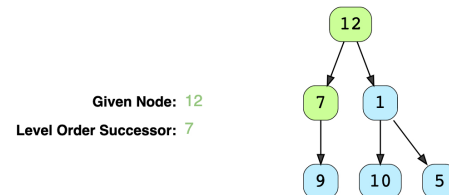
#### Example 1:



#### Example 2:



#### Example 3:



### Try it yourself

Try solving this question here:

[Java](#) [Python3](#) [JS](#) [C++](#)

```
2
3
4 constructor(val) {
5   this.val = val;
6   this.left = null;
7   this.right = null;
8 }
9
10
11 const find_successor = function(root, key) {
12   // TODO: Write your code here
13   return null;
14 };
15
16
17 var root = new TreeNode(12)
18 root.left = new TreeNode(7)
19 root.right = new TreeNode(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 (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

- 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 (medium)
- 'K' Closest Numbers (medium)
- Maximum Distinct Elements (medium)
- Sum of Elements (medium)
- Rearrange String (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: K-way merge

- Introduction
- Merge K Sorted Lists (medium)
- Kth Smallest Number in M Sorted Lists (Medium)
- Kth Smallest Number in a Sorted

```
20 root.left.left = new TreeNode(9)
21 root.right.left = new TreeNode(10)
22 root.right.right = new TreeNode(5)
23 result = find_successor(root, 12)
24 if (result != null)
25     console.log(result.val)
26 result = find_successor(root, 9)
27 if (result != null)
28     console.log(result.val)
29
```

RUN

SAVE

RESET



Close

✓ Succeeded

## Solution

This problem follows the [Binary Tree Level Order Traversal](#) pattern. We can follow the same **BFS** approach. The only difference will be that we will not keep track of all the levels. Instead we will keep inserting child nodes to the queue. As soon as we find the given node, we will return the next node from the queue as the level order successor.

## Code

Here is what our algorithm will look like; most of the changes are in the highlighted lines:

Java

Python3

C++

JS

```
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 function find_successor(root, key) {
13     if (root === null) {
14         return null;
15     }
16
17     const queue = new Deque();
18     queue.push(root);
19     while (queue.length > 0) {
20         currentNode = queue.shift();
21         // insert the children of current node in the queue
22         if (currentNode.left !== null) {
23             queue.push(currentNode.left);
24         }
25         if (currentNode.right !== null) {
26             queue.push(currentNode.right);
27         }
28         // break if we have found the key
```

RUN

SAVE

RESET



Close

Output

9.197s

7  
10

## 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 →

Minimum Depth of a Binary Tree (easy)

Connect Level Order Siblings (medium)

✓ Mark as Completed

Report an Issue Ask a Question

