

Grokking the Coding Interview: Patterns for Coding Questions

35% completed

(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 (easy)

Ceiling of a Number (medium)

Path With Given Sequence (medium)

We'll cover the following

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

Problem Statement

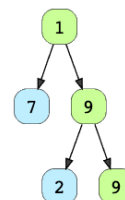
Given a binary tree and a number sequence, find if the sequence is present as a root-to-leaf path in the given tree.

Example 1:

Sequence: [1, 9, 9]

Output: true

Explanation: The tree has a path 1 -> 9 -> 9.

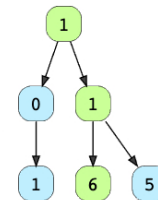


Example 2:

Sequence: [1, 0, 7]

Output: false

Explanation: The tree does not have a path 1 -> 0 -> 7.



Try it yourself

Try solving this question here:

JavaPython3JSC++

```
1 import java.util.*;
2
3 class TreeNode {
4     int val;
5     TreeNode left;
6     TreeNode right;
7
8     TreeNode(int x) {
9         val = x;
10    }
11 };
12
13 class PathWithGivenSequence {
14     public static boolean findPath(TreeNode root, int[] sequence) {
15         // TODO: Write your code here
16         return false;
17     }
18
19     public static void main(String[] args) {
20         TreeNode root = new TreeNode(1);
21         root.left = new TreeNode(0);
22         root.right = new TreeNode(1);
23         root.left.left = new TreeNode(1);
24         root.right.left = new TreeNode(6);
25         root.right.right = new TreeNode(5);
26
27         System.out.println("Tree has path sequence: " + PathWithGivenSequence.findPath(root, new int[] { 1, 0, 7 }));
28         System.out.println("Tree has path sequence: " + PathWithGivenSequence.findPath(root, new int[] { 1, 1, 9 }));
29     }
30 }
```

RUNSAVERESET

Solution

This problem follows the [Binary Tree Path Sum](#) pattern. We can follow the same **DFS** approach and additionally, track the element of the given sequence that we should match with the current node. Also, we can return **false** as soon as we find a mismatch between the sequence and the node value.

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

MW

Explore

Tracks

My Courses

Edpresso

Refer a Friend

Create

Code

Here is what our algorithm will look like:

JavaPython3C++JS

```
23 if (currentNode == null)
24     return false;
25
26 if (sequenceIndex >= sequence.length || currentNode.val != sequence[sequenceIndex])
27     return false;
28
29 // if the current node is a leaf, add it is the end of the sequence, we have found a path!
30 if (currentNode.left == null && currentNode.right == null && sequenceIndex == sequence.length - 1)
31     return true;
32
33 // recursively call to traverse the left and right sub-tree
34 // return true if any of the two recursive call return true
35 return findPathRecursive(currentNode.left, sequence, sequenceIndex + 1)
36     || findPathRecursive(currentNode.right, sequence, sequenceIndex + 1);
37 }
38
39 public static void main(String[] args) {
40     TreeNode root = new TreeNode(1);
41     root.left = new TreeNode(0);
42     root.right = new TreeNode(1);
43     root.left.left = new TreeNode(1);
44     root.right.left = new TreeNode(6);
45     root.right.right = new TreeNode(5);
46
47     System.out.println("Tree has path sequence: " + PathWithGivenSequence.findPath(root, new int[] { 1, 0, 1 }));
48     System.out.println("Tree has path sequence: " + PathWithGivenSequence.findPath(root, new int[] { 1, 1, 1 }));
49 }
50 }
```

RUNSAVERESET

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 $O(N)$ in the worst case. This space will be used to store the recursion stack. The worst case will happen when the given tree is a linked list (i.e., every node has only one child).

Interviewing soon? We've partnered with [Hired](#) so that companies apply to you, instead of the other way around. [See how](#)

← Back

Sum of Path Numbers (medium)

✓ MARK AS COMPLETED

Next →

Count Paths for a Sum (medium)