



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Experiment-2.1

**Student Name:** Himanshu

**Branch:** BE-CSE

**Semester:** 6<sup>th</sup>

**Subject Name:** Competitive Coding-II

**UID:** 20BCS7944

**Section/Group:** 905/A

**Date of Performance:** 30/03/2023

**Subject Code:** 20CSP-351

### 1. Aim:

To demonstrate the concept of Tree.

### 2. Objective:

- The objective is to build problem solving capability and to learn the basic concepts of Tree data structures.
- The implementation of Path Sum which shows and brushes up the concept of Tree.
- The implementation of Same Tree.

### 3. LeetCode code and output:

- **Path Sum -**

```
class Solution {
public boolean hasPathSum(TreeNode root, int sum) {
    return root == null ? false : DFS(root, 0, sum);
}
public boolean DFS(TreeNode node, int tmp, int sum){
    boolean flag = false;
    tmp += node.val;
    if(node.left == null && node.right == null){
        return tmp == sum;
    }
    if(node.left != null){
        flag = flag || DFS(node.left, tmp, sum);
    }
    if(!flag && node.right != null){
        flag = flag || DFS(node.right, tmp, sum);
    }
    return flag;
}
}
```

## OUTPUT:

### 112. Path Sum

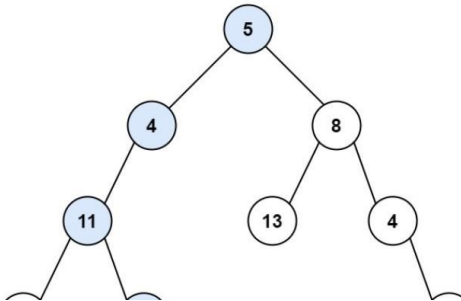
Easy 8.1K 927

Companies

Given the `root` of a binary tree and an integer `targetSum`, return `true` if the tree has a **root-to-leaf** path such that adding up all the values along the path equals `targetSum`.

A **leaf** is a node with no children.

Example 1:



```

7  *   TreeNode() {}
8  *   TreeNode(int val) { this.val = val; }
9  *   TreeNode(int val, TreeNode left, TreeNode right) {
10 *       this.val = val;
11 *       this.left = left;
12 *       this.right = right;
13 *   }
14 * }
15 **/
16
17 class Solution {
18 public boolean hasPathSum(TreeNode root, int sum) {
19     return root == null ? false : DFS(root, 0, sum);
20 }
21 public boolean DFS(TreeNode node, int tmp, int sum){
22     boolean flag = false;
  
```

Testcase Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

root =  
[5,4,8,11,null,13,4,7,2,null,null,null,1]

targetSum =

Console

Run

Submit

- **Same Tree -**

```

class Solution {
    public boolean isSameTree(TreeNode p, TreeNode q) {
        if(p==null || q==null)
            return (p==q);
        return (p.val==q.val) && isSameTree(p.left,q.left) && isSameTree(p.right,q.right);
    }
}
  
```

## OUTPUT:

### 100. Same Tree

Easy



9K



182

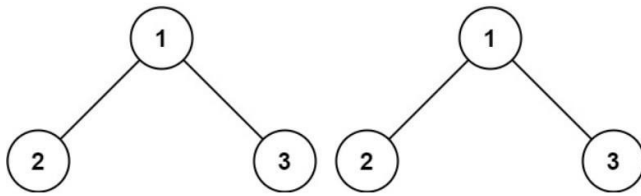


Companies

Given the roots of two binary trees  $p$  and  $q$ , write a function to check if they are the same or not.

Two binary trees are considered the same if they are structurally identical, and the nodes have the same value.

#### Example 1:



Input:  $p = [1,2,3]$ ,  $q = [1,2,3]$

Output: true



```
1 /**
2  * Definition for a binary tree node.
3  * public class TreeNode {
4  *     int val;
5  *     TreeNode left;
6  *     TreeNode right;
7  *     TreeNode() {}
8  *     TreeNode(int val) { this.val = val; }
9  *     TreeNode(int val, TreeNode left, TreeNode right) {
10 *         this.val = val;
11 *         this.left = left;
12 *         this.right = right;
13 *     }
14 * }
15 */
```

Testcase Result

Accepted Runtime: 0 ms

Case 1

Case 2

Case 3

Input

$p =$   
 $[1,2,3]$