UNIVERSITY INSTITUTE OF ENGINEERING

Department of Computer Science & Engineering

Subject Name: Competitive Coding - II

Subject Code: 20CSP351

Submitted to:

Er. Daulat Ram

Submitted by:

Name: Nikhil Kumar

UID: 20BCS1817

Section: 20BCS716_DM

Group: B

INDEX

Ex. No	List of Experiments	Conduct (MM: 12)	Viva (MM: 10)	Record (MM: 8)	Total (MM: 30)	Remarks/ Signature
1.1	To implement the concept of Arrays, Queues and Stack and Linked List					
1.2	To demonstrate the concept of String Matching algorithms					
1.3	To demonstrate the concept of Heap model					
1.4	To demonstrate the concept of Hashing					
2.1	To demonstrate the concept of Trees					
2.2	To demonstrate the concept of Graph					
2.3	To demonstrate the concept of Divide and Conquer					
3.1	To demonstrate the concept of Greedy approach					
3.2	To demonstrate the concept of Backtracking					

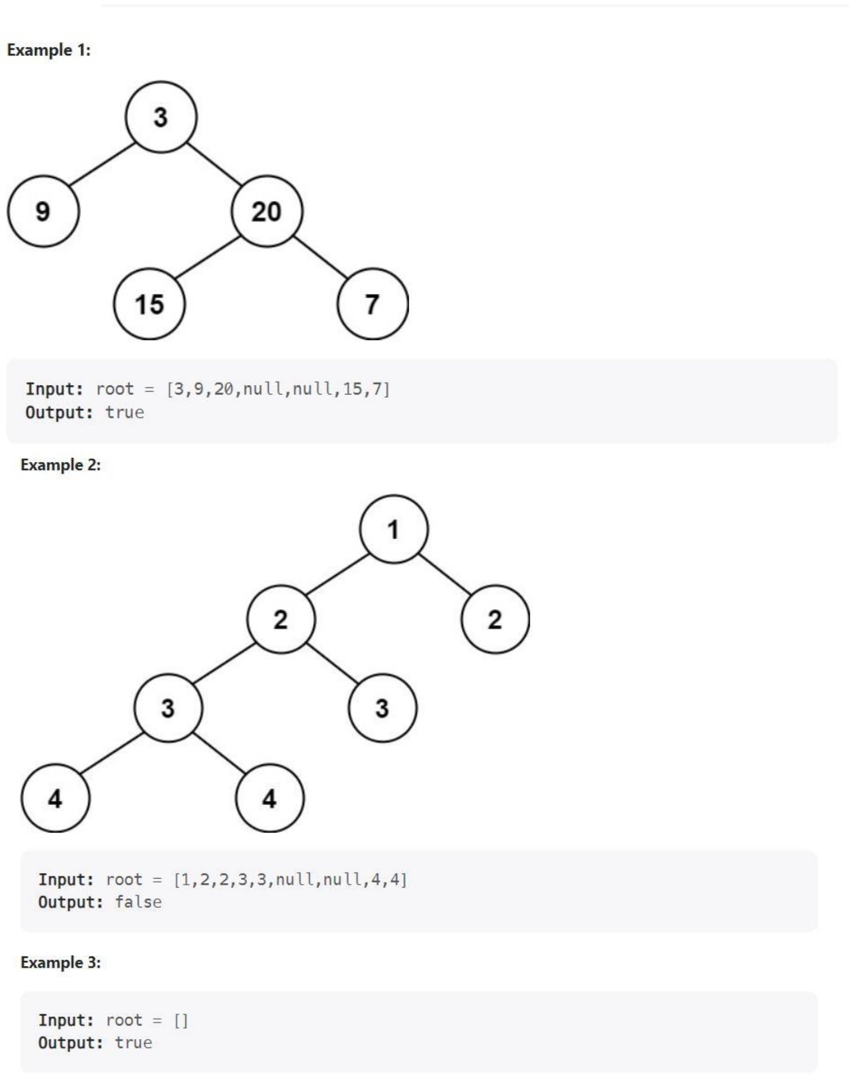
3.3	To demonstrate the concept of Dynamic Programming					

EXPERIMENT 5

Student Name: Nikhil Kumar **UID: 20BCS1817**
Branch: BE-CSE **Section/Group: 716/B**
Semester: 6th **Date of Performance: 02/04/23**
Subject Name: Competitive Coding-II
Subject Code: 20CSP-351

PROBLEM 1: BALANCED BINARY TREE PROBLEM

STATEMENT:



Example 1:

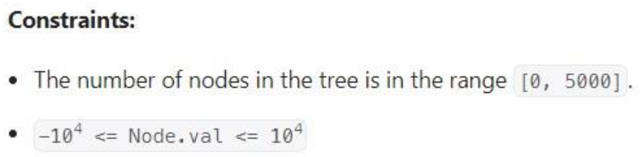
```
Input: root = [3,9,20,null,null,15,7]
Output: true
```

Example 2:

```
Input: root = [1,2,2,3,3,null,null,4,4]
Output: false
```

Example 3:

```
Input: root = []
Output: true
```



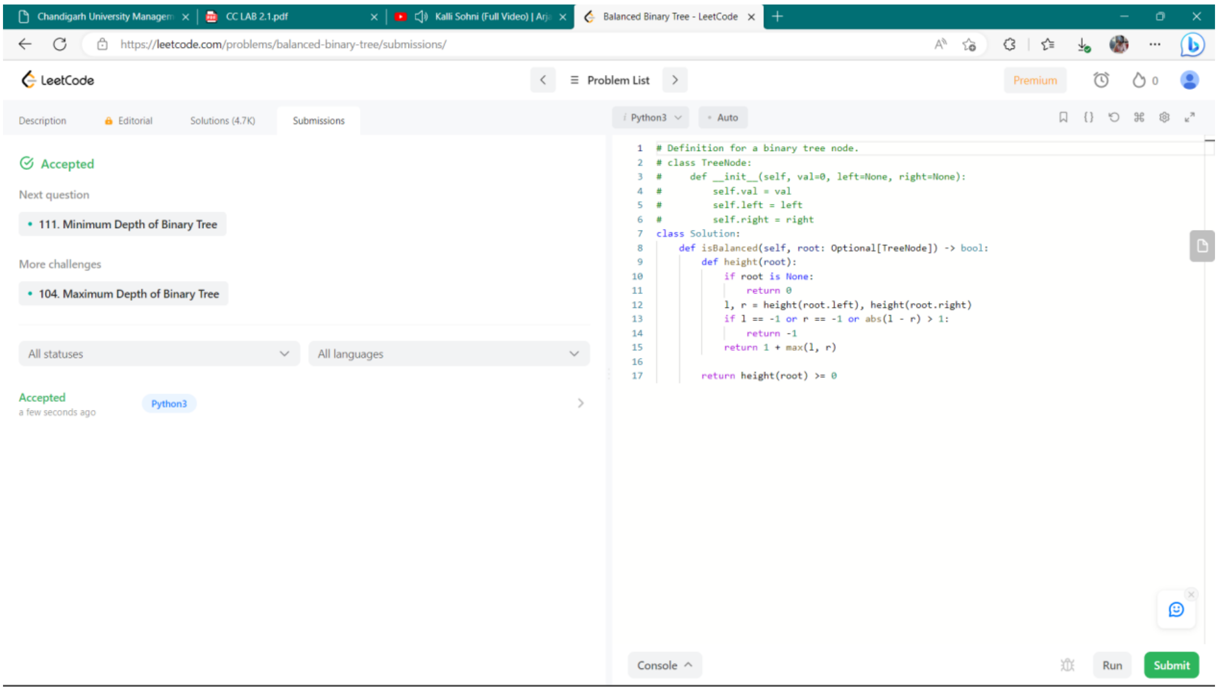
Constraints:

- The number of nodes in the tree is in the range `[0, 5000]`.
- -10^4 `<= Node.val <=` 10^4

CODE:

```
class Solution { public:
   bool ans = true;     int
solve(TreeNode* root)
   {
      if(root        ==        NULL)
return 0;        int left = solve(root-
>left);        int right = solve(root-
>right);         if(abs(left-right)>1)
ans  =  false;                return
max(left,right)+1;
   }
   bool isBalanced(TreeNode* root) {
solve(root);       return ans;
   }
};
```

OUTPUT:


Chandigarh University Managem
CC LAB 2.1.pdf
Kalli Sohni (Full Video) | Arj
Balanced Binary Tree - LeetCode
https://leetcode.com/problems/balanced-binary-tree/submissions/
LeetCode
Problem List
Premium
Description
Editorial
Solutions (4.7K)
Submissions
Python3
Auto
Console
Run
Submit

Accepted

Next question

- 111. Minimum Depth of Binary Tree

More challenges

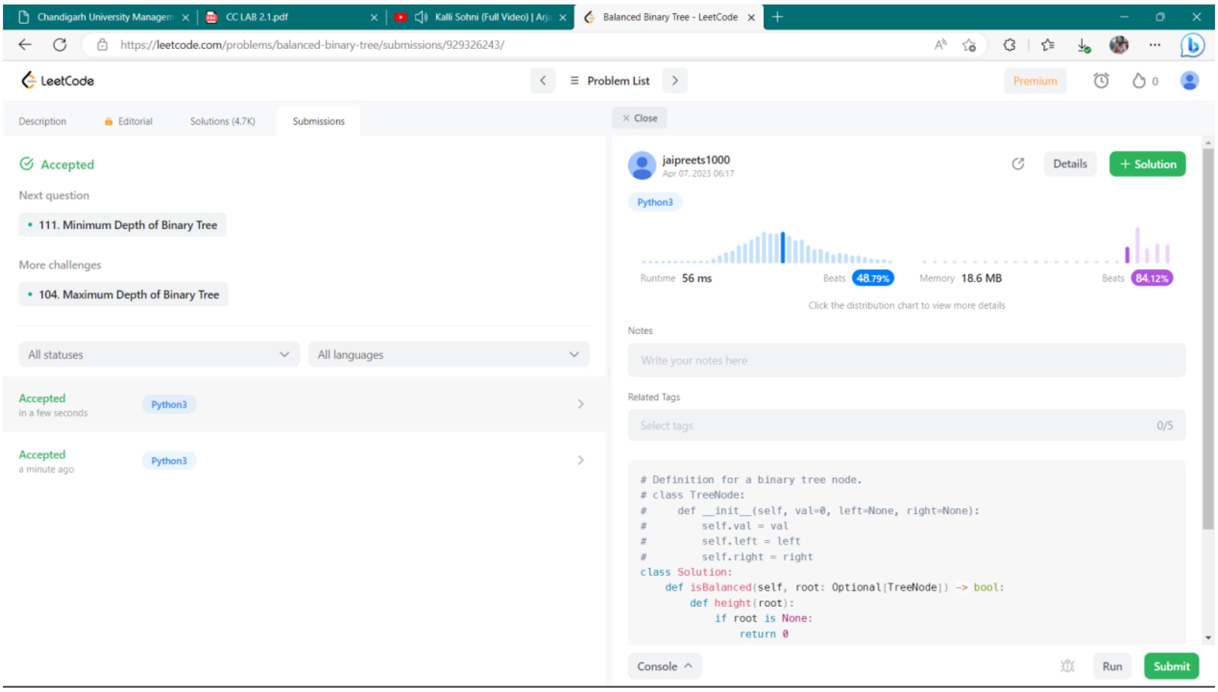
- 104. Maximum Depth of Binary Tree

All statuses | All languages

Accepted — a few seconds ago — Python3

```python
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution:
    def isBalanced(self, root: Optional[TreeNode]) -> bool:
        def height(root):
            if root is None:
                return 0
            l, r = height(root.left), height(root.right)
            if l == -1 or r == -1 or abs(l - r) > 1:
                return -1
            return 1 + max(l, r)

        return height(root) >= 0
```


Chandigarh University Managem
CC LAB 2.1.pdf
Kalli Sohni (Full Video) | Arj
Balanced Binary Tree - LeetCode
https://leetcode.com/problems/balanced-binary-tree/submissions/929326243/
LeetCode
Problem List
Premium
Description
Editorial
Solutions (4.7K)
Submissions
Close
Details
+ Solution
Console
Run
Submit

Accepted

Next question

- 111. Minimum Depth of Binary Tree

More challenges

- 104. Maximum Depth of Binary Tree

All statuses | All languages

Accepted — in a few seconds — Python3

Accepted — a minute ago — Python3

jaipreets1000
Apr 07, 2023 06:17

Python3

Runtime 56 ms Beats 48.79%

Memory 18.6 MB Beats 84.12%

Click the distribution chart to view more details

Notes

Write your notes here

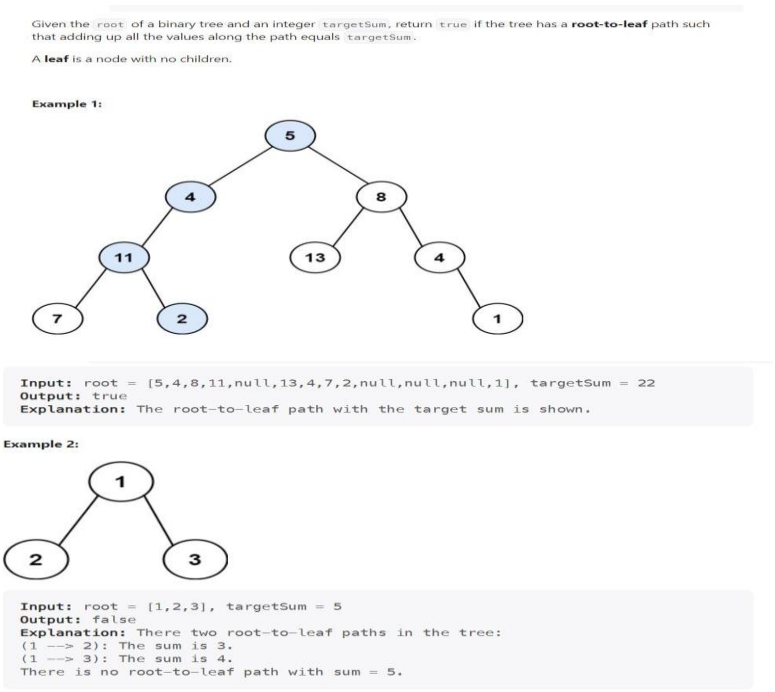
Related Tags

Select tags 0/5

```python
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution:
    def isBalanced(self, root: Optional[TreeNode]) -> bool:
        def height(root):
            if root is None:
                return 0
```

PROBLEM 2: PATH SUM PROBLEM

STATEMENT:



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`.

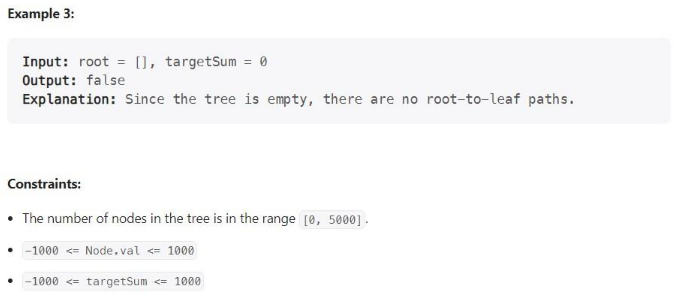
A **leaf** is a node with no children.

Example 1:

```
Input: root = [5,4,8,11,null,13,4,7,2,null,null,null,1], targetSum = 22
Output: true
Explanation: The root-to-leaf path with the target sum is shown.
```

Example 2:

```
Input: root = [1,2,3], targetSum = 5
Output: false
Explanation: There two root-to-leaf paths in the tree:
(1 --> 2): The sum is 3.
(1 --> 3): The sum is 4.
There is no root-to-leaf path with sum = 5.
```



Example 3:

```
Input: root = [], targetSum = 0
Output: false
Explanation: Since the tree is empty, there are no root-to-leaf paths.
```

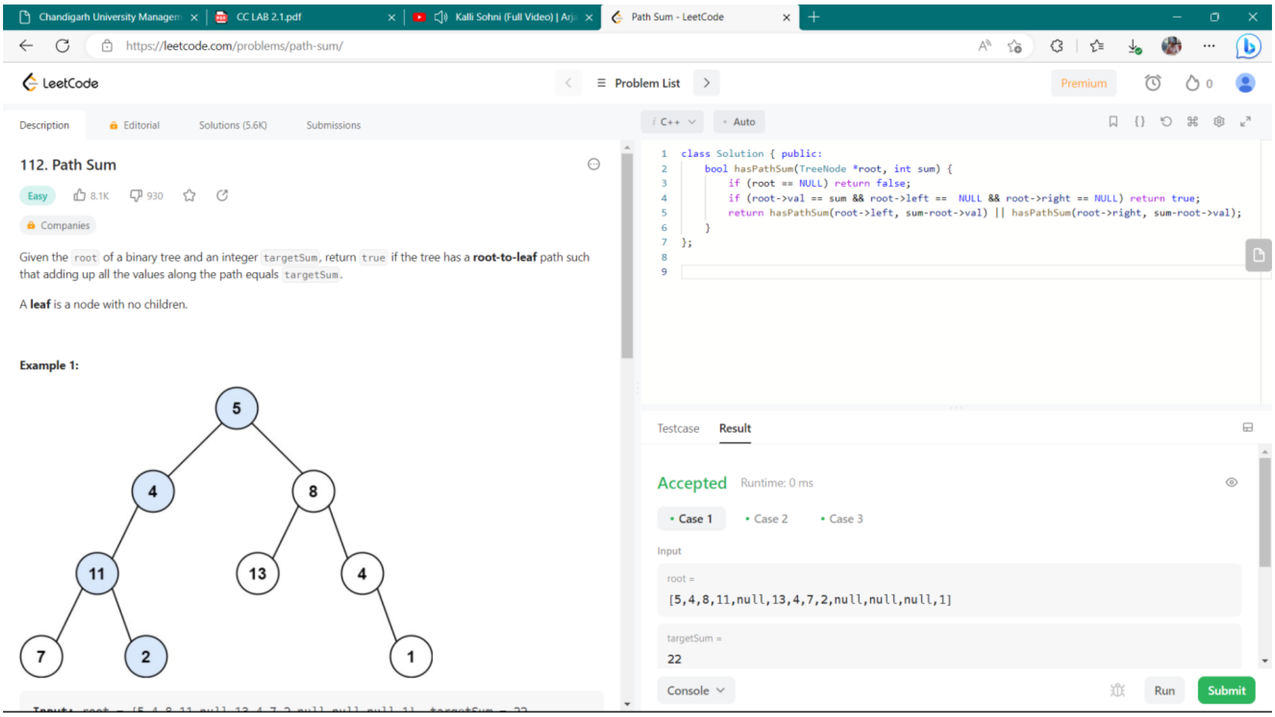
Constraints:

- The number of nodes in the tree is in the range `[0, 5000]`.
- `-1000 <= Node.val <= 1000`
- `-1000 <= targetSum <= 1000`

CODE:

```
class Solution { public:
    bool hasPathSum(TreeNode *root, int sum) {
        if (root == NULL) return false;
        if (root->val == sum && root->left ==  NULL && root->right == NULL)
return true;
        return hasPathSum(root->left, sum-root->val) || hasPathSum(root->right,
sum-root->val);
    }
};
```

OUTPUT:



https://leetcode.com/problems/path-sum/

112. Path Sum

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:

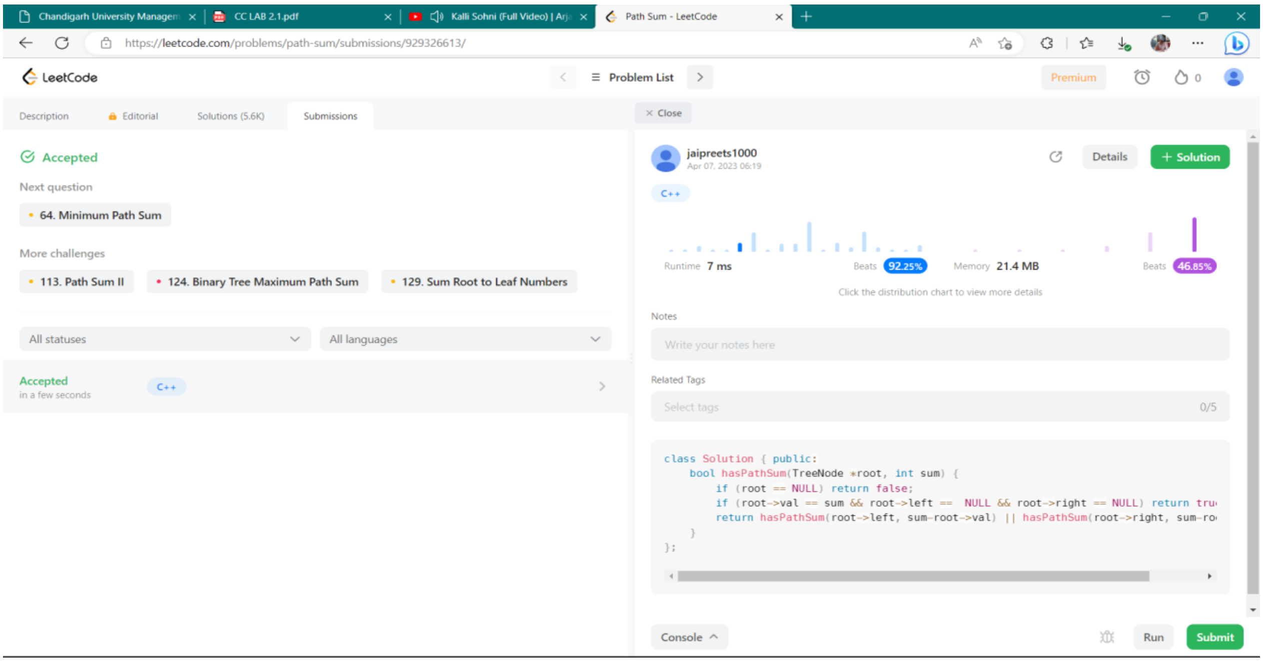
```
class Solution { public:
    bool hasPathSum(TreeNode *root, int sum) {
        if (root == NULL) return false;
        if (root->val == sum && root->left ==  NULL && root->right == NULL) return true;
        return hasPathSum(root->left, sum-root->val) || hasPathSum(root->right, sum-root->val);
    }
};
```

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 =
22
```



https://leetcode.com/problems/path-sum/submissions/929326613/

Accepted

Next question

64. Minimum Path Sum

More challenges

113. Path Sum II
124. Binary Tree Maximum Path Sum
129. Sum Root to Leaf Numbers

All statuses
All languages

Accepted in a few seconds C++

jaipreets1000
Apr 07, 2023 06:19

Runtime 7 ms Beats 92.25% Memory 21.4 MB Beats 46.85%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```
class Solution { public:
    bool hasPathSum(TreeNode *root, int sum) {
        if (root == NULL) return false;
        if (root->val == sum && root->left ==  NULL && root->right == NULL) return tru
        return hasPathSum(root->left, sum-root->val) || hasPathSum(root->right, sum-ro
    }
};
```