Experiment 2.1

Student Name: Yash Gupta **UID: 20BCS5009**
Branch: BE-CSE **Section/Group:20BCS_DM-716 B**
Semester: 6 **Date of Performance: 03/04/23**
Subject Name: CC LAB **Subject Code: 20CSP_351**

1. Aim:

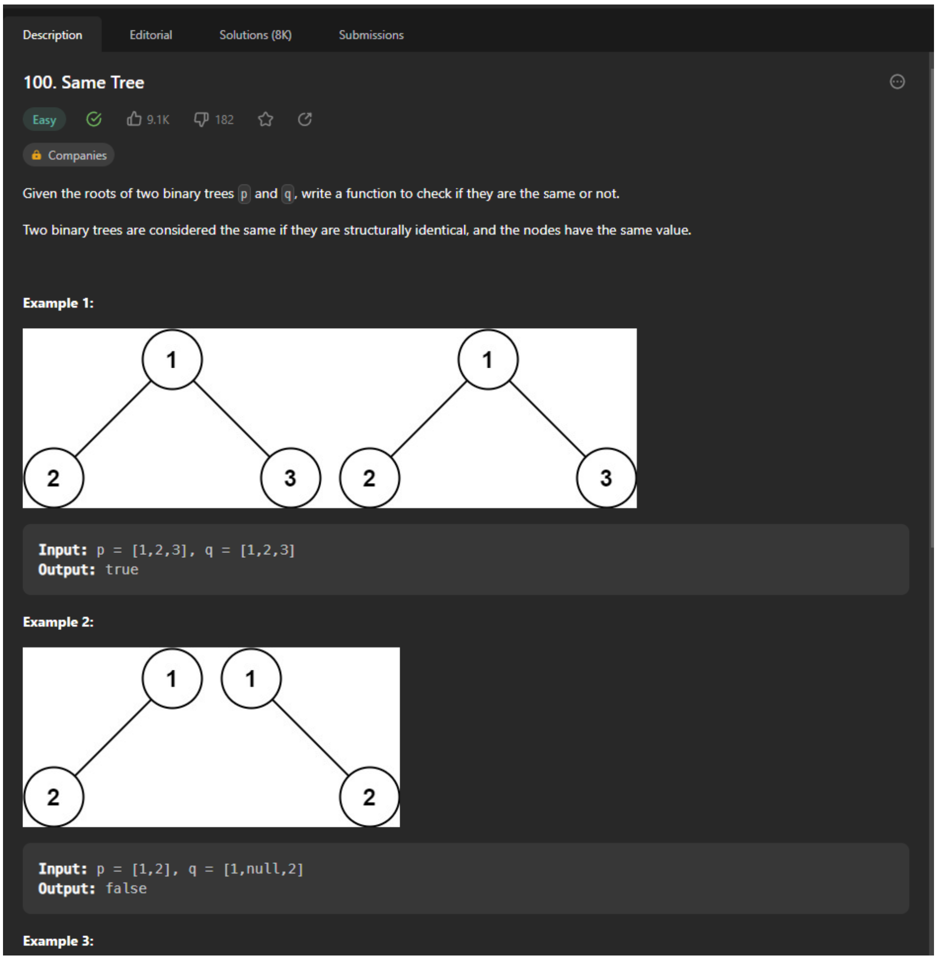
To implement the concept of trees.

2. Objective:

- The objective is to build problem solving capability and to learn the basic concepts of data structures.
- Understand the problem and find out better approach to solve particular problem

3. LeetCode code and output:

- Sample Tree



100. Same Tree

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.

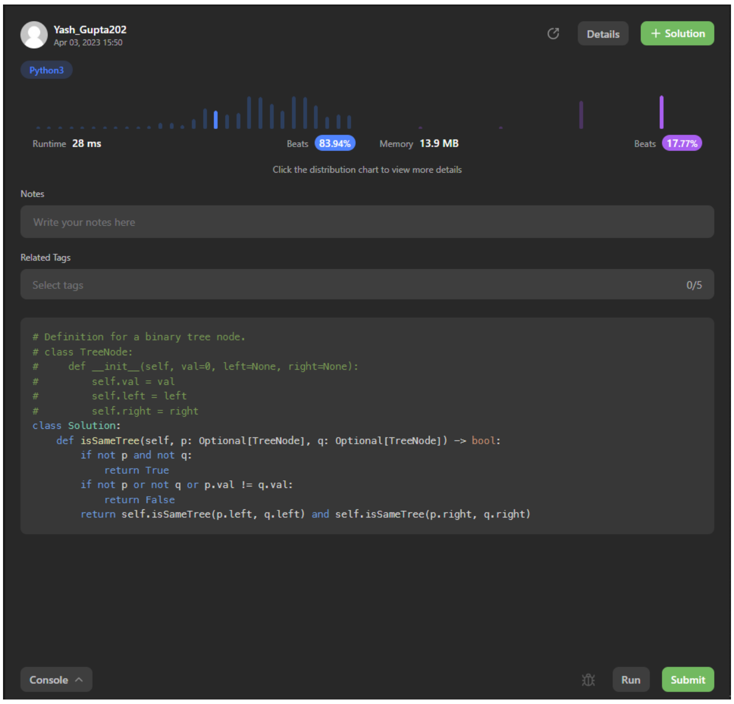
Example 1:

```
Input: p = [1,2,3], q = [1,2,3]
Output: true
```

Example 2:

```
Input: p = [1,2], q = [1,null,2]
Output: false
```

Example 3:



Yash_Gupta202
Apr 03, 2023 15:50

Runtime 28 ms Beats 83.94% Memory 13.9 MB Beats 17.77%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```
# 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 isSameTree(self, p: Optional[TreeNode], q: Optional[TreeNode]) -> bool:
        if not p and not q:
            return True
        if not p or not q or p.val != q.val:
            return False
        return self.isSameTree(p.left, q.left) and self.isSameTree(p.right, q.right)
```

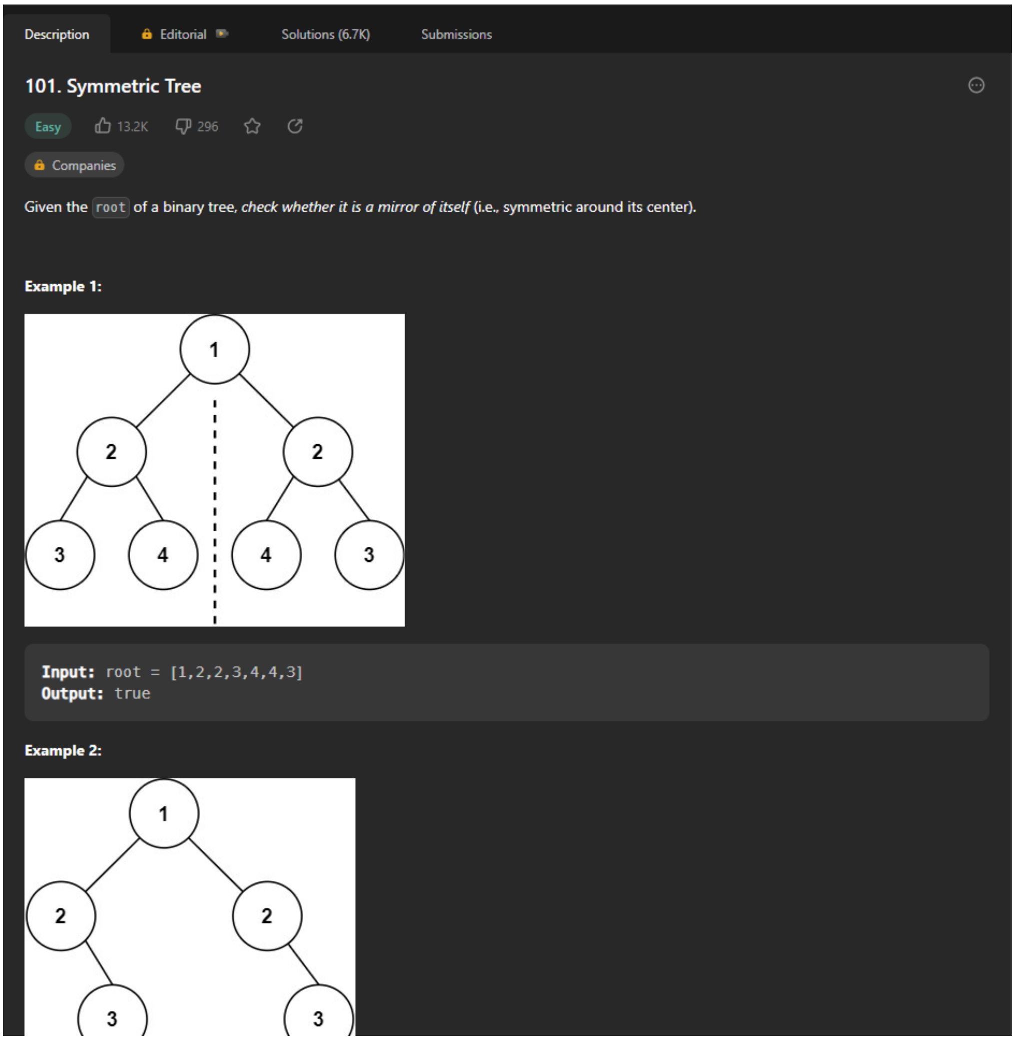
```
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution:
    def isSameTree(self, p: Optional[TreeNode], q: Optional[TreeNode]) -> bool:
        if not p and not q:
            return True
        if not p or not q or p.val != q.val:
            return False
        return self.isSameTree(p.left, q.left) and self.isSameTree(p.right, q.right)
```

- **Symmetric Tree**

```
class Solution(object):

    def compare(self,rootleft,rootright):
        if rootleft == None or rootright == None:
            return rootleft==rootright
        return rootleft.val == rootright.val and self.compare(rootleft.left,rootright.right) and
self.compare(rootleft.right,rootright.left)

    def isSymmetric(self, root):
        if not root:
            return True
        return self.compare(root.left,root.right)
```



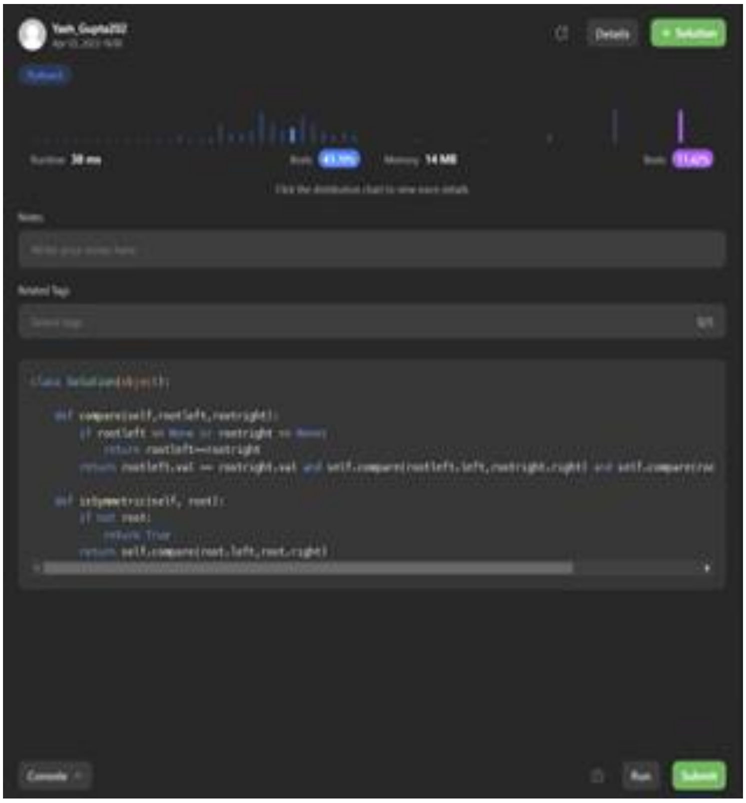
101. Symmetric Tree

Given the `root` of a binary tree, *check whether it is a mirror of itself* (i.e., symmetric around its center).

Example 1:

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

Example 2:



```
class Solution(object):

    def compare(self,rootleft,rootright):
        if (rootleft == None or rootright == None):
            return rootleft==rootright
        return rootleft.val == rootright.val and self.compare(rootleft.left,rootright.right) and self.compare(roo

    def isSymmetric(self, root):
        if not root:
            return True
        return self.compare(root.left,root.right)
```