

Task-2

J.Sreekar Sarma
VU21CSEN0300040

1. Flatten Binary Tree to Linked List

```
class Solution:
    def flatten(self, root: TreeNode) -> None:
        curr = root

        while curr:
            if curr.left != None:
                p = curr.left
                while p.right != None:
                    p = p.right

                p.right = curr.right

            curr.right = curr.left
            curr.left = None

            curr = curr.right
```

OUTPUT

The screenshot shows a LeetCode submission interface for the problem 'Flatten Binary Tree to Linked List'. The submission is by 'Sreekar Sarma Josyabhatla' and is marked as 'Accepted'. The runtime is 0 ms, which is 100.00% faster than other submissions, and the memory usage is 17.93 MB, which is 9.01% less than other submissions. The code is written in Python3 and implements the following logic:

```
class Solution:
    def flatten(self, root: TreeNode) -> None:
        curr = root

        while curr:
            if curr.left != None:
                p = curr.left
                while p.right != None:
                    p = p.right

                p.right = curr.right

            curr.right = curr.left
            curr.left = None

            curr = curr.right
```

The test case shows the input root = [1, 2, 5, 3, 4, null, 6] and a diagram of the resulting linked list structure:

```
graph TD
    1((1)) --> 2((2))
    1 --> 5((5))
    2 --> 3((3))
    2 --> 4((4))
    5 --> 6((6))
```

2. Trapping Rain Water

```
class Solution(object):
    def trap(self, height):
        n = len(height)
        ans = 0
        st = []
```

Task-2

J.Sreekar Sarma
VU21CSEN0300040

```
for r in range(n):
    while st and height[st[-1]] < height[r]:
        m = st.pop()

    if not st:
        break

    l = st[-1]
    h = min(height[r] - height[m], height[l] -
height[m])

    w = r - l - 1

    ans += h * w
    st.append(r)
return ans
```

OUTPUT

The screenshot displays the LeetCode interface for the 'Trapping Rain Water' problem. The top section shows the problem title and a submission status of 'Accepted' by Sreekar Sarma Josyabhatla. Below this, a performance summary indicates a runtime of 12 ms (beating 62.52%) and a memory usage of 19.55 MB (beating 5.67%). A bar chart visualizes the runtime performance across various test cases. The code editor on the right shows the submitted Python solution, which uses a stack to calculate the trapped water volume. The test result section at the bottom confirms the solution is accepted for the provided input: [0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1], resulting in an output of 6.

```
class Solution(object):
    def trap(self, height):
        n = len(height)
        ans = 0
        st = []
```