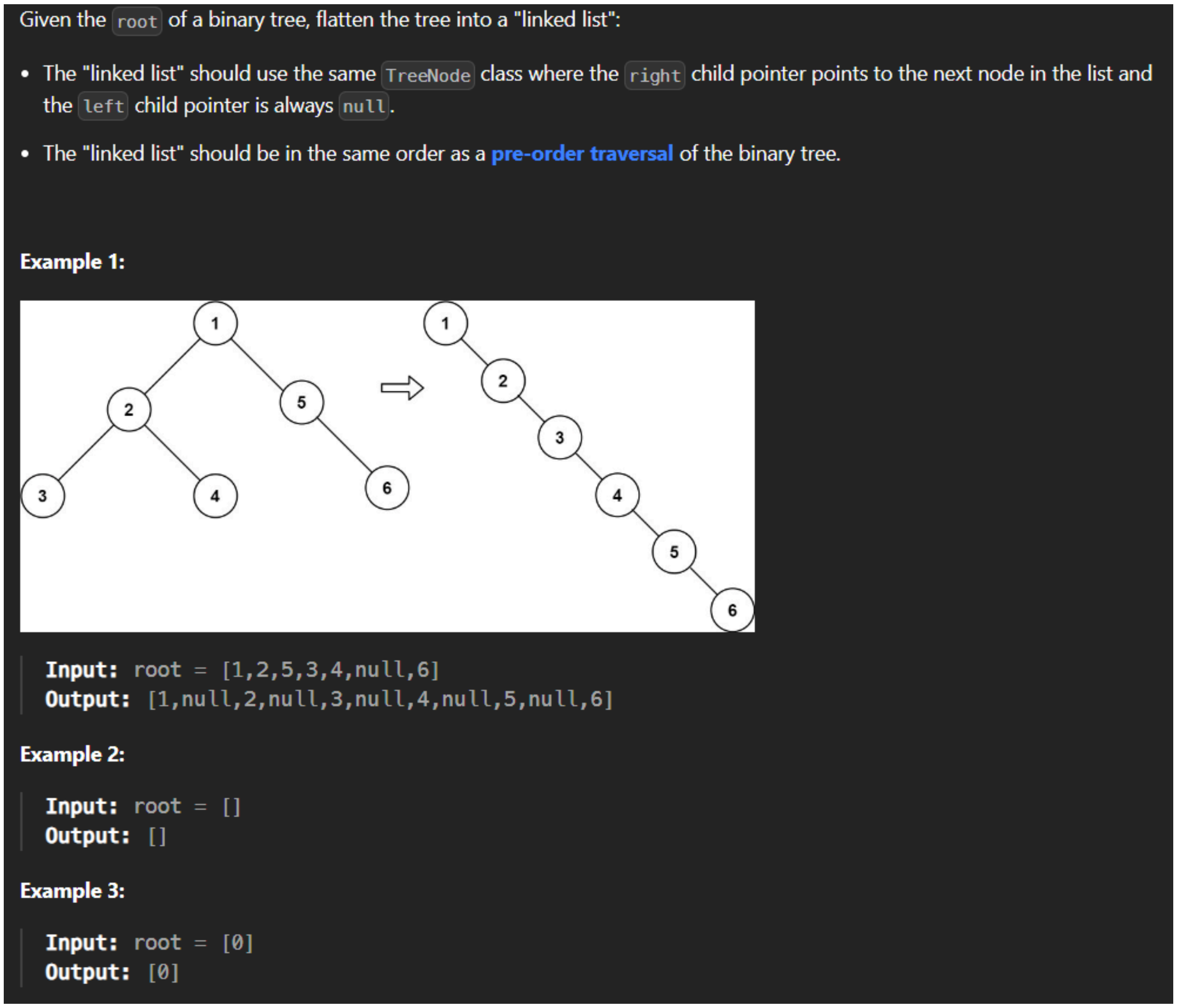
114. Flatten Binary Tree to Linked List

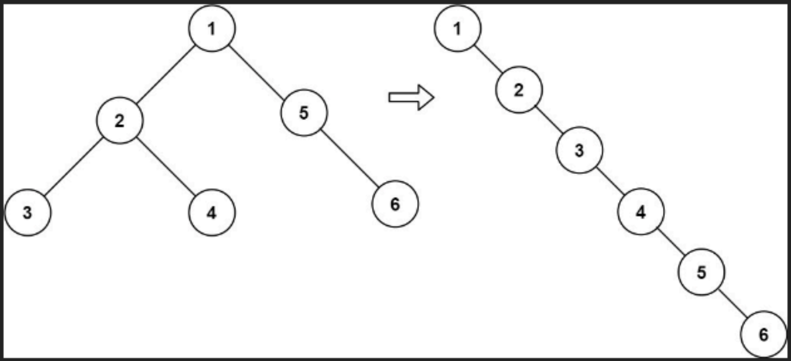
Problem Statement

1. 

Given the `root` of a binary tree, flatten the tree into a "linked list":

- The "linked list" should use the same `TreeNode` class where the `right` child pointer points to the next node in the list and the `left` child pointer is always `null`.
- The "linked list" should be in the same order as a **pre-order traversal** of the binary tree.

Example 1:



```
Input: root = [1,2,5,3,4,null,6]
Output: [1,null,2,null,3,null,4,null,5,null,6]
```

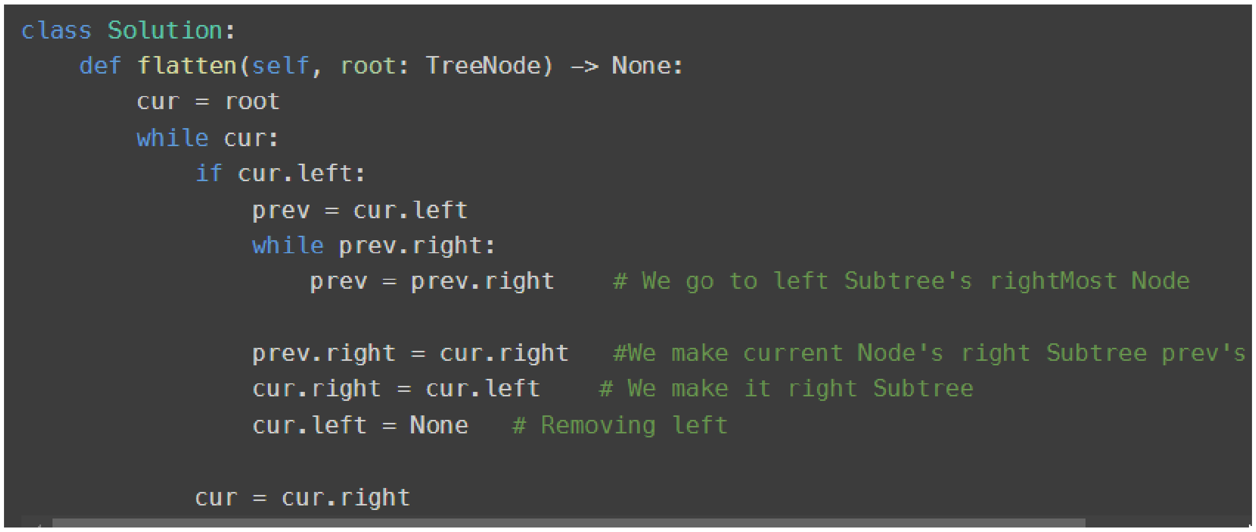
Example 2:

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

Example 3:

```
Input: root = [0]
Output: [0]
```

Python code:

1. 

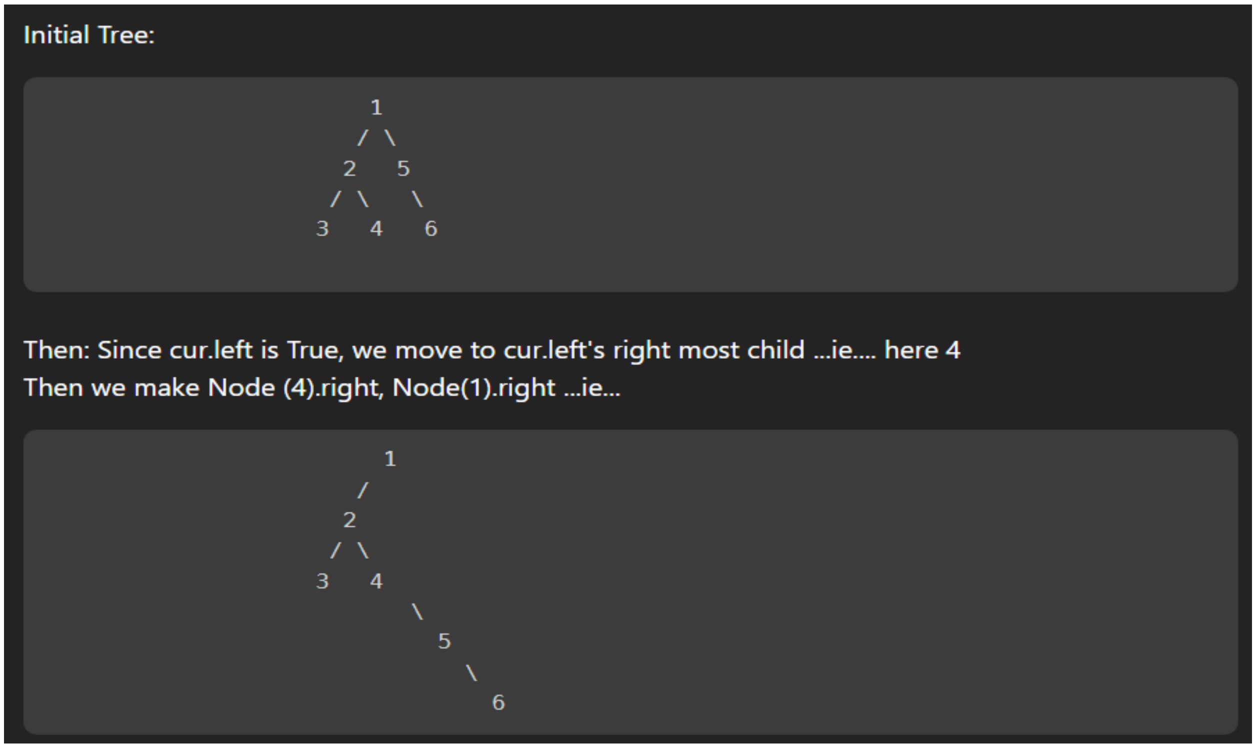
```python
class Solution:
    def flatten(self, root: TreeNode) -> None:
        cur = root
        while cur:
            if cur.left:
                prev = cur.left
                while prev.right:
                    prev = prev.right    # We go to left Subtree's rightMost Node

                prev.right = cur.right   #We make current Node's right Subtree prev's
                cur.right = cur.left    # We make it right Subtree
                cur.left = None   # Removing left

            cur = cur.right
```

Approach:

1. First we will check whether there is left child for our cur…if yes…we check if the left node has right child node…if yes..we assign prev = prev.right
2. Prev.right must be linked to cur.right
3. Just check this links to get intuition

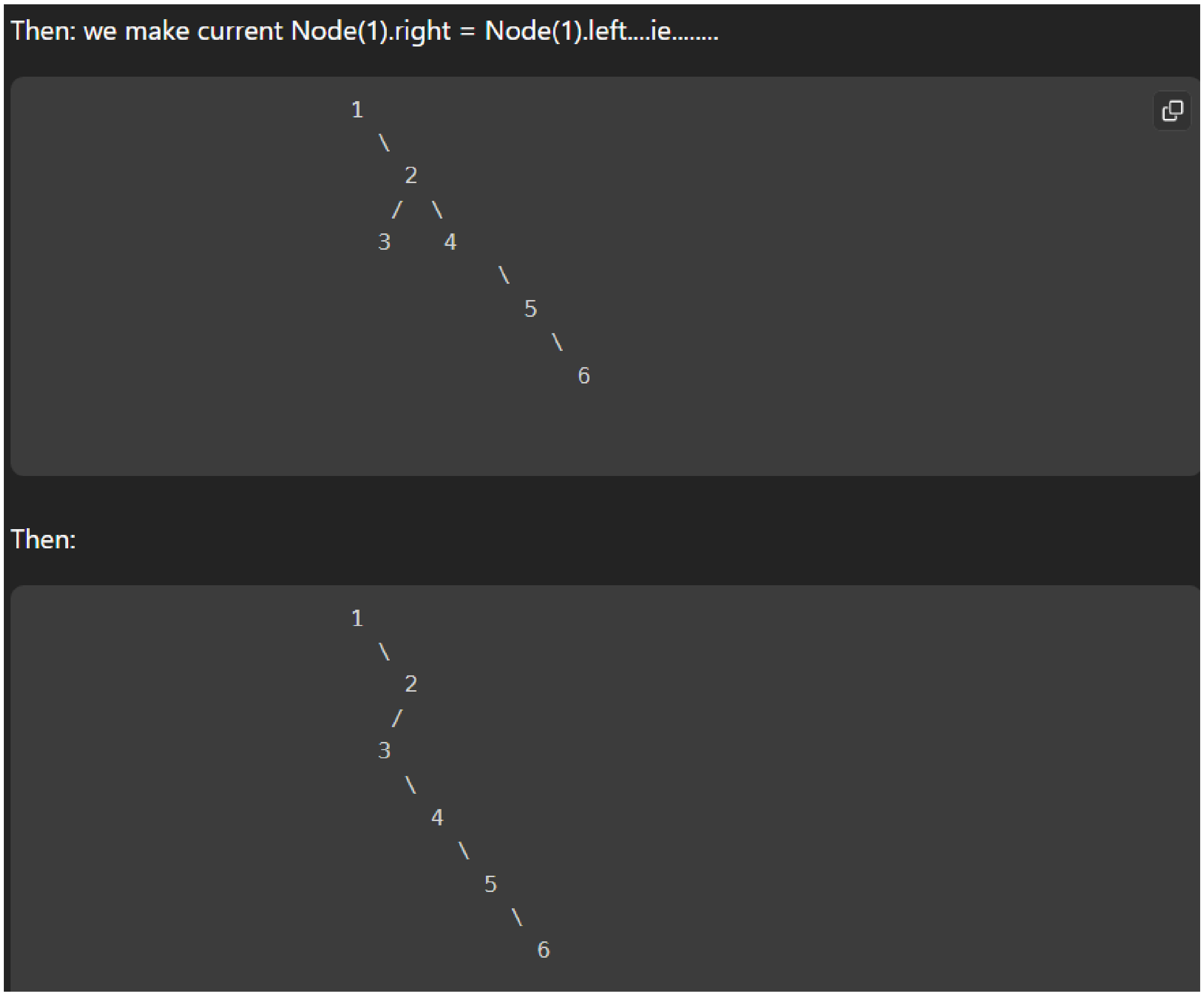


Initial Tree:

```
        1
       / \
      2   5
     / \   \
    3   4   6
```

Then: Since cur.left is True, we move to cur.left's right most child ...ie.... here 4
Then we make Node (4).right, Node(1).right ...ie...

```
         1
        /
      2
     / \
    3   4
         \
          5
           \
            6
```

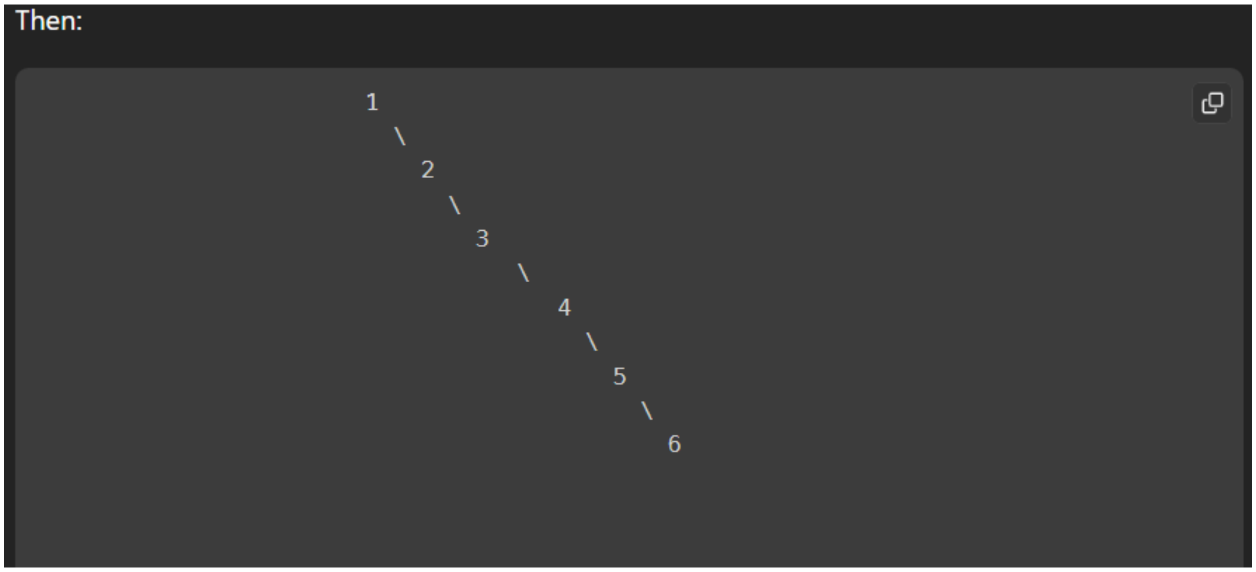

Then: we make current Node(1).right = Node(1).left....ie........

```
1
  \
    2
   / \
  3    4
           \
             5
               \
                 6
```

Then:

```
1
  \
    2
   /
  3
    \
      4
        \
          5
            \
              6
```

4.


Then:

```
1
  \
    2
      \
        3
           \
              4
                \
                  5
                    \
                      6
```