< Back

Python Inorder Transverse

liangyonghit ★ 64 Last Edit: October 24, 2018 2:20 PM 3.7K VIEWS

64

class Solution(object):

```
def treeToDoublyList(self, root):
    if not root: return
    dummy = Node(0, None, None)
    prev = dummy
    stack, node = [], root
    while stack or node:
        while node:
            stack.append(node)
            node = node.left
        node = stack.pop()
        node.left, prev.right, prev = prev, node, node
        node = node.right
    dummy.right.left, prev.right = prev, dummy.right
    return dummy.right
```

Comments: 10 Best Most Votes Newest to Oldest Oldest to Newest

Type comment here... (Markdown is supported)

Post

JN1992 ★ 7 October 22, 2018 12:01 AM

I found this solution the most straightforward. It's just a minor modification of the BST inorder traversal template. Good job.

7 Reply

NotADwarf ★ 3 December 1, 2019 3:49 PM

This is not an in-place solution as you create an additional data structure (the stack).

2 Show 1 reply Reply

theseungjin ★ 97 January 19, 2020 4:58 AM

Thank you for this solution. It's doesn't meet the in-place constraint of the question but this really helped me further my understanding of iterative inorder traversal using stack.

1 Reply

helloworldhelloworld ★ 1 June 25, 2019 4:32 AM

Great solution! But I am confused on "dummy.right.left, prev.right = prev, dummy.right
return dummy.right" Could you please explain that? Thanks!!!

0 Show 1 reply Reply

healthycola ★ 1 January 8, 2019 9:47 PM

I like the stack approach. Clean!

0 Reply

cjhlucky ★ 2 January 8, 2019 12:50 PM

great work

0 Reply

MingXu-123 ★ 31 May 8, 2020 11:31 PM

```
class Solution:
    def inOrder(self, root):
        if not root:
            return
        self.inOrder(root.left)
        if not self.head:
            self.head = root
        if self.prev:
            self.prev.right = root
            root.left = self.prev
```

Read More