Thursday, March 10, 2022 9:19 AM

Populate Inorder Successor for all nodes
Medium Accuracy: 47.39% Submissions: 22285 Points: 4

Given a Binary Tree, write a function to populate next pointer for all nodes. The next pointer for every node should be set to point to inorder successor.

Example 1:

Input:
10
/ \
8 12
/
3
Output: 3->8 8->10 10->12 12->-1**Explanation:** The inorder of the above tree is :3 8 10 12. So the next pointer of node 3 is pointing to 8 , next pointer of 8 is pointing to 10 and so on.And next pointer of 12 ispointing to -1 as there is no inorder successor
of 12.
Example 2:

Input:1
/ \
2 3**Output:** 2->1 1->3 3->-1
Your Task:

You do not need to read input or print anything. Your task is to complete the function **populateNext()** that takes the root node of the binary tree as input parameter.

Expected Time Complexity: O(N)

Expected Auxiliary Space: O(N)

Constraints:

1<=n<=10^5

1<=data of the node<=10^5

From <https://practice.geeksforgeeks.org/problems/populate-inorder-successor-for-all-nodes/1#>

116. Populating Next Right Pointers in Each Node

Medium

6017220Add to ListShare

You are given a **perfect binary tree** where all leaves are on the same level, and every parent has two children. The binary tree has the following definition:

```
struct Node {
  int val;
  Node *left;
  Node *right;
  Node *next;
}
```

Populate each next pointer to point to its next right node. If there is no next right node, the next pointer should be set to `NULL`.

Initially, all next pointers are set to `NULL`.

Example 1:

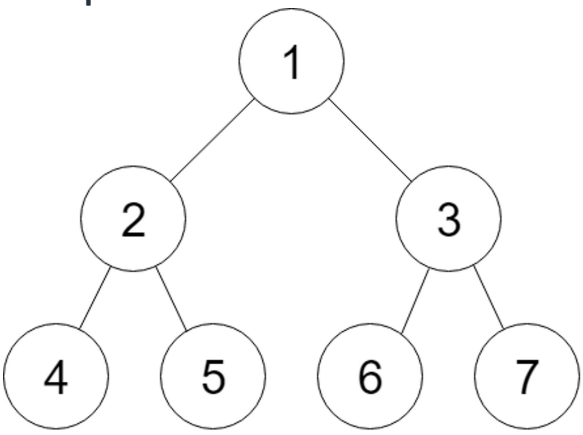

Figure A

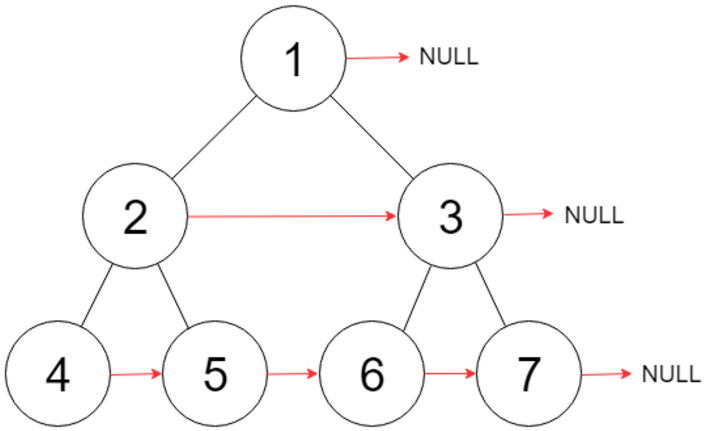

Figure B

Input: root = [1,2,3,4,5,6,7]

Output: [1,#,2,3,#,4,5,6,7,#]

Explanation: Given the above perfect binary tree (Figure A), your function should populate each next pointer to point to its next right node, just like in Figure B. The serialized output is in level order as connected by the next pointers, with '#' signifying the end of each level.

Example 2:

Input: root = []

Output: []

Constraints:

- The number of nodes in the tree is in the range `[0, 212 - 1]`.
- `-1000 <= Node.val <= 1000`

Follow-up:

- You may only use constant extra space.
- The recursive approach is fine. You may assume implicit stack space does not count as extra space for this problem.

From <https://leetcode.com/problems/populating-next-right-pointers-in-each-node/>