

117. Populating Next Right Pointers in Each Node II

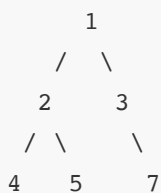
Follow up for problem "*Populating Next Right Pointers in Each Node*".

What if the given tree could be any binary tree? Would your previous solution still work?

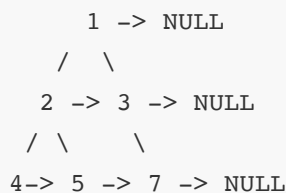
Note:

- You may only use constant extra space.

For example, Given the following binary tree,



After calling your function, the tree should look like:



每次注意处理它的前一个节点和下一级的头节点。用层序遍历，提供了层序遍历一种新的思路。

```
/**
 * Definition for binary tree with next pointer.
 * struct TreeLinkNode {
 *   int val;
 *   TreeLinkNode *left, *right, *next;
 *   TreeLinkNode(int x) : val(x), left(NULL), right(NULL), next(NULL) {}
 * };
 */
class Solution {
public:
    void connect(TreeLinkNode *root) {
        if( root == NULL ) return;

        TreeLinkNode* head = NULL; /// 下一级的头节点
        TreeLinkNode* pre = NULL;  /// 下一级的previous节点
```

```

TreeLinkNode* cur = root;  /// 当前层的当前节点

while( cur !=NULL ){
    while( cur!= NULL ){
        /// 左孩子
        if( cur->left != NULL ){
            if( pre != NULL ) pre->next = cur->left;
            else head = cur -> left;

            pre = cur->left;
        }

        /// 右孩子
        if( cur -> right != NULL ){
            if( pre != NULL ) pre->next = cur -> right;
            else head = cur -> right;          /// 前面已经断节了

            pre = cur -> right;
        }
    }

    cur = cur -> next;      /// 移到下一个兄弟节点
    head = NULL;
    pre = NULL;
}
};

```