```
/*
给定一个二叉树，原地将它展开为链表。

例如，给定二叉树

    1
   / \
  2   5
 / \   \
3   4   6

将其展开为：

1
 \
  2
   \
    3
     \
      4
       \
        5
         \
          6

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/flatten-binary-tree-to-linked-list
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 分别递归的展开左右子树,在左子树不为空的情况下,整体将左子树插入到右子树.
- 遍历顺序可以是左右根也可以是右左根.

---

方法一:C++_递归展开

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */


/*
分别递归展开左右子树,然后将左子树全部展开到右子树.
*/

/**
```

```cpp
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution
{
    private:
        void __dfsPostOrder(TreeNode* node)
        {
            /*当前节点为空*/
            if(node == NULL)
            {
                return;
            }
            /* 展开左子树*/
            __dfsPostOrder(node->left);
            /* 展开右子树*/
            __dfsPostOrder(node->right);
            /* 将左子树插入到右子树*/
            if(node->left)
            {
                TreeNode* node_right_temp = node->right ; /*保存原来的右子树根节点*/
                TreeNode* node_left_tail  = node->left  ;

                /*寻找尾巴*/
                while(node_left_tail&&node_left_tail->right)
                {
                    node_left_tail = node_left_tail->right;
                }
                /*将左子树整体插入右子树*/
                node->right             = node->left;
                node->left              = NULL;
                node_left_tail->right   = node_right_temp;
                node_left_tail->left    = NULL;
            }
        }
    public:
        void flatten(TreeNode* root)
        {
            __dfsPostOrder(root);
            return ;
        }
};
/*
执行结果:
通过
显示详情
执行用时 :4 ms, 在所有 C++ 提交中击败了98.35% 的用户
内存消耗 :9.6 MB, 在所有 C++ 提交中击败了91.48%的用户
*/
```

AlimyBreak
<2019.10.10>