



喜刷刷

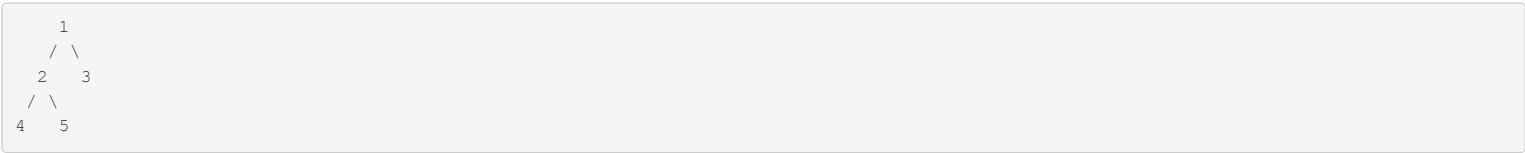
Monday, November 17, 2014

[LeetCode新题] Binary Tree Upside Down

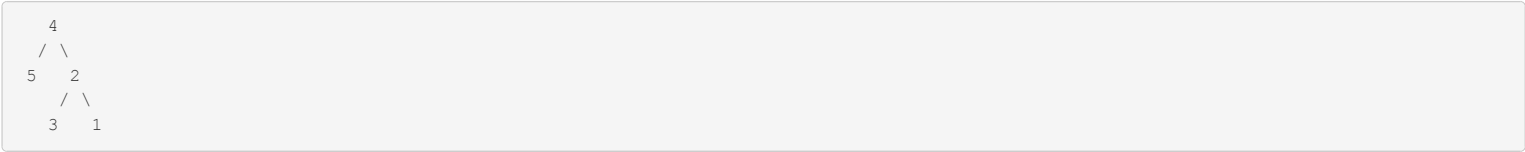
Given a binary tree where all the right nodes are either leaf nodes with a sibling (a left node that shares the same parent node) or empty, flip it upside down and turn it into a tree where the original right nodes turned into left leaf nodes. Return the new root.

For example:

Given a binary tree `{1,2,3,4,5}`,



return the root of the binary tree `[4,5,2,#,#,3,1]`.



思路：

LeetCode最近出了收费模式的新题，需要买他们出的电子书才能做。不过既然已经免费使用了它家这么多资源，就当买本书做点贡献。

这题第一眼看上去觉得没头绪，不知道怎么上下翻转和左右翻转。但在纸上画几个例子就清楚了。以题目的例子来解释：

1. 对于一个parent来说，加入有right node，必须得有left node。而有left node，right node可以为空。而right node必须为叶子节点。所以该树每层至多有2个节点，并且2节点有共同的parent。
2. 所以对于最底层来说，必有一个left node，而这个left node则为整个新树的根——例子中的4为最底层的左节点，最后成为新树的root。
3. 原树的根节点，变为了新树的最右节点。
3. 对于子树1 2 3来说，需要在以2为根的子树2 4 5建立成新树4 5 2后，插入到新树的最右节点2下面。原树的根节点root为left child，原树root->right为新树的left nnode

递归实现：

About Me



Yanbing Shi
Follow 0

[View my complete profile](#)

Blog Archive

- ▼ 2014 (130)
 - ▼ November (130)
 - [C++ Specific Questions - Overloading VS Overriding...](#)
 - [Microsoft Onsite Interview Checklist](#)
 - [基础data structure/algorithm列表](#)
 - [面试中的clarifying questions](#)
 - [\[LeetCode\] Wildcard Matching](#)
 - [\[LeetCode\] Regular Expression Matching](#)
 - [\[LeetCode\] Max Points on a Line](#)
 - [\[LeetCode\] Word Ladder I, II](#)
 - [\[LeetCode新题\] Intersection of Two Linked Lists](#)
 - [\[LeetCode\] First Missing Positive](#)
 - [\[LeetCode\] Simplify Path](#)
 - [\[LeetCode\] LRU Cache](#)
 - [\[LeetCode\] Merge Intervals](#)
 - [\[LeetCode\] Insert Interval](#)
 - [\[LeetCode\] Longest Valid Parentheses](#)
 - [\[LeetCode\] Largest Rectangle in Histogram](#)
 - [\[LeetCode\] Unique Binary Search Trees I, II](#)
 - [\[LeetCode\] Distinct Subsequences](#)
 - [\[LeetCode\] Longest Consecutive Sequence](#)
 - [\[LeetCode\] Permutation Sequence](#)

```

1 class Solution {
2 public:
3     TreeNode *upsideDownBinaryTree(TreeNode *root) {
4         TreeNode *temp, *newRoot = NULL;
5         temp = buildUpsideDownBT(root, newRoot);
6         return newRoot;
7     }
8
9     TreeNode *buildUpsideDownBT(TreeNode *root, TreeNode *&newRoot) {
10        if(!root) return root;
11        if(!root->left && !root->right) {
12            newRoot = root;
13            return root;
14        }
15        TreeNode *parent = buildUpsideDownBT(root->left, newRoot);
16        parent->left = root->right;
17        parent->right = root;
18        root->left = root->right = NULL;
19        return parent->right;
20    }
21 };

```

总结：

1. 这个递归的核心是，每次建立好一个新的子树后，要返回新子树的最右节点（ln 19），以便上层的节点可以接回到这个节点的下面。
2. 但如果只返回最右节点，则我们无法知道最后整个新树的根在哪里。所以再base case里必须给新根赋值(ln 12)
3. 每次需要reset最右节点的left/right node，否则最后一层递归，递归到例子中的1节点时，返回前1节点的left/right node仍然为原来的值，而并不为NULL。

Posted by Yanbing Shi at 8:57 AM



+2 Recommend this on Google

Labels: algorithm, binary tree, data structure, Leetcode, recursive

5 comments:



Hejia Pan November 22, 2014 at 9:00 PM

觉得这题就是后序变成层序.

[Reply](#)

▼ Replies



Ann GUO September 16, 2015 at 7:05 PM

赞！



Ann GUO September 16, 2015 at 7:10 PM

赞！



Charles Hu November 5, 2015 at 10:46 AM

[\[LeetCode\] Next Permutation](#)

[\[LeetCode\] Palindrome Partitioning I, II](#)

[\[LeetCode\] Text Justification](#)

[\[LeetCode\] Edit Distance](#)

[\[LeetCode\] Decode Ways](#)

[\[LeetCode\] ZigZag Conversion](#)

[\[LeetCode\] Reverse Words in a String](#)

[\[LeetCode\] Longest Palindromic Substring](#)

[\[LeetCode\] Surrounded Regions](#)

[\[LeetCode\] Set Matrix Zeroes](#)

[\[LeetCode\] Unique Paths I, II](#)

[\[LeetCode\] Triangle](#)

[\[LeetCode\] Gas Station](#)

[\[LeetCode\] Best Time to Buy and Sell Stock I, II, ...](#)

[\[LeetCode\] Jump Game I, II](#)

[\[LeetCode\] Maximum Product Subarray](#)

[\[LeetCode\] Maximum Subarray](#)

[\[LeetCode\] Word Break I, II](#)

[\[LeetCode\] Anagrams](#)

[\[LeetCode\] Spiral Matrix I, II](#)

[\[LeetCode\] Rotate Image](#)

[\[LeetCode\] Climbing Stairs](#)

[\[LeetCode\] Multiply Strings](#)

[\[LeetCode\] Length of Last Word](#)

[\[LeetCode\] Roman to Integer](#)

[\[LeetCode\] Integer to Roman](#)

[\[LeetCode\] String to Integer \(atoi\)](#)

[\[LeetCode\] Count and Say](#)

[\[LeetCode\] Longest Common Prefix](#)

[\[LeetCode\] Palindrome Number](#)

[\[LeetCode\] Reverse Integer](#)

[\[LeetCode\] Plus One](#)

[\[LeetCode\] Pascal's Triangle I, II](#)

[\[LeetCode\] Single Number I, II](#)

[\[LeetCode\] Merge k Sorted Lists](#)

[\[LeetCode\] Reverse Nodes in k-Group](#)

[\[LeetCode\] Add Binary](#)

[\[LeetCode\] Add Two Numbers](#)

[\[LeetCode\] Swap Nodes in Pairs](#)

[\[LeetCode 新题\] Read N](#)



我也写了一个. 每次只返回根. 但是在递归前记录要链接的节点. 在自己电脑上运行了一下, 没发现什么问题. 大侠们确认一下?

```
TreeNode* upDown(TreeNode*root){
if (!root) return nullptr;
if (!root->left) return root;

assert(root->left);
TreeNode* l= root->left;
TreeNode* r= root->right;
TreeNode* newH= upDown(root->left);
l->right= root;
l->left= upDown(r);
root->left=0;
root->right=0;
if (r) { r->left=0; r->right=0;}
return newH;
}
```

[Reply](#)



gary zhang July 18, 2016 at 11:45 PM

不是很理解题目。。这样子满足题目的条件吗？

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

[Reply](#)

Enter your comment...

Comment as:

Select profile...

[Publish](#)

[Preview](#)

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

[Characters Given Read4](#)

[\[LeetCode\] Reverse Linked List II](#)

[\[LeetCode\] Reorder List](#)

[\[LeetCode\] Partition List](#)

[\[LeetCode\] Rotate List](#)

[\[LeetCode\] Clone Graph](#)

[\[LeetCode\] Copy List with Random Pointer](#)

[\[LeetCode\] Insertion Sort List](#)

[\[LeetCode\] Merge Two Sorted Lists](#)

[\[LeetCode\] Remove Duplicates from Sorted List I, I...](#)

[\[LeetCode\] Remove Nth Node From End of List](#)

[\[LeetCode\] Valid Parentheses](#)

[\[LeetCode\] Evaluate Reverse Polish Notation](#)

[\[LeetCode 新题\] Min Stack](#)

[\[LeetCode\] Restore IP Addresses](#)

[\[LeetCode\] Generate Parentheses](#)

[\[LeetCode\] Word Search](#)

[\[LeetCode\] Gray Code](#)

[\[LeetCode\] Valid Sudoku, Sudoku Solver](#)

[\[LeetCode\] N-Queens I, II](#)

[\[LeetCode\] Letter Combinations of a Phone Number](#)

[\[LeetCode\] Permutations I, II](#)

[\[LeetCode\] Subsets I, II](#)

[\[LeetCode\] Combination Sum I, II](#)

[\[LeetCode\] Combinations](#)

[\[LeetCode\] Substring with Concatenation of All Wor...](#)

[\[LeetCode\] Implement strStr\(\) - KMP 解法](#)

[\[LeetCode\] Merge Sorted Array](#)

[\[LeetCode 新题\] Binary Tree Upside Down](#)

[\[LeetCode\] Trapping Rain Water](#)

[\[LeetCode\] Linked List Cycle I, II](#)

[\[LeetCode\] Minimum Window Substring](#)

[\[LeetCode\] Longest Substring Without Repeating Cha...](#)

[\[LeetCode\] Valid Palindrome](#)

[\[LeetCode\] Remove Duplicates from Sorted Array I, ...](#)

[\[LeetCode\] Remove Element](#)
[\[LeetCode\] Sort Colors](#)
[\[LeetCode\] Container With Most Water](#)
[\[LeetCode\] 4Sum](#)
[\[LeetCode\] 3Sum Closest](#)
[\[LeetCode\] 3Sum](#)

Simple template. Powered by [Blogger](#).