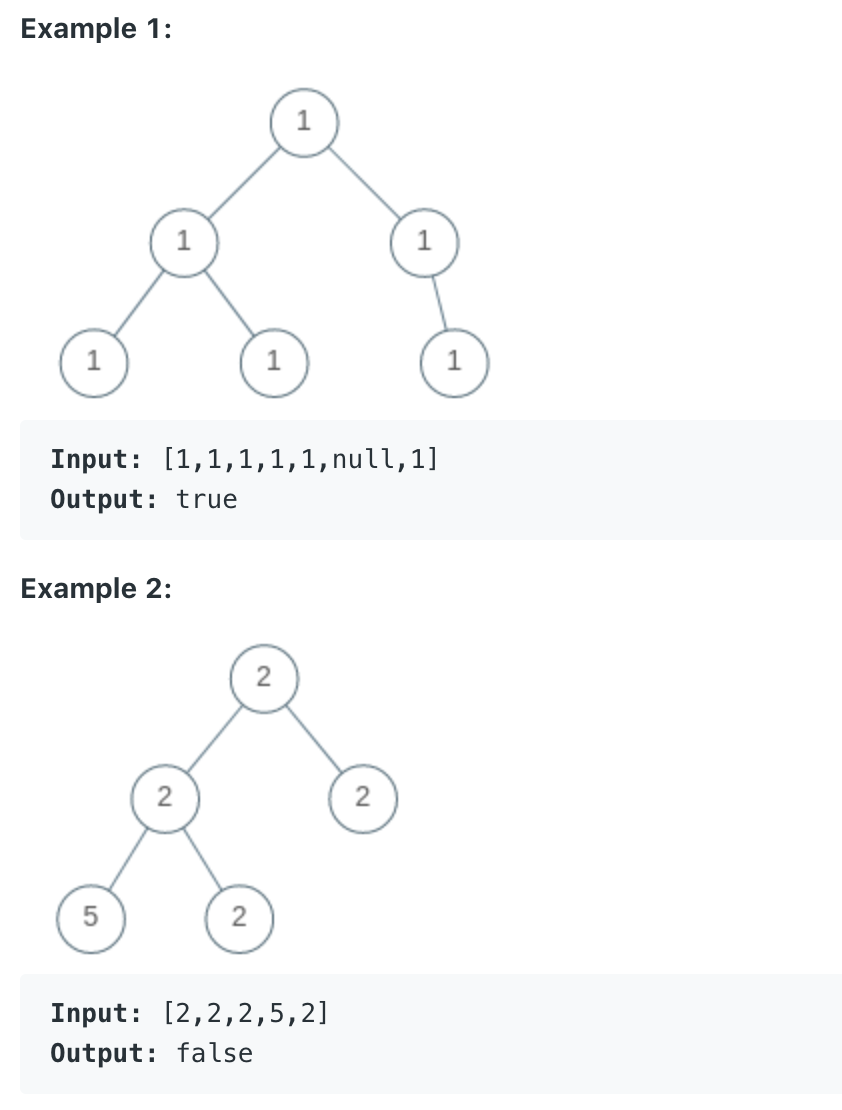
**965. Univalued Binary Tree**

**题目：**A binary tree is *univalued* if every node in the tree has the same value.

Return true if and only if the given tree is univalued. 问二叉树的每个节点的值是不是都是一样的。

****

这题和上次的100很像，但100是比较两个二叉树的，这题是比较自身的值是否一致。所以大致思路会延续上次100，但会有细微不同

**思路：递归**

递归有两大部分：终点（递归出口，逻辑边界）和自身调用部分（递归表达式，规律等）

写好一个递归算法要把握三个方面：1.提取重复逻辑 2.合适的退出条件3.控制逻辑边界。

（copy上次的，拿过来再强调一下）

**终点：**

1. 合适的退出条件：

发现左支或右支与当前节点的值不同，return 0

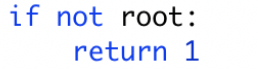
**注意： 这里自己写很容易写错成（root.val != root.right）,思考题：为什么这样写是不对的？正确答案应该怎么写？**

**答案：**正确写法**：**（root.val != root.right.val）

root.right是右支，而不是右支节点值。root.val 肯定是一直都不会等于root.right的，即treeNode != val。

1. 控制逻辑边界

当两个二叉树都为空时，即递归到了叶子结点时，return 1。



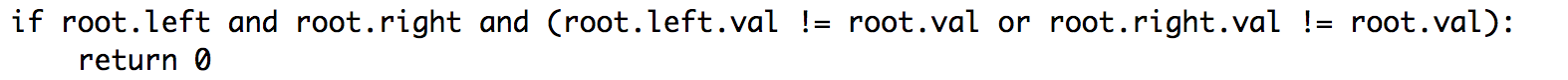
自身调用部分：

1. 提取重复逻辑：

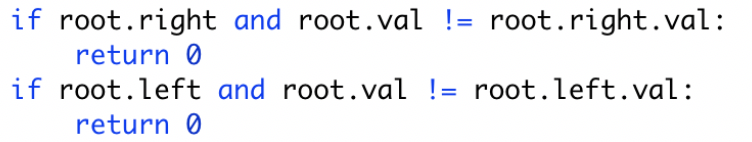
这题的重复逻辑就是不断的判断左支或右支与当前节点的值是否相同，因为该逻辑可以直接写在if的判断语句中，所以当if判断完成后直接就是return，中间没有额外的计算操作。

**所以这一题的重点就是在与如何写if判断语句，也就是下面的when**

**When何时用这部分**:我觉得要是理不清怎么写，一定要先把左支和右支分开写，至少条理会清楚点。If判断语句也不要写太长。我当时第一次左右放一起写就导致忘记某些特殊case。

**错误示范：**

**正确示范：**



何时对自身进行调用：即不是叶子结点，也没有遇到不同值时，也就是else

**How这部分怎么计算:** 基本没什么计算部分，省略

What返回什么:

return 1: 当递归到了叶子结点时

return 0:值不一样时

return result: 用and逻辑连接左右两支的递归返回结果。

这个网址是用BFS和DFS解决二叉树问题的，也可以看看：

<https://blog.csdn.net/fuxuemingzhu/article/details/85385974#_46>

<https://blog.csdn.net/chlele0105/article/details/38759593>

这个网址是讲class内部函数调用的，可以看看

<https://blog.csdn.net/CLHugh/article/details/75000104>

