<https://www.nowcoder.com/practice/cf7e25aa97c04cc1a68c8f040e71fb84?tpId=13&&tqId=11214&rp=1&ru=/ta/coding-interviews&qru=/ta/coding-interviews/question-ranking>

**牛客网的不好之处----测试用例无法自己编辑,这个题的题目描述垃圾的一逼!!!**

## my--核心思想:计算树高度(层数),每层要写满

分析:算法难度不难啊

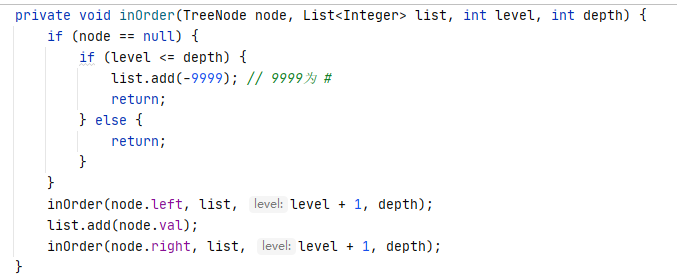
my思路:

叶子节点的空左右子节点不计入 # 利用树的高度写

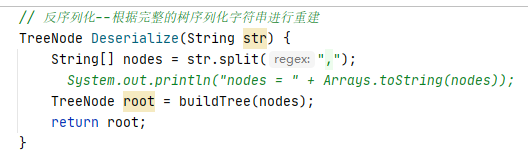
**序列化**注意拼串--很无聊繁琐的



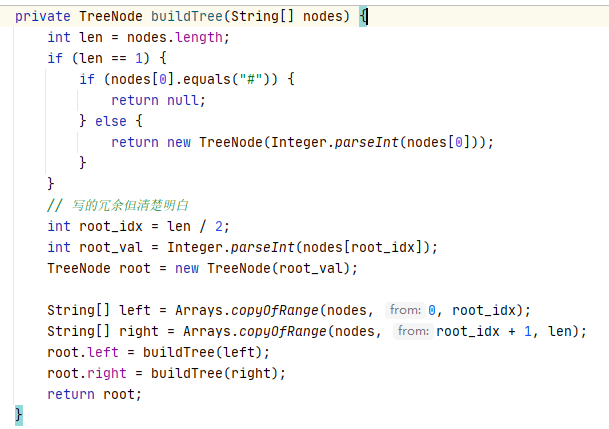
inOrder dfs遍历注意高度和当前深度的判断



**反序列化:**



递归,根据中序去找根,以及左右子树的区间

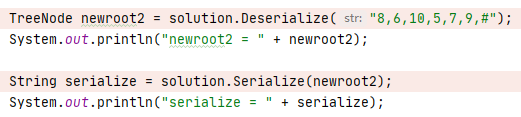


**核心逻辑: 我将depth层的二叉树写满(空为#) 因此一共2^(n-1)个,中间位置的元素是根**

**这种序列化是逻辑自洽的,题目案例跑不通**

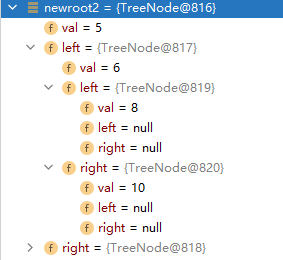
**执行结果:**

测试用例:



结果:

重建的树



根据此树序列化结果:



**如下的例子就不符合上述规定的要求:**



输入串的长度(节点个数)就不符合!!!!

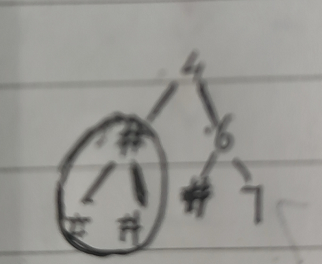
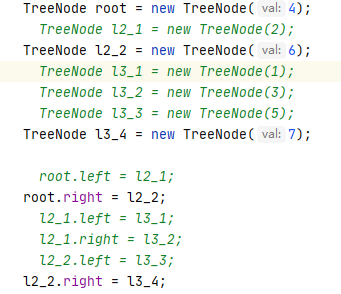


这样的才符合,高度depth必须写满!!!,空的用#

这棵树以5为根,右子树全部为空,但递归的时候会出问题,因此还需要完善代码,不妨看看参考答案!

**补充--总结错在何处**

考虑如下的一棵树--图示如下:



**序列化**结果为:



这确实是正确的中序遍历结果,但入上图所画,4的左子树没有补满,所以**无法反序列化**

## my--写满版代码修改

序列化部分OK了



反序列化有问题,递归到4左子树, 三个null如何判断? 取index=1 建立新节点就会空指针异常!

再添加一个小判断即可:



小tips : dfs中序遍历的时候,带着**传参的list泛型必须写TreeNode** (便于处理null) 别写Integer

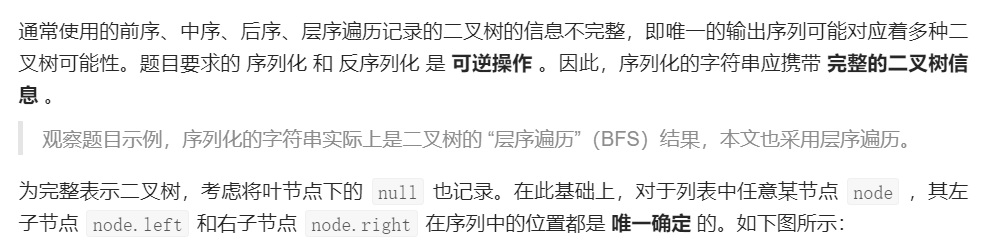
最终通过了,总结来看其实逻辑并不难!!!!!加油吧老铁

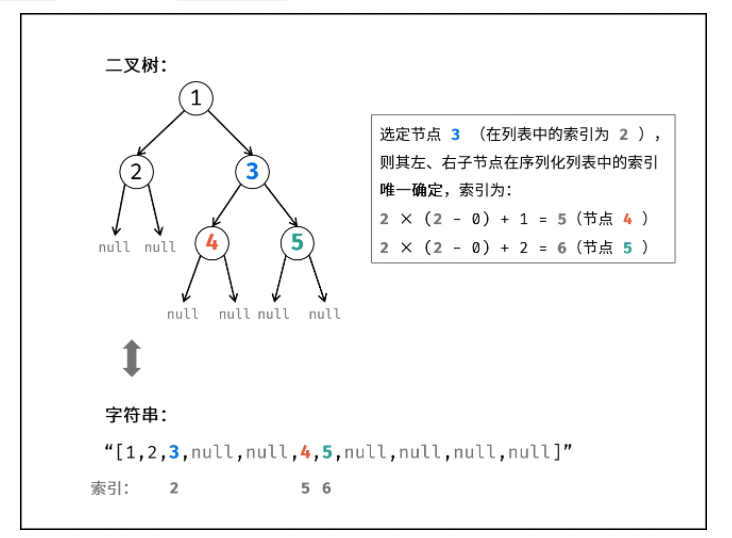
## 参考--bfs队列

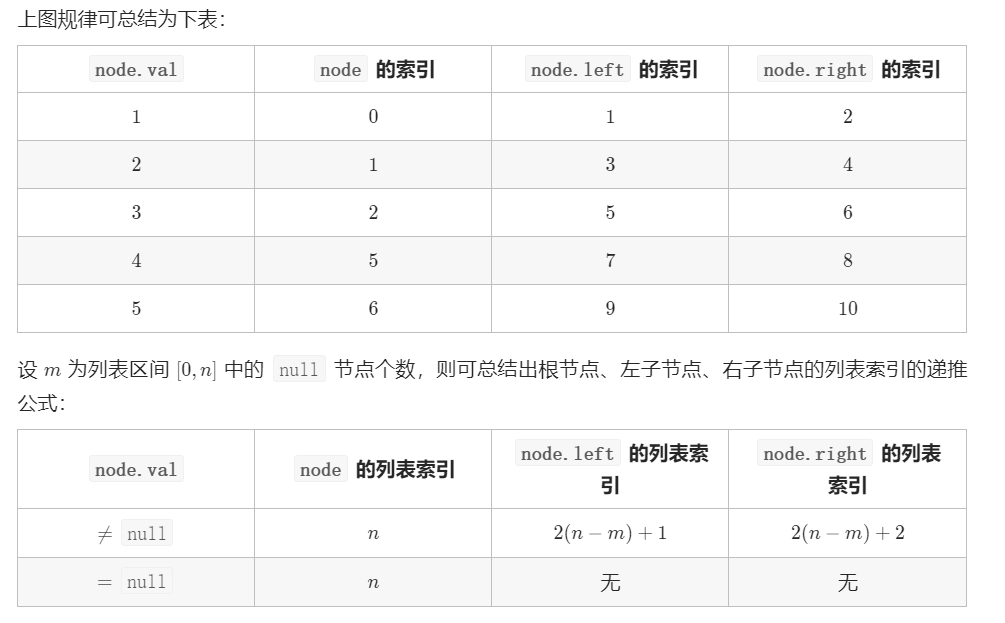
<https://leetcode-cn.com/problems/xu-lie-hua-er-cha-shu-lcof/submissions/>

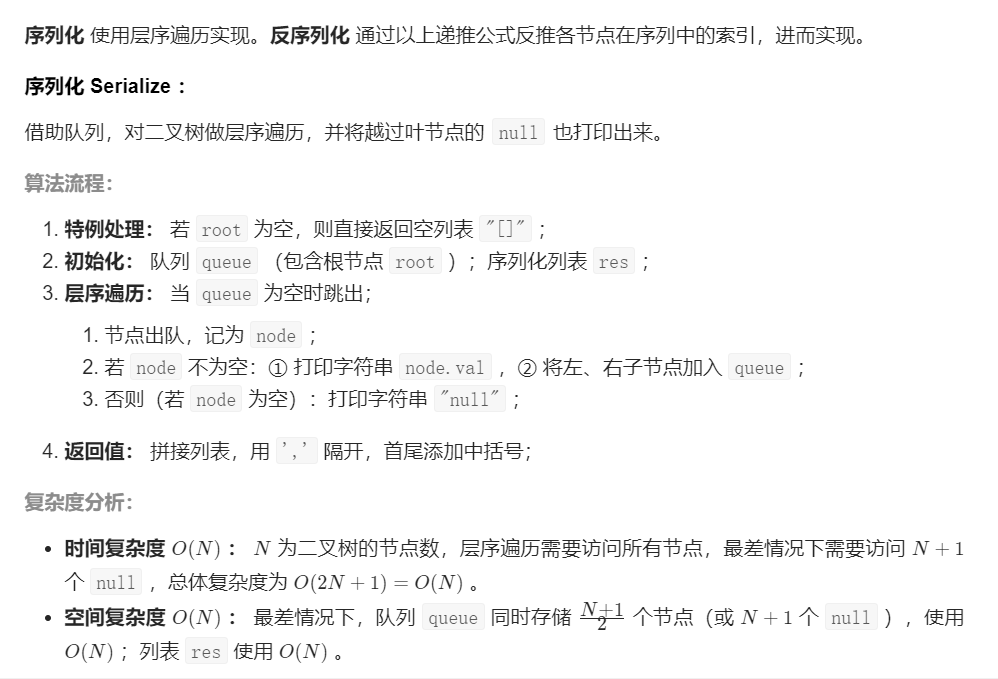
其实逻辑真的并不难!!!!!!!

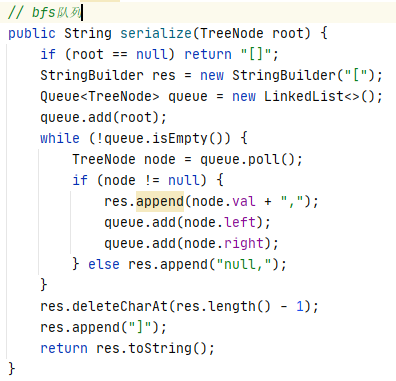
### 1--序列化



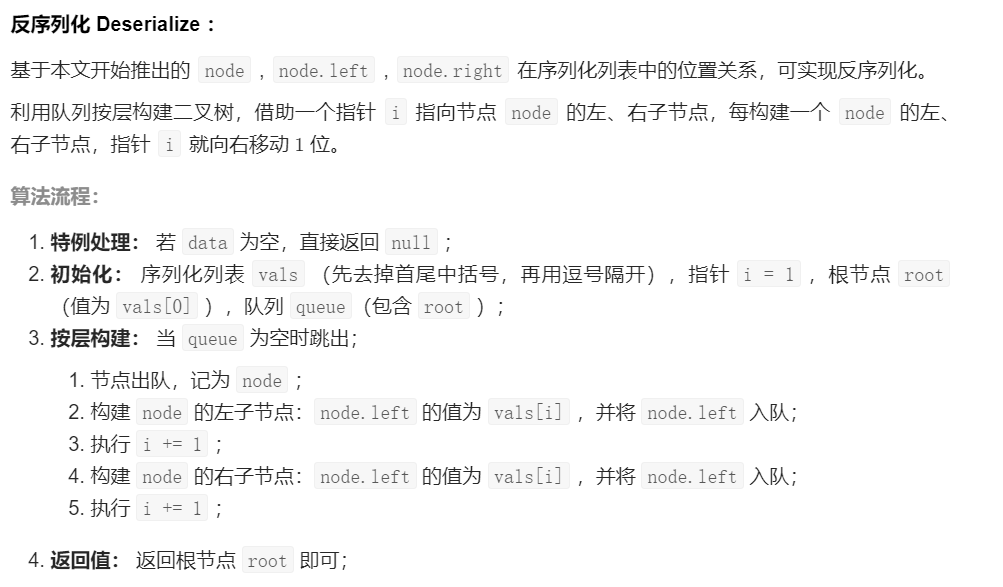


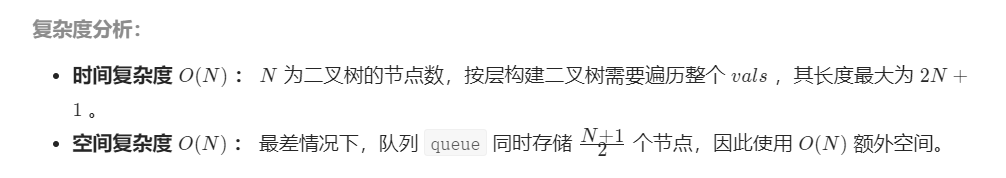


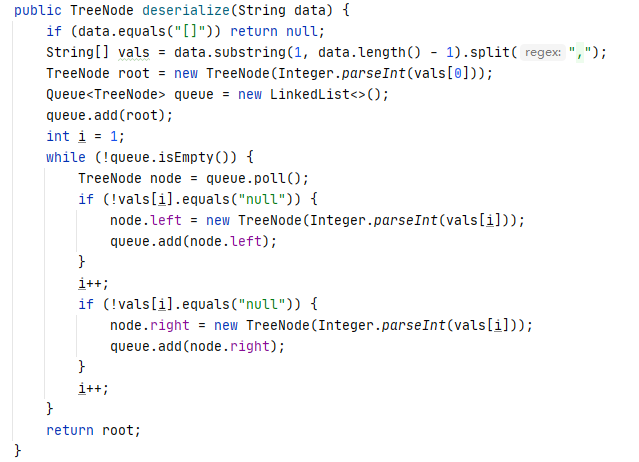




### 2--反序列化





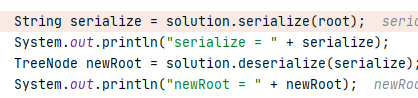


### 3--测试结论

不难发现按**这种约定**,一颗层序遍历为 1--2,3--4,5,6,7 的三层树

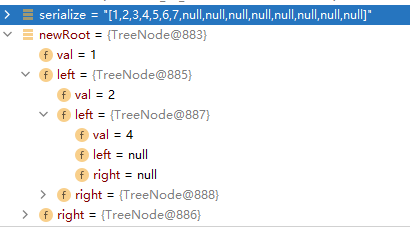
序列化后为:有8个null

--测试用例:

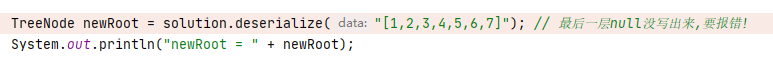


--测试结果:





注意:如果反序列化的时候输入**没有给出叶子节点的8个null子节点**,就会报错!



**总结**:如何**’约定’ 规则**是很重要的,也是比较容易迷惑的点

只要两个函数可以**互相转化**(可逆的感觉)就ok,序列化方案自行规定即可