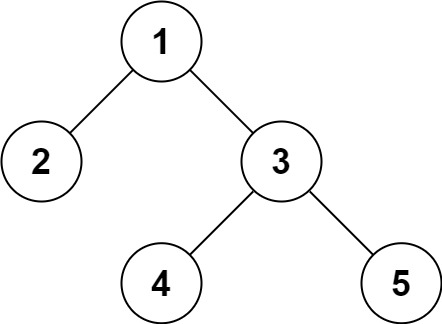
# 题目

请实现两个函数，分别用来序列化和反序列化二叉树。

你需要设计一个算法来实现二叉树的序列化与反序列化。这里不限定你的序列 / 反序列化算法执行逻辑，你只需要保证一个二叉树可以被序列化为一个字符串并且将这个字符串反序列化为原始的树结构。

提示：输入输出格式与LeetCode目前使用的方式一致，详情请参阅LeetCode序列化二叉树的格式。你并非必须采取这种方式，你也可以采用其他的方法解决这个问题。

示例：



输入：root = [1,2,3,null,null,4,5]

输出：[1,2,3,null,null,4,5]

注意：本题与主站 297 题相同：

https://leetcode-cn.com/problems/serialize-and-deserialize-binary-tree/

# 分析

/\*\*

 \* Definition for a binary tree node.

 \* struct TreeNode {

 \*     int val;

 \*     TreeNode \*left;

 \*     TreeNode \*right;

 \*     TreeNode(int x) : val(x), left(NULL), right(NULL) {}

 \* };

 \*/

class Codec {

private:

    TreeNode \*next\_node(stringstream &ss)

    {

        string tmp;

        ss >> tmp;

        if(tmp[0] == 'x')

        {

            return nullptr;

        }

        else

        {

            return new TreeNode(stoi(tmp));

        }

    }

public:

    // Encodes a tree to a single string.

    string serialize(TreeNode\* root) {

        string res;

        TreeNode \*cur = root;

        stack<TreeNode \*> stk;

        while(cur || !stk.empty())

        {

            while(cur)

            {

                res += to\_string(cur->val);

                res.push\_back(' ');

                stk.push(cur);

                cur = cur->left;

            }

            res.push\_back('x');

            res.push\_back(' ');

            cur = stk.top();

            stk.pop();

            cur = cur->right;

        }

        res.push\_back('x');

        res.push\_back(' ');

        return res;

    }

    // Decodes your encoded data to tree.

    TreeNode\* deserialize(string data) {

        stringstream ss(data);

        TreeNode \*root = next\_node(ss);

        TreeNode \*cur = root;

        stack<TreeNode\*> stk;

        while(cur || !stk.empty())

        {

            while(cur)

            {

                stk.push(cur);

                cur = cur->left = next\_node(ss);

            }

            cur = stk.top();

            stk.pop();

            cur = cur->right = next\_node(ss);

        }

        return root;

    }

};

// Your Codec object will be instantiated and called as such:

// Codec codec;

// codec.deserialize(codec.serialize(root));