一、题目说明

题目297. Serialize and Deserialize Binary Tree，序列号和反序列化二叉树。难度是Hard！

二、我的解答

这个题目用了3个小时，用的非递归遍历算法。用stack实现：

```
class Codec{
    public:
        // Encodes a tree to a single string.
        string serialize(TreeNode* root) {
            //先根遍历二叉树
            string res;

            if(root ==NULL) return "[null]";
            stack<TreeNode*> st;
            st.push(root);
            res.append(to_string(root->val));
            res.push_back(',');

            while(! st.empty()){
                TreeNode* tmp = st.top();
                st.pop();

                if(tmp->left !=NULL){
                    res.append(to_string(tmp->left->val));
                    res.push_back(',');
                }else{
                    res.append("null,");
                }

                if(tmp->right !=NULL){
                    st.push(tmp->right);
                    res.append(to_string(tmp->right->val));
                    res.push_back(',');
                }else{
                    res.append("null,");
                }

                if(tmp->left !=NULL){
                    st.push(tmp->left);
                }
            }
            res.pop_back();

            return "["+res+"]";
        }
        // Decodes your encoded data to tree.
        TreeNode* deserialize(string data) {
            data = data.substr(1, data.size() - 2);

            TreeNode* root =NULL,*p;

            int len = data.size();
            if(data.compare("null")==0){
```

```
                return root;
            }

            if(!data.empty()){
                stack<TreeNode*> st;
                int l=0,r=0;
                while(data[r]!=','){
                    r++;
                }
                string str = data.substr(l,r-l);
                //cout<<str<<"->";
                int cur = stoi(str);
                root = new TreeNode(cur);
                st.push(root);
                while(! st.empty()){
                    p = st.top();
                    st.pop();

                    //左子树
                    l=r+1,r=l;
                    while(r<len && data[r]!=','){
                        r++;
                    }
                    str = data.substr(l,r-l);
                    //cout<<str<<"->";
                    if(str.compare("null") !=0){
                        cur = stoi(str);
                        p->left = new TreeNode(cur);
                    }

                    //右子树
                    l=r+1,r=l;
                    while(r<len && data[r]!=','){
                        r++;
                    }
                    str = data.substr(l,r-l);
                    //cout<<str<<"->";
                    if(str.compare("null") !=0){
                        cur = stoi(str);
                        p->right = new TreeNode(cur);
                        st.push(p->right);
                    }

                    if(p->left !=NULL){
                        st.push(p->left);
                    }
                }
                return root;
            }else{
                return root;
            }
        }
};
```

性能如下：

```
Runtime: 40 ms, faster than 83.45% of C++ online submissions for Serialize and Deserialize Binary Tree.
Memory Usage: 32.2 MB, less than 44.83% of C++ online submissions for Serialize and Deserialize Binary Tree.
```

三、优化措施

无