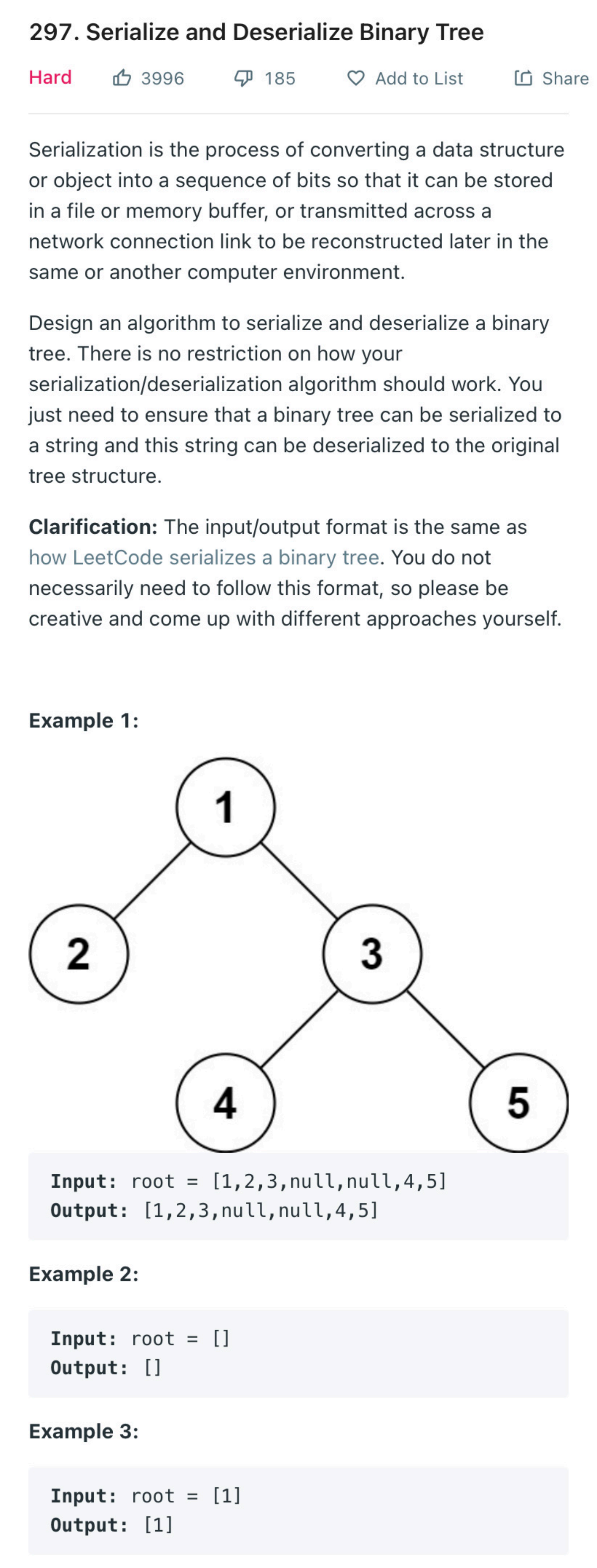


297. Serialize and Deserialize Binary Tree

Hard 3996 185 Add to List Share

Serialization is the process of converting a data structure or object into a sequence of bits so that it can be stored in a file or memory buffer, or transmitted across a network connection link to be reconstructed later in the same or another computer environment.

Design an algorithm to serialize and deserialize a binary tree. There is no restriction on how your serialization/deserialization algorithm should work. You just need to ensure that a binary tree can be serialized to a string and this string can be deserialized to the original tree structure.

Clarification: The input/output format is the same as how LeetCode serializes a binary tree. You do not necessarily need to follow this format, so please be creative and come up with different approaches yourself.

Example 1:

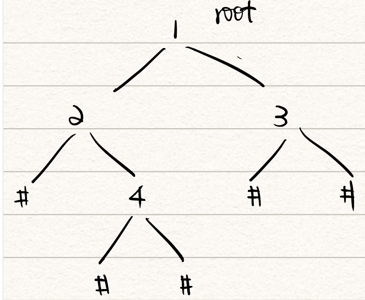
```
Input: root = [1,2,3,null,null,4,5]
Output: [1,2,3,null,null,4,5]
```

Example 2:

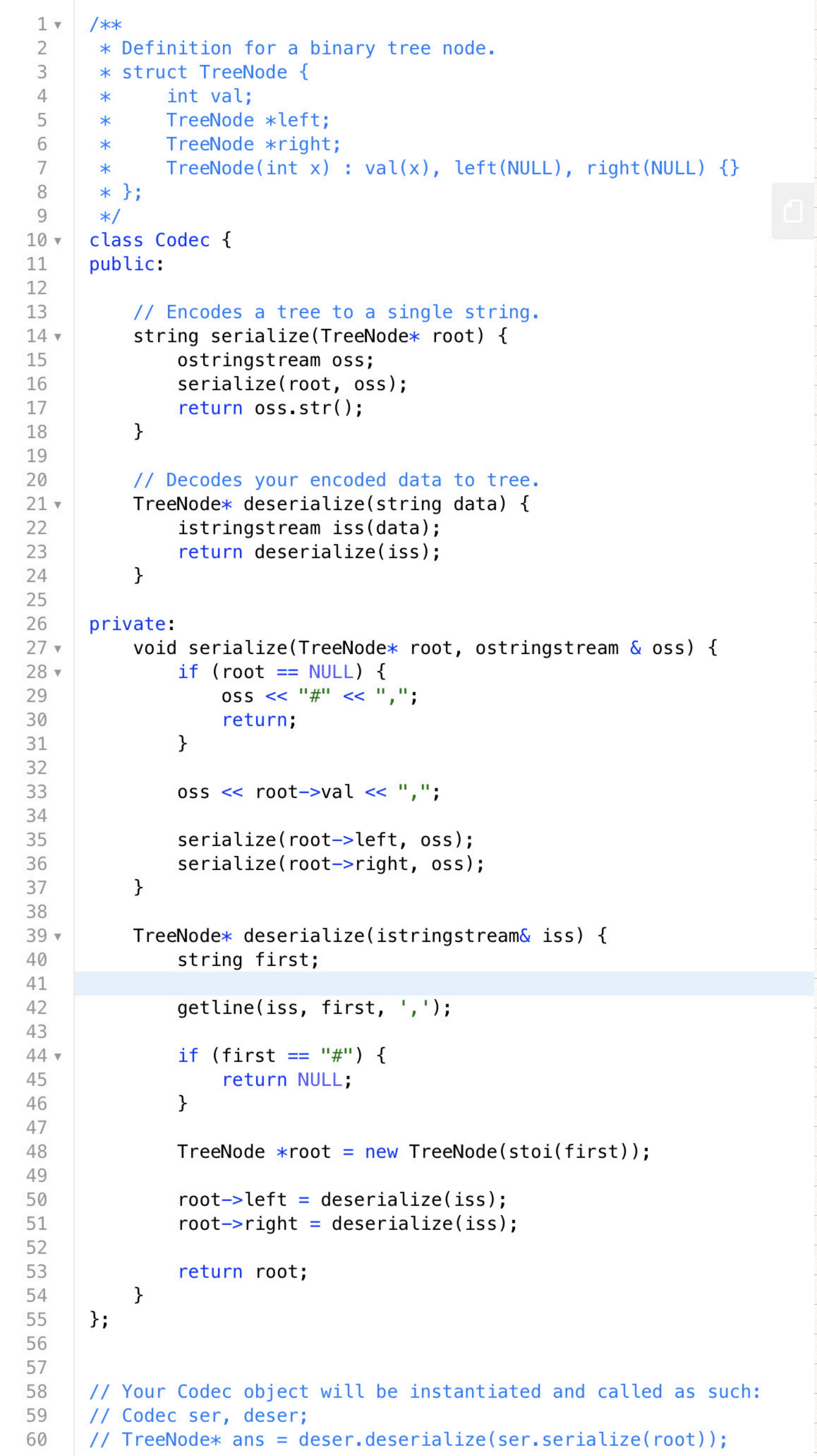
```
Input: root = []
Output: []
```

Example 3:

```
Input: root = [1]
Output: [1]
```



$\overset{root}{1}, \overbrace{2, \#, 4, \#, \#,}^{root.left} \overbrace{3, \#, \#,}^{root.right}$



```
/**
 * 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 {
public:

    // Encodes a tree to a single string.
    string serialize(TreeNode* root) {
        ostringstream oss;
        serialize(root, oss);
        return oss.str();
    }

    // Decodes your encoded data to tree.
    TreeNode* deserialize(string data) {
        istringstream iss(data);
        return deserialize(iss);
    }

private:
    void serialize(TreeNode* root, ostringstream & oss) {
        if (root == NULL) {
            oss << "#" << ",";
            return;
        }

        oss << root->val << ",";

        serialize(root->left, oss);
        serialize(root->right, oss);
    }

    TreeNode* deserialize(istringstream& iss) {
        string first;

        getline(iss, first, ',');

        if (first == "#") {
            return NULL;
        }

        TreeNode *root = new TreeNode(stoi(first));

        root->left = deserialize(iss);
        root->right = deserialize(iss);

        return root;
    }
};


// Your Codec object will be instantiated and called as such:
// Codec ser, deser;
// TreeNode* ans = deser.deserialize(ser.serialize(root));
```