449. Serialize and Deserialize BST

link

用的105的思路, 但是写麻烦了

105 link

```
public class Codec {

    // Encodes a tree to a single string.
    public String serialize(TreeNode root) {
        if(root == null) return "";
        StringBuilder sb = new StringBuilder();
        Stack<TreeNode> stack = new Stack();
        while(root != null || !stack.isEmpty()){
            while(root != null){
                sb.append(root.val + ",");
                stack.push(root);
                root = root.left;
            }
            root = stack.pop();
            root = root.right;
        }
        return sb.toString();
    }

    // Decodes your encoded data to tree.
    public TreeNode deserialize(String data) {
        if(data == "") return null;
        List<Integer> preOrder = getArray(data);
        List<Integer> inOrder = new ArrayList(preOrder);
        Collections.sort(inOrder);
        int size = inOrder.size();
        Map<Integer, Integer> map = new HashMap();
        for(int i = 0; i < size; ++i)
            map.put(inOrder.get(i), i);
        TreeNode root = new TreeNode(preOrder.get(0));
        Stack<TreeNode> stack = new Stack();
        stack.push(root);
        for(int i = 1; i < size; ++i){
            int val = preOrder.get(i);
            TreeNode node = new TreeNode(val);
            if(map.get(stack.peek().val) > map.get(val)){
                stack.peek().left = node;
            }else{
                TreeNode parent = null;
                while(!stack.isEmpty() && map.get(stack.peek().val) < map.get(val)){
                    parent = stack.pop();
                }
                parent.right = node;
            }
            stack.push(node);
        }
        return root;
    }
    public List<Integer> getArray(String data){
        List<Integer> res = new ArrayList();
        int i = 0, j = 0;
        while(i < data.length() && j < data.length()){
            if(data.charAt(i) == ','){
                res.add(Integer.parseInt(data.substring(j,i)));
                j = i + 1;
            }
            ++i;
        }
        return res;
    }
}
```

因为是bst所以自带比较左或者右边不用map和inorder就可以

```
    // Decodes your encoded data to tree.
    public TreeNode deserialize(String data) {
        if(data == "") return null;
        List<Integer> preOrder = getArray(data);
        int size = preOrder.size();
        TreeNode root = new TreeNode(preOrder.get(0));
        Stack<TreeNode> stack = new Stack();
        stack.push(root);
        for(int i = 1; i < size; ++i){
            int val = preOrder.get(i);
            TreeNode node = new TreeNode(val);
            if(stack.peek().val > val){
                stack.peek().left = node;
            }else{
                TreeNode parent = null;
                while(!stack.isEmpty() && stack.peek().val < val){
                    parent = stack.pop();
                }
                parent.right = node;
            }
            stack.push(node);
        }
        return root;
    }
```