Solution@21-08-23

449. serialize and deserialize bst

```
public class Codec {

    // Encodes a tree to a single string.
    public String serialize(TreeNode root) {
        StringBuilder sb = new StringBuilder();
        dfs(root, sb);
        return sb.toString();
    }
    private void dfs(TreeNode node, StringBuilder sb){
        if(node == null) return;
        else{
            sb.append(node.val).append(",");
            dfs(node.left, sb);
            dfs(node.right, sb);
        }
    }

    // Decodes your encoded data to tree.
    public TreeNode deserialize(String data) {
        if(data.trim().length() == 0) return null;
        String[] tokens = data.split(",");
        TreeNode root = new TreeNode(Integer.parseInt(tokens[0]));
        for(int i = 1; i < tokens.length; i++){
            insert(root, Integer.parseInt(tokens[i]));
        }
        return root;
    }
    private TreeNode insert(TreeNode node, int num){
        if(node == null) return new TreeNode(num);
        else if(num < node.val || node.val == num){
            node.left = insert(node.left, num);
        }else if(num > node.val){
            node.right = insert(node.right, num);
        }
        return node;
    }
}
```

349. Intersection of Two Arrays

```
public class Solution {
    public int[] intersection(int[] nums1, int[] nums2) {
        Set<Integer> set = new HashSet<>();
        Set<Integer> intersect = new HashSet<>();
        for (int i = 0; i < nums1.length; i++) {
            set.add(nums1[i]);
        }
        for (int i = 0; i < nums2.length; i++) {
            if (set.contains(nums2[i])) {
                intersect.add(nums2[i]);
            }
        }
        int[] result = new int[intersect.size()];
        int i = 0;
        for (Integer num : intersect) {
            result[i++] = num;
        }
        return result;
    }
}
```