Daily Practice Problem Date 25/11/2023

Problem :1 Serialize and Deserialize

```
package ishwarchavan.com;

import java.util.*;
class TreeNode {     //A binary tree Node has key,pointer to left and right children

	int val;
	TreeNode left;
	TreeNode right;
	TreeNode(int x) { val = x; }
}
public class SerializeAndDeseilize {
	TreeNode root;

	public static String serialize(TreeNode root){   // Encodes a tree to a single string.
		if (root == null) {
			return null;
		}
		Stack<TreeNode> s = new Stack<>();
		s.push(root);

		List<String> l = new ArrayList<>();
		while (!s.isEmpty()) {
			TreeNode t = s.pop();

			if (t == null) {   // If current node is NULL, store marker
				l.add("#");
			}
			else {
				l.add("" + t.val);   // Else, store current node and recur for its children
				s.push(t.right);
				s.push(t.left);
			}
		}
		return String.join(",", l);
	}

	static int t;

	public static TreeNode deserialize(String data)  // Decodes your encoded data to tree.
	{
		if (data == null)
			return null;
		t = 0;
		String[] arr = data.split(",");
		return helper(arr);
	}

	public static TreeNode helper(String[] arr){   //function created
		if (arr[t].equals("#"))
			return null;
		TreeNode root = new TreeNode(Integer.parseInt(arr[t]));  // Create node with this item and recur for children
		t++;
		root.left = helper(arr);
		t++;
		root.right = helper(arr);
		return root;
	}
	static void inorder(TreeNode root){// A simple inorder traversal used for testing the constructed tree
```

```
            if (root != null) {
                  inorder(root.left);
                  System.out.print(root.val + " ");
                  inorder(root.right);
            }
      }
      public static void main(String args[]){     //main program started

       SerializeAndDeseilize  tree = new  SerializeAndDeseilize();   // Construct a
tree shown in the above figure
            tree.root = new TreeNode(20);
            tree.root.left = new TreeNode(8);
            tree.root.right = new TreeNode(22);
            tree.root.left.left = new TreeNode(4);
            tree.root.left.right = new TreeNode(12);
            tree.root.left.right.left = new TreeNode(10);
            tree.root.left.right.right = new TreeNode(14);

            String serialized = serialize(tree.root);
            System.out.println("Serialized view of the tree:");
            System.out.println(serialized);
            System.out.println();


            TreeNode t = deserialize(serialized);   // Deserialize the stored tree into
root1

            System.out.println("Inorder Traversal of the tree constructed"+ " from
serialized String:");
            inorder(t);
      }
}
```

Problem :1 Intersection of two arrays

```
package ishwarchavan.com;

public class IntersectionOfTwoArrays {      //class created

         public static void main(String args[]) {      //main program started
            int myArray1[] = {23, 36, 96, 78, 55};    //arrar crated
            int myArray2[] = {78, 45, 19, 73, 55};
            System.out.println("Intersection of the two arrays ::");    //printing
message

            for(int i = 0; i<myArray1.length; i++ ) {    //outer loop  iterating
               for(int j = 0; j<myArray2.length; j++) {   //inner loop
                  if(myArray1[i]==myArray2[j]) {     //condition checking
                     System.out.println(myArray2[j]);
                  }
               }
            }
         }
      }
```