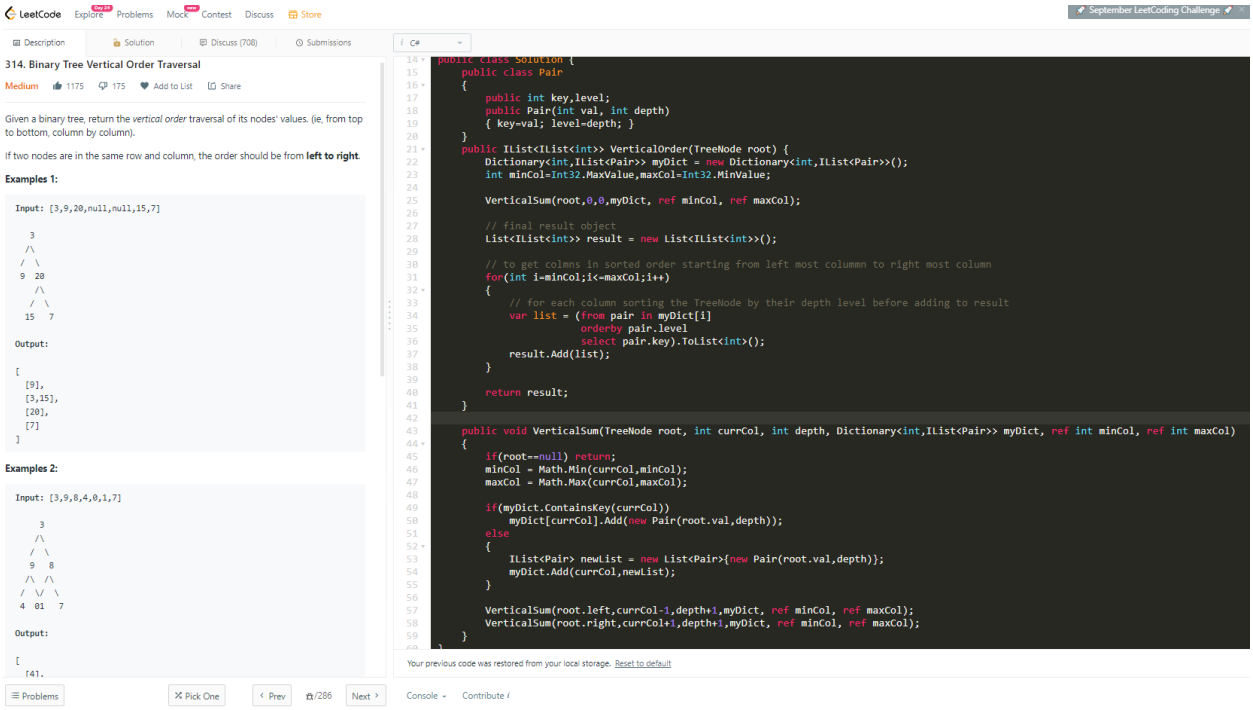
// https://leetcode.com/problems/binary-tree-vertical-order-traversal/




```
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     public int val;
 *     public TreeNode left;
 *     public TreeNode right;
 *     public TreeNode(int val=0, TreeNode left=null, TreeNode right=null) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
public class Solution {
    public class Pair
    {
        public int key,level;
        public Pair(int val, int depth)
        { key=val; level=depth; }
    }
    public IList<IList<int>> VerticalOrder(TreeNode root) {
        Dictionary<int,IList<Pair>> myDict = new Dictionary<int,IList<Pair>>();
        int minCol=Int32.MaxValue,maxCol=Int32.MinValue;

        VerticalSum(root,0,0,myDict, ref minCol, ref maxCol);
```

```
        // final result object
        List<IList<int>> result = new List<IList<int>>();

        // to get colmns in sorted order starting from left most columnn to right most column
        for(int i=minCol;i<=maxCol;i++)
        {
            // for each column sorting the TreeNode by their depth level before adding to result
            var list = (from pair in myDict[i]
                        orderby pair.level
                        select pair.key).ToList<int>();
            result.Add(list);
        }
        return result;
    }
    // DFS based approach
    public void VerticalSum(TreeNode root, int currCol, int depth, Dictionary<int,IList<Pair>> myDict, ref
int minCol, ref int maxCol)
    {
        if(root==null) return;
        minCol = Math.Min(currCol,minCol);
        maxCol = Math.Max(currCol,maxCol);

        if(myDict.ContainsKey(currCol))
            myDict[currCol].Add(new Pair(root.val,depth));
        else
        {
            IList<Pair> newList = new List<Pair>{new Pair(root.val,depth)};
            myDict.Add(currCol,newList);
        }

        VerticalSum(root.left,currCol-1,depth+1,myDict, ref minCol, ref maxCol);
        VerticalSum(root.right,currCol+1,depth+1,myDict, ref minCol, ref maxCol);
    }
}
```

Binary Tree Vertical Order Traversal

Submission Detail

212 / 212 test cases passed.

Status: Accepted

Runtime: **252 ms**

Memory Usage: **31.5 MB**

Submitted: **0 minutes ago**

Accepted Solutions Runtime Distribution