Question : Leetcode for Maximum Width of Binary Tree

Solution:

```
class Solution {
    public int widthOfBinaryTree(TreeNode root) {
        if (root == null)
      return 0;


    int ans = 0;
    Deque<Pair<TreeNode, Integer>> q = new ArrayDeque<>(); // {node, index}
    q.offer(new Pair<>(root, 1));


    while (!q.isEmpty()) {
      final int offset = q.peekFirst().getValue() * 2;
      ans = Math.max(ans, q.peekLast().getValue() - q.peekFirst().getValue() + 1);
      for (int sz = q.size(); sz > 0; --sz) {
        final TreeNode node = q.peekFirst().getKey();
        final int index = q.pollFirst().getValue();
        if (node.left != null)
          q.offer(new Pair<>(node.left, index * 2 - offset));
        if (node.right != null)
          q.offer(new Pair<>(node.right, index * 2 + 1 - offset));
      }
    }


    return ans;


    }
}
```

Question: Leet code for Merge Two 2D Arrays by Summing Values

Solution:

```
class Solution {
    public int[][] mergeArrays(int[][] nums1, int[][] nums2) {
        final int kMax = 1000;
    List<int[]> ans = new ArrayList<>();
    int[] count = new int[kMax + 1];


    addCount(nums1, count);
    addCount(nums2, count);


    for (int i = 1; i <= kMax; ++i)
      if (count[i] > 0)
        ans.add(new int[] {i, count[i]});


    return ans.stream().toArray(int[][] ::new);
  }


  private void addCount(int[][] nums, int[] count) {
    for (int[] idAndVal : nums) {
      final int id = idAndVal[0];
      final int val = idAndVal[1];
      count[id] += val;
    }}
}
```