Leetcode

Leetcode Solution 1:-

```
class Solution {
    public int singleNonDuplicate(int[] nums) {
        int left = 0, right = nums.length - 1;
        while (left < right) {
            int mid = (left + right) >> 1;
            // if ((mid % 2 == 0 && nums[mid] != nums[mid + 1]) || (mid % 2
== 1 && nums[mid] !=
            // nums[mid - 1])) {
            if (nums[mid] != nums[mid ^ 1]) {
                right = mid;
            } else {
                left = mid + 1;
            }
        }
        return nums[left];
    }
}
```

Leetcode Solution 2:-

```
class Solution {
    public List<Integer> largestValues(TreeNode root) {
        List<Integer> ans = new ArrayList<>();
        if (root == null) {
            return ans;
        }
        Deque<TreeNode> q = new ArrayDeque<>();
        q.offer(root);
        while (!q.isEmpty()) {
            int t = q.peek().val;
            for (int i = q.size(); i > 0; --i) {
                TreeNode node = q.poll();
                t = Math.max(t, node.val);
                if (node.left != null) {
                    q.offer(node.left);
                }
                if (node.right != null) {
                    q.offer(node.right);
                }
            }
            ans.add(t);
        }
        return ans;
    }
}
```