We provide C++ and Java codes for your reference.

|  |
| --- |
| /\*\*  \* Definition for a binary tree node.  \* public class TreeNode {  \* int val;  \* TreeNode left;  \* TreeNode right;  \* TreeNode(int x) { val = x; }  \* }  \*/  class Solution {  public List<List<Integer>> levelOrder(TreeNode root) {  List<List<Integer>> ans = new ArrayList<>();  Queue<TreeNode> q = new LinkedList<>();  if (root != null) {  q.offer(root);  }  TreeNode cur;  while (!q.isEmpty()) {  int size = q.size();  List<Integer> subAns = new LinkedList<Integer>();  for (int i = 0; i < size; ++i) { // traverse nodes in the same level  cur = q.poll();  subAns.add(cur.val); // visit the root  if (cur.left != null) {  q.offer(cur.left); // push left child to queue if it is not null  }  if (cur.right != null) {  q.offer(cur.right); // push right child to queue if it is not null  }  }  ans.add(subAns);  }  return ans;  }  } |

Since each node in the tree will be pushed into the queue exactly once, the time complexity for level-order traversal is O(N), where N is the total number of nodes in the tree.

What about the space complexity? We have to maintain a queue to help us to do the traversal. And the size of the queue will be at most N because each node will be pushed into the queue exactly once. Therefore, the space complexity of level-order traversal is also O(N).