



Assignment Solutions | Binary Trees 2 | Week 17

1. Diameter of Binary Tree $O(n)$ approach using a user defined data type [LeetCode 543]

Sol:

```
class Solution {
public:
    int ans = 0;
    int dfs(TreeNode* root) {
        if(!root) return 0;
        int l = dfs(root->left), r = dfs(root->right);
        ans = max(ans, l+r);
        return max(l,r)+1;
    }
    int diameterOfBinaryTree(TreeNode* root) {
        dfs(root);
        return ans;
    }
};
```

2. Level Order Traversal (Using Queue) [LeetCode 102]

Sol:

```

class Solution {
public:
    vector<vector<int>> levelOrder(TreeNode* root) {
        vector<vector<int>> ans;
        queue<TreeNode*> q;
        if(!root) return ans;
        q.push(root);
        while(q.size()) {
            int n = q.size(); vector<int> v;
            for(int i = 0; i < n; i++) {
                auto f = q.front(); q.pop();
                v.push_back(f -> val);
                if(f -> left) q.push(f -> left);
                if(f -> right) q.push(f -> right);
            }
            ans.push_back(v);
        }
        return ans;
    }
};

```

3. *Level order traversal (Right to Left)

Sol:

```

class Solution {
public:
    vector<vector<int>> levelOrder(TreeNode* root) {
        vector<vector<int>> ans;
        queue<TreeNode*> q;
        if(!root) return ans;
        q.push(root);
        while(q.size()) {
            int n = q.size(); vector<int> v;
            for(int i = 0; i < n; i++) {
                auto f = q.front(); q.pop();
                v.push_back(f -> val);
                if(f -> right) q.push(f -> right);
                if(f -> left) q.push(f -> left);
            }
            ans.push_back(v);
        }
        return ans;
    }
};

```

4. Zigzag Level Order Traversal [LeetCode 103]

Sol:

```

class Solution {
public:
    vector<vector<int>> zigzagLevelOrder(TreeNode* root) {
        if(!root) return {};
        queue<TreeNode*> q;
        q.push(root);
        vector<vector<int>> ans;
        while(q.size()) {
            int n = q.size();
            vector<int> temp;
            for(int i = 0; i < n; i++) {
                TreeNode* f = q.front(); q.pop();
                temp.push_back(f->val);
                if(f->left) q.push(f->left);
                if(f->right) q.push(f->right);
            }
            if(ans.size() % 2 == 1) {
                reverse(temp.begin(), temp.end());
            }
            ans.push_back(temp);
        }
        return ans;
    }
};

```

Note:- Please try to invest time doing the assignments which are necessary to build a strong foundation. Do not directly Copy Paste using Google or ChatGPT. Please use your brain 😊.
