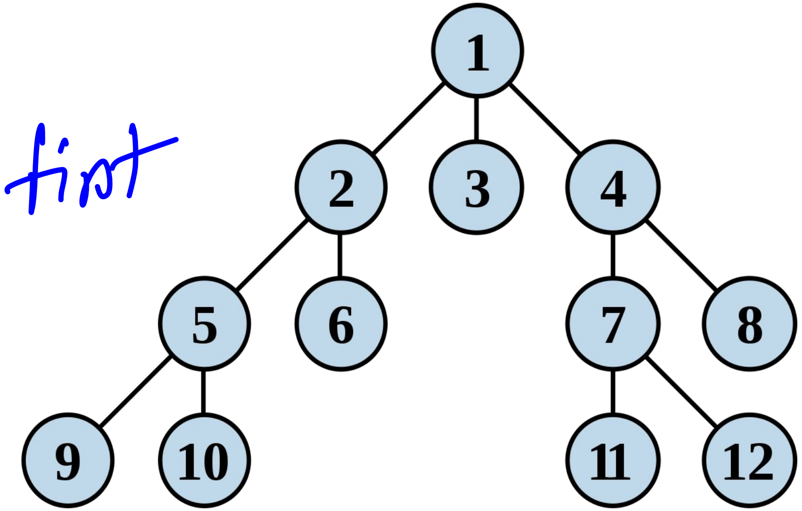
Trees 2

BFS Traversal in a Tree

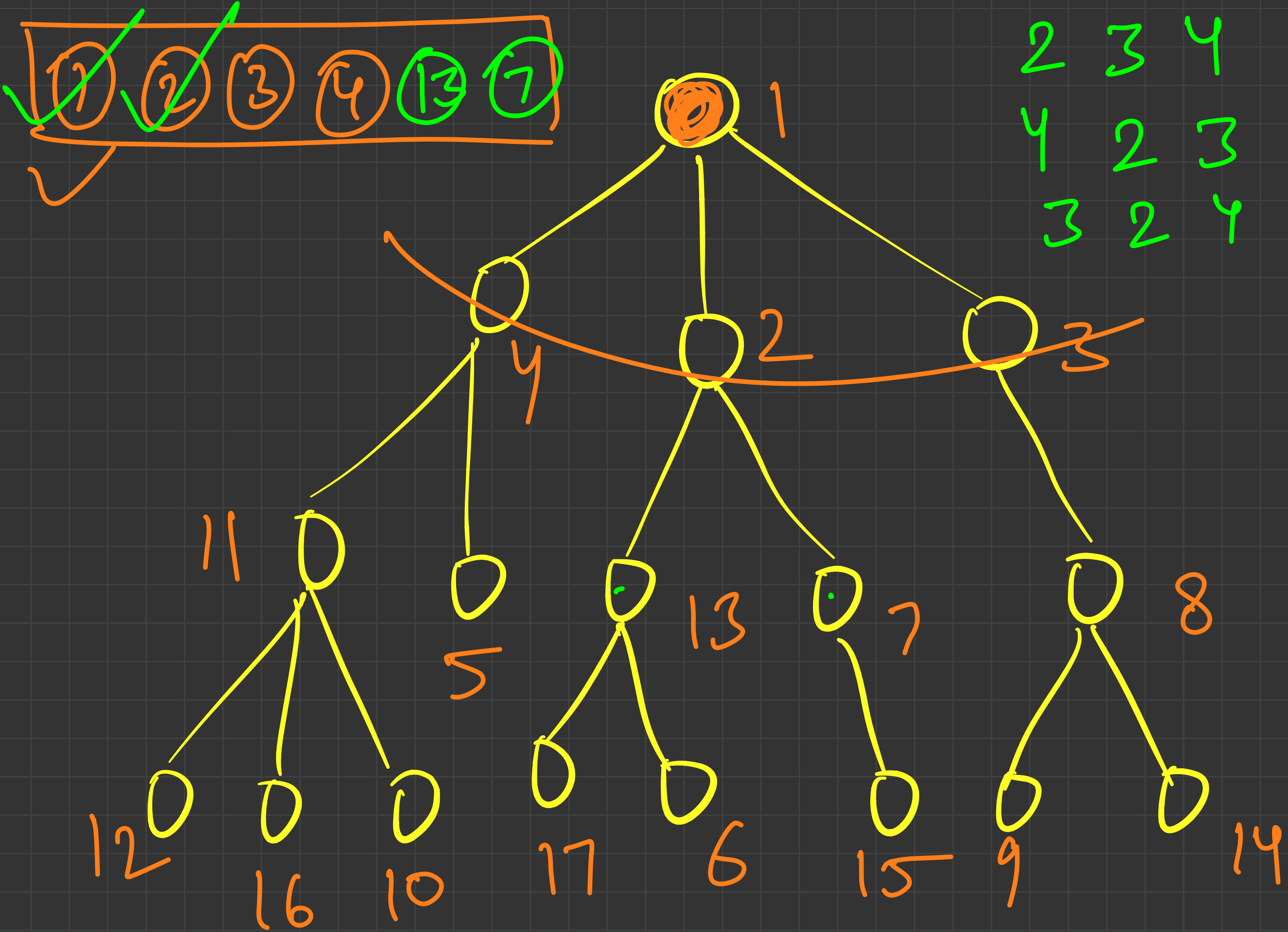
BFS
~~B~~readth first Search

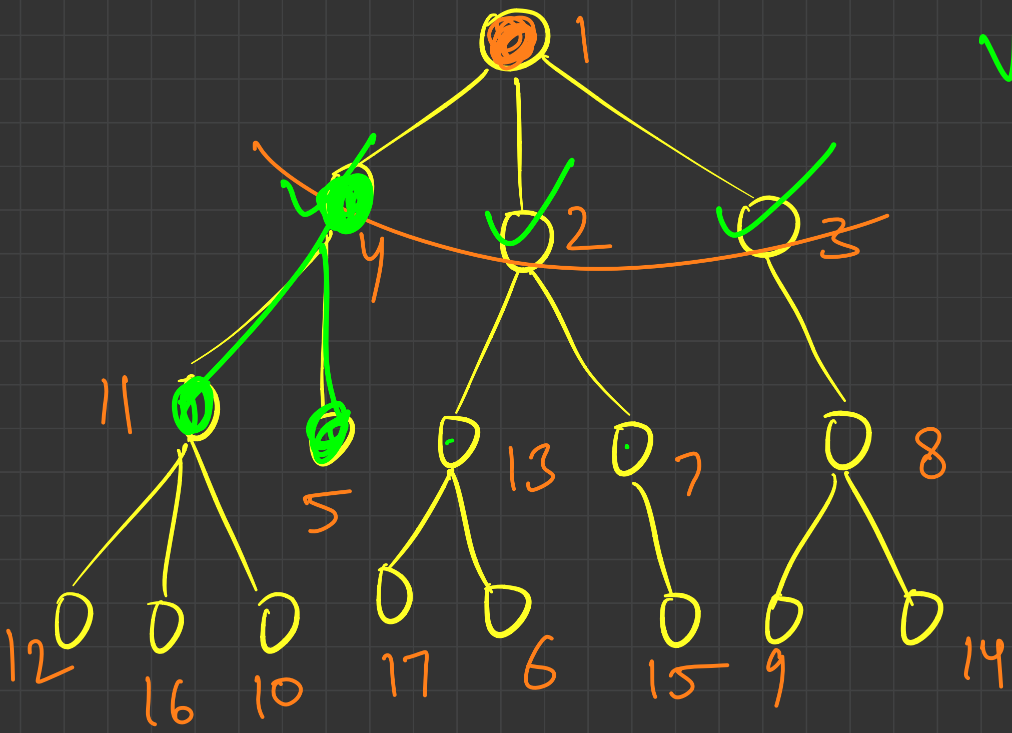


DFS

Depth first Search

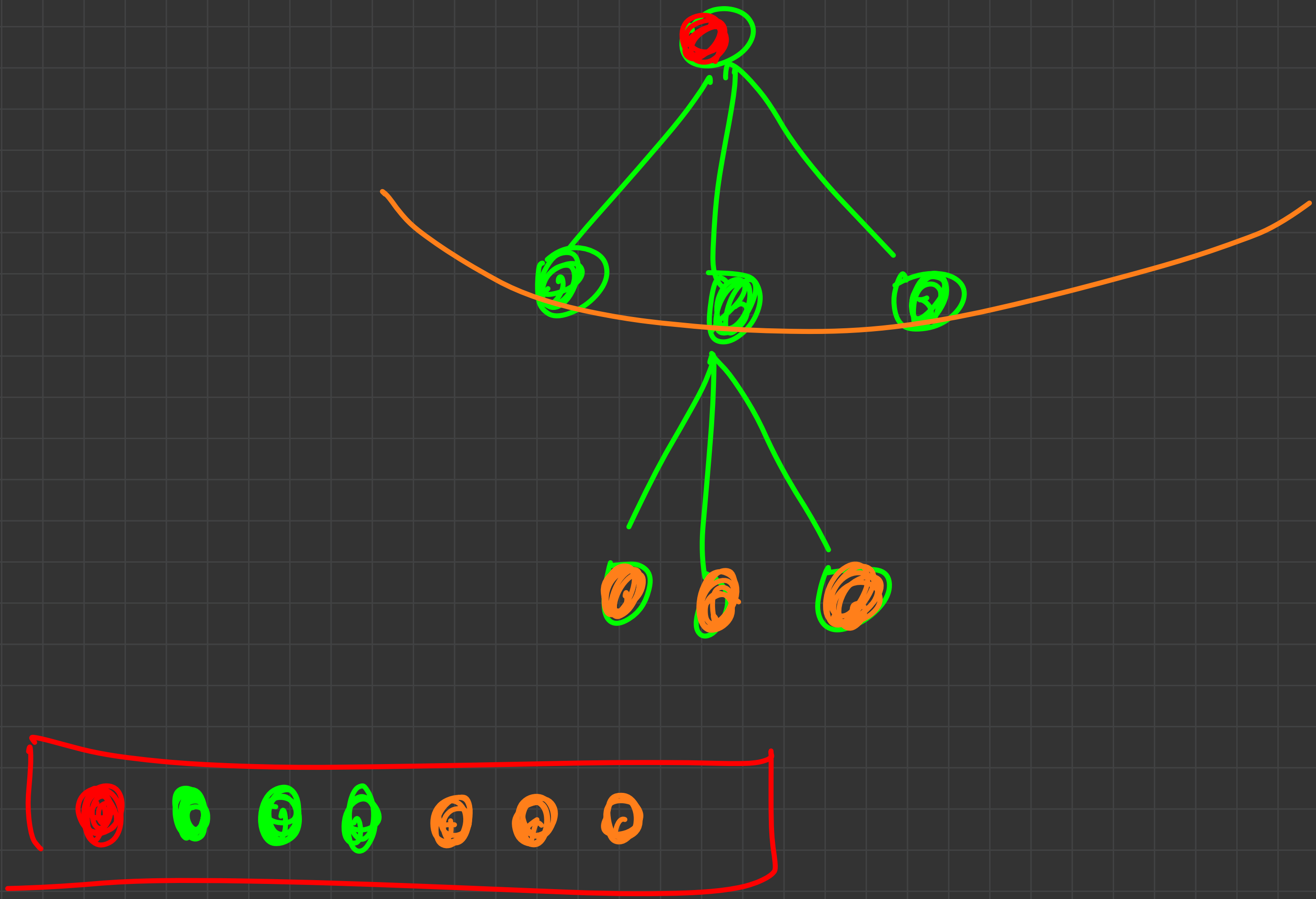
Nodes are numbered in the order in which they are visited


1
2
3
4
13
7
1
4
2
3
11
5
13
7
8
12
16
10
17
6
15
9
14
2 3 4
4 2 3
3 2 4



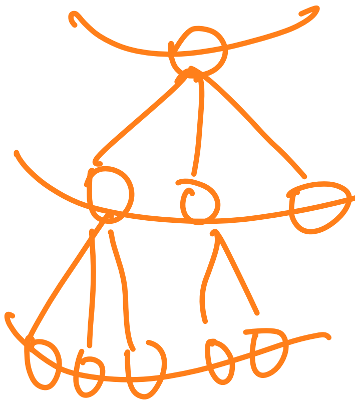
vis [1, 4, 2, 3

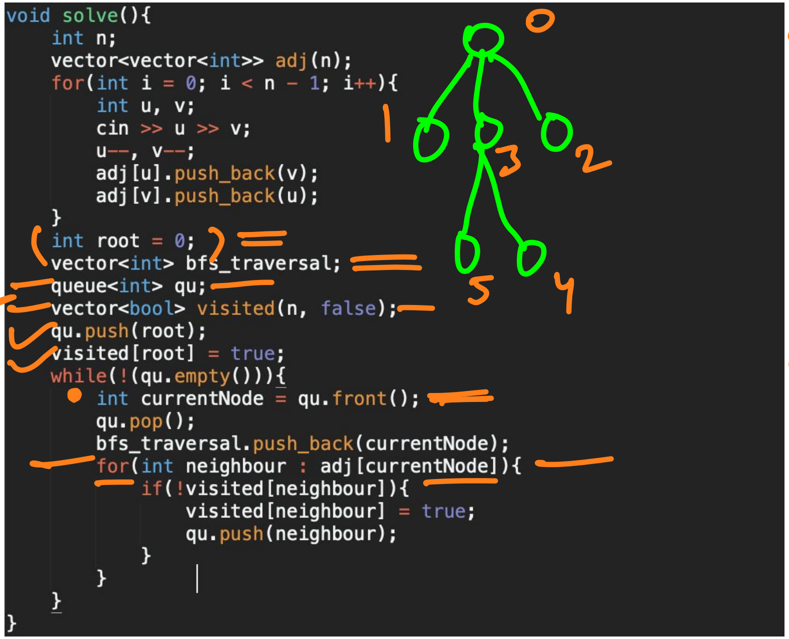
BFS : $\boxed{1}$ $\boxed{4}$ $\boxed{2}$ $\boxed{3}$ $\boxed{11}$ $\boxed{5}$



BFS Traversal in a Tree

Implementation:





```cpp
void solve(){
    int n;
    vector<vector<int>> adj(n);
    for(int i = 0; i < n - 1; i++){
        int u, v;
        cin >> u >> v;
        u--, v--;
        adj[u].push_back(v);
        adj[v].push_back(u);
    }
    int root = 0;
    vector<int> bfs_traversal;
    queue<int> qu;
    vector<bool> visited(n, false);
    qu.push(root);
    visited[root] = true;
    while(!(qu.empty())){
        int currentNode = qu.front();
        qu.pop();
        bfs_traversal.push_back(currentNode);
        for(int neighbour : adj[currentNode]){
            if(!visited[neighbour]){
                visited[neighbour] = true;
                qu.push(neighbour);
            }
        }
    }
}
```

queue $\boxed{2\ 4\ 5}$

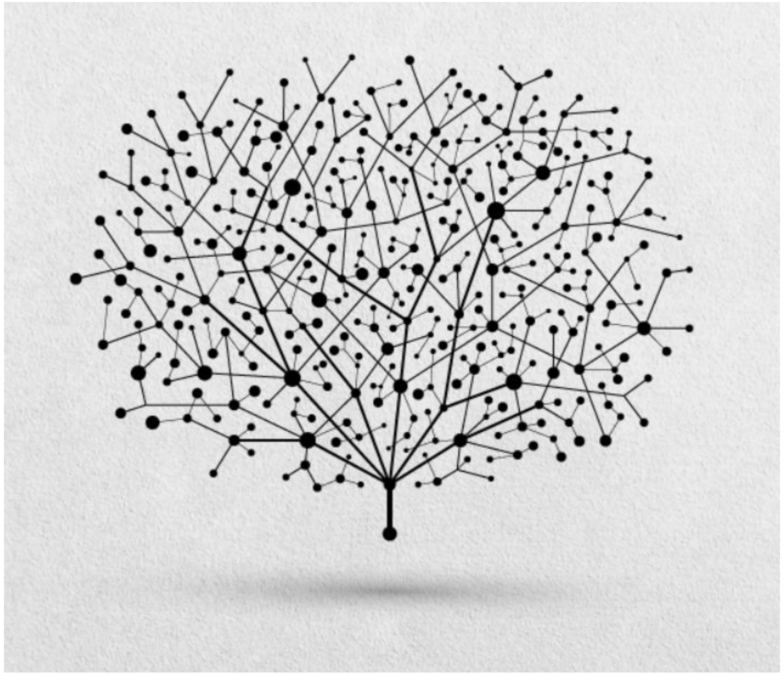
0 1 3 2 4 5

vis | T | T | T | T | T | T |

O(N)

Time Complexity: O(N)

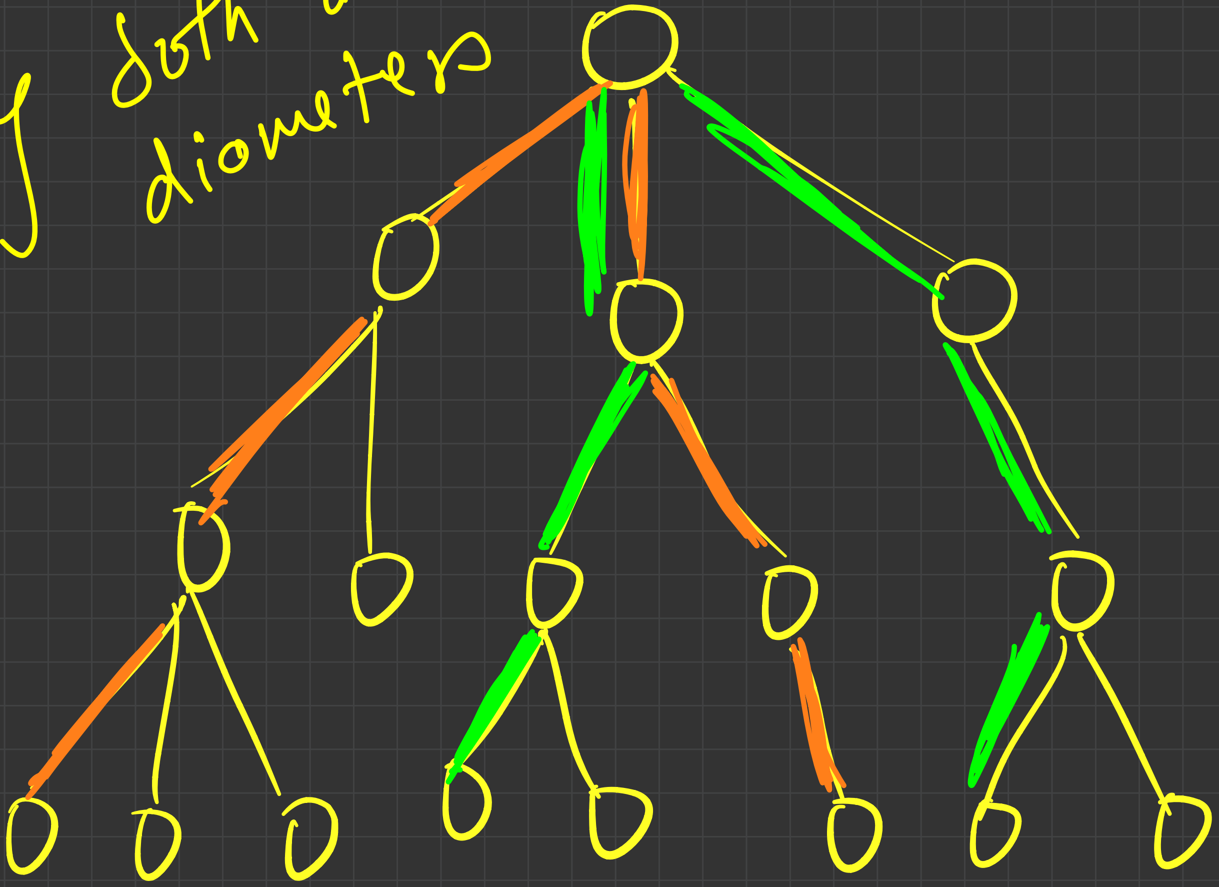
Diameter of a Tree



Diameter of a tree = Maximum distance between any 2 nodes in the tree

Problem: Link

🟩
🟧 } both are diameters



① Pick a random node X from the tree ✓

② Do a DFS from X and find the node with maximum distance from X and call it Y

$\underline{O(n)}$

③ Do a dfs from Y and find the node with max distance from Y call it

Z

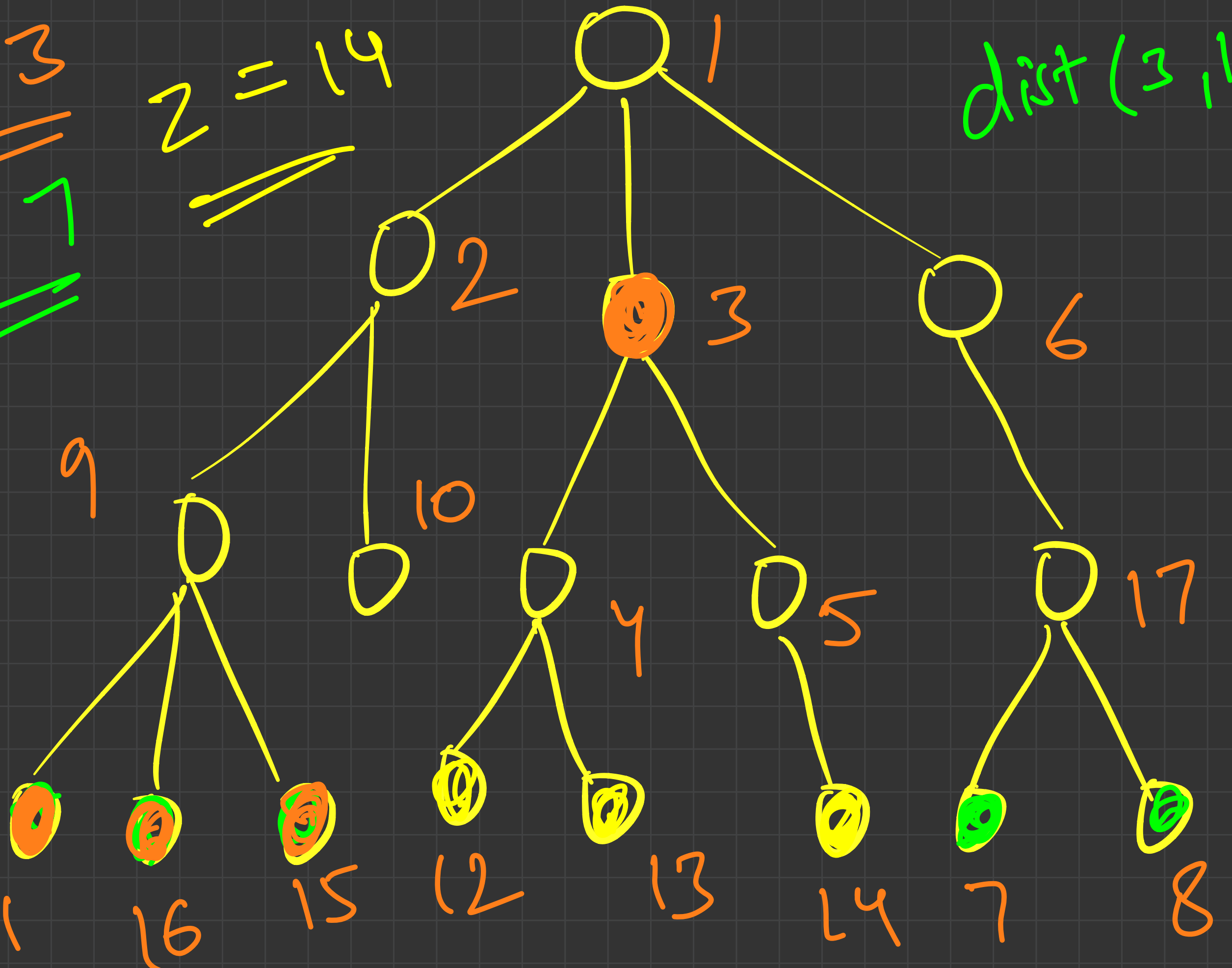
dist (Y, Z) = diameter

$x = 3$

$y = 7$

$z = 14$

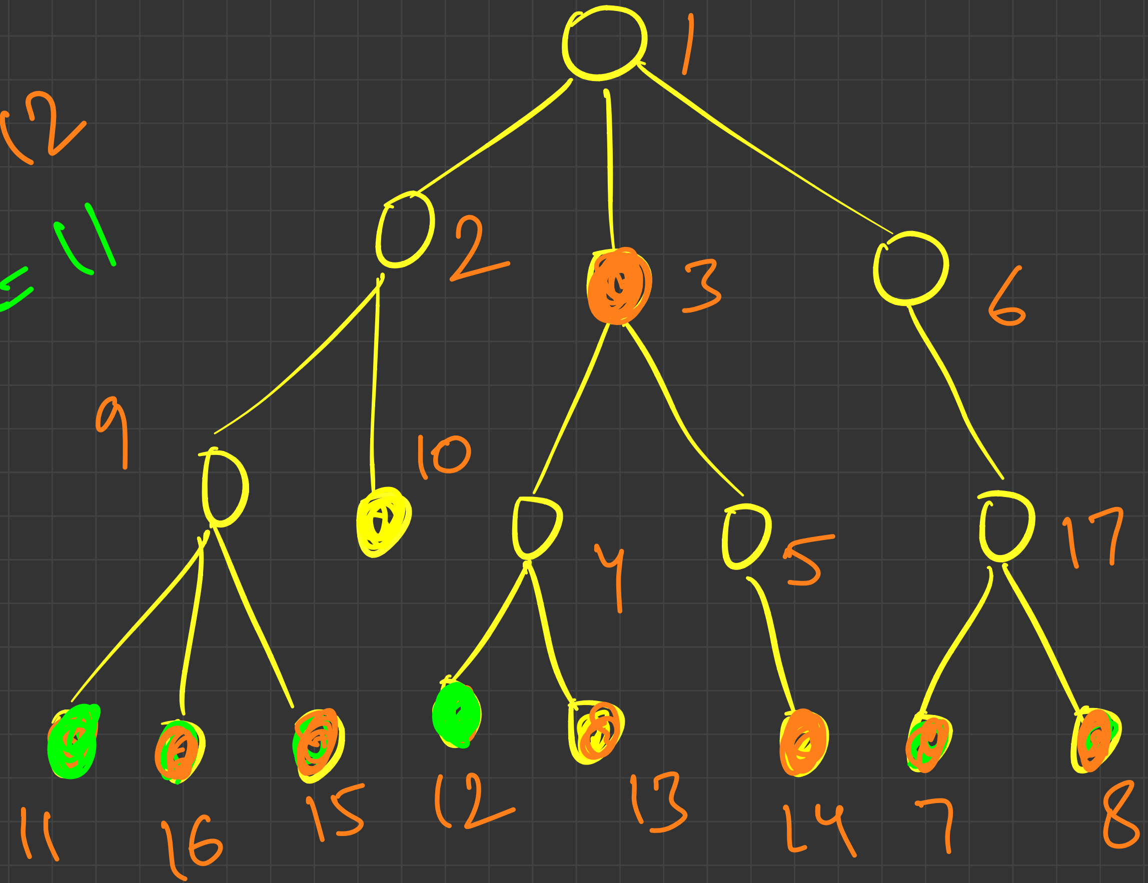
$dist(3, 11) = 4$



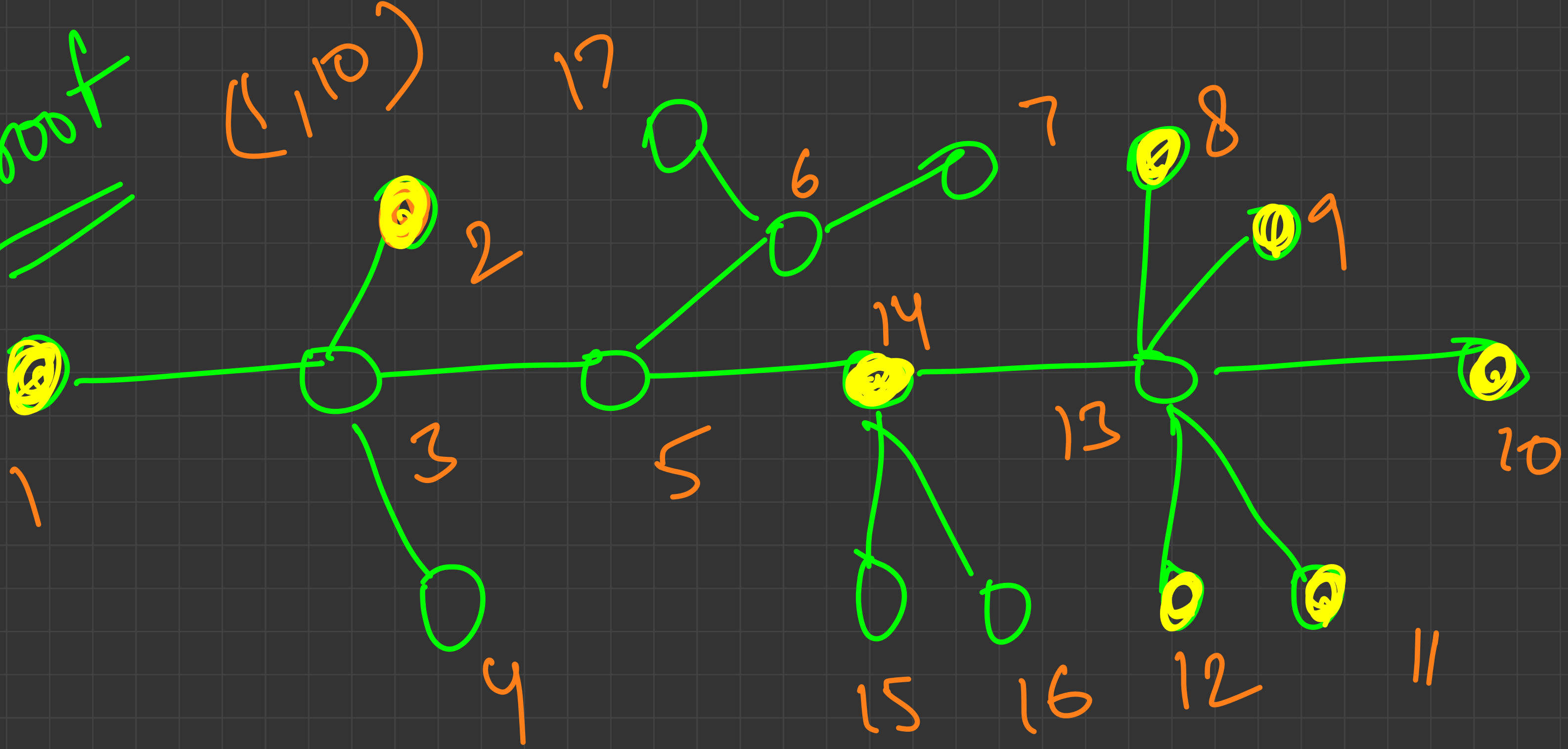
$X = 10$

$Y = 12$

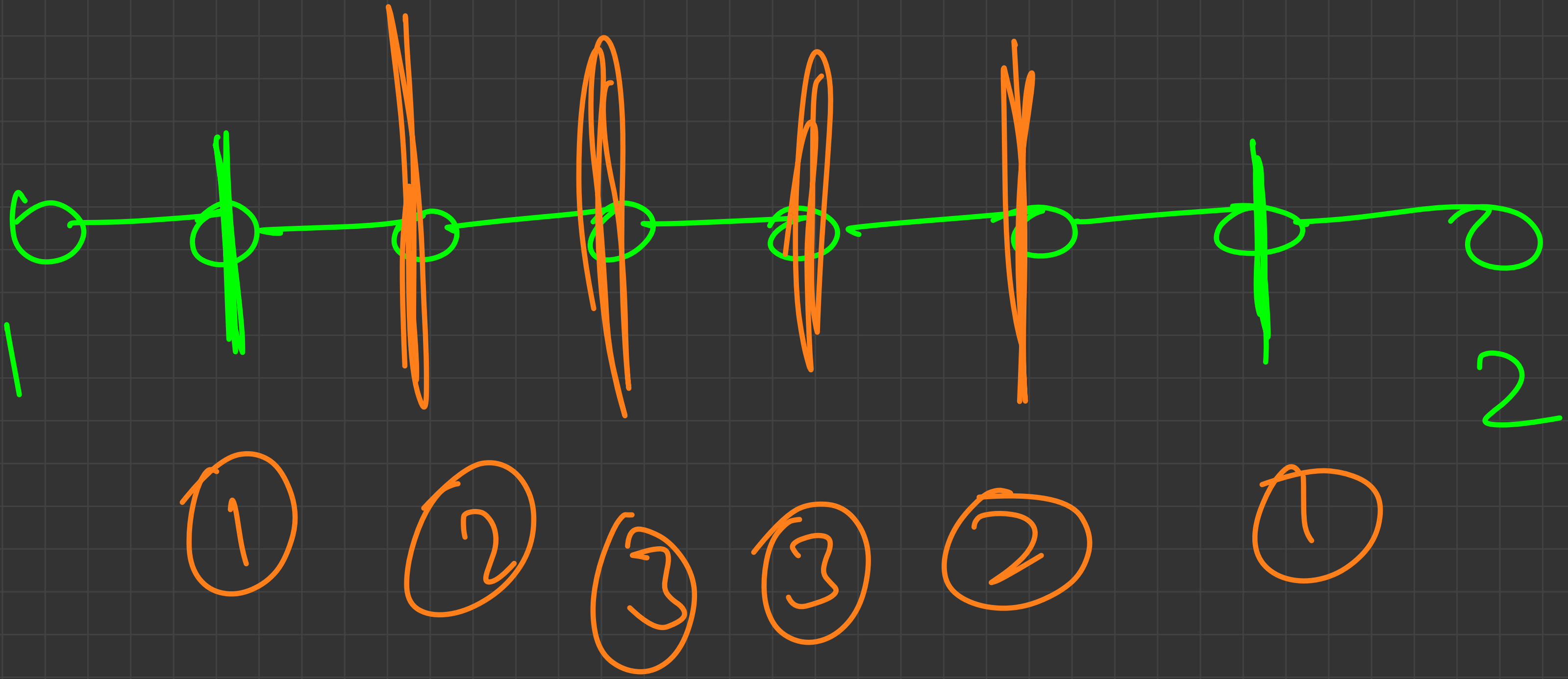
$Z = 11$

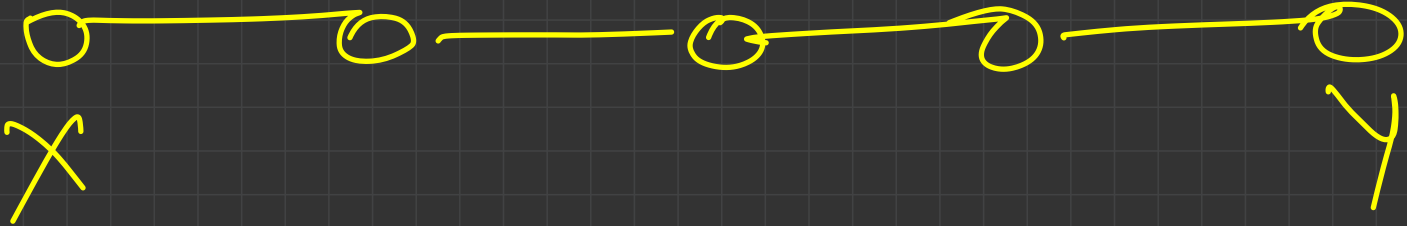


Proof
=

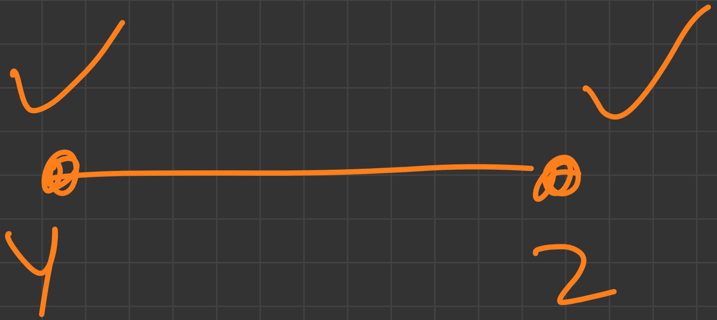


DFS from a random node X, the node which has the max distance from X will always be the end point


1
2
1
2
3
3
2
1

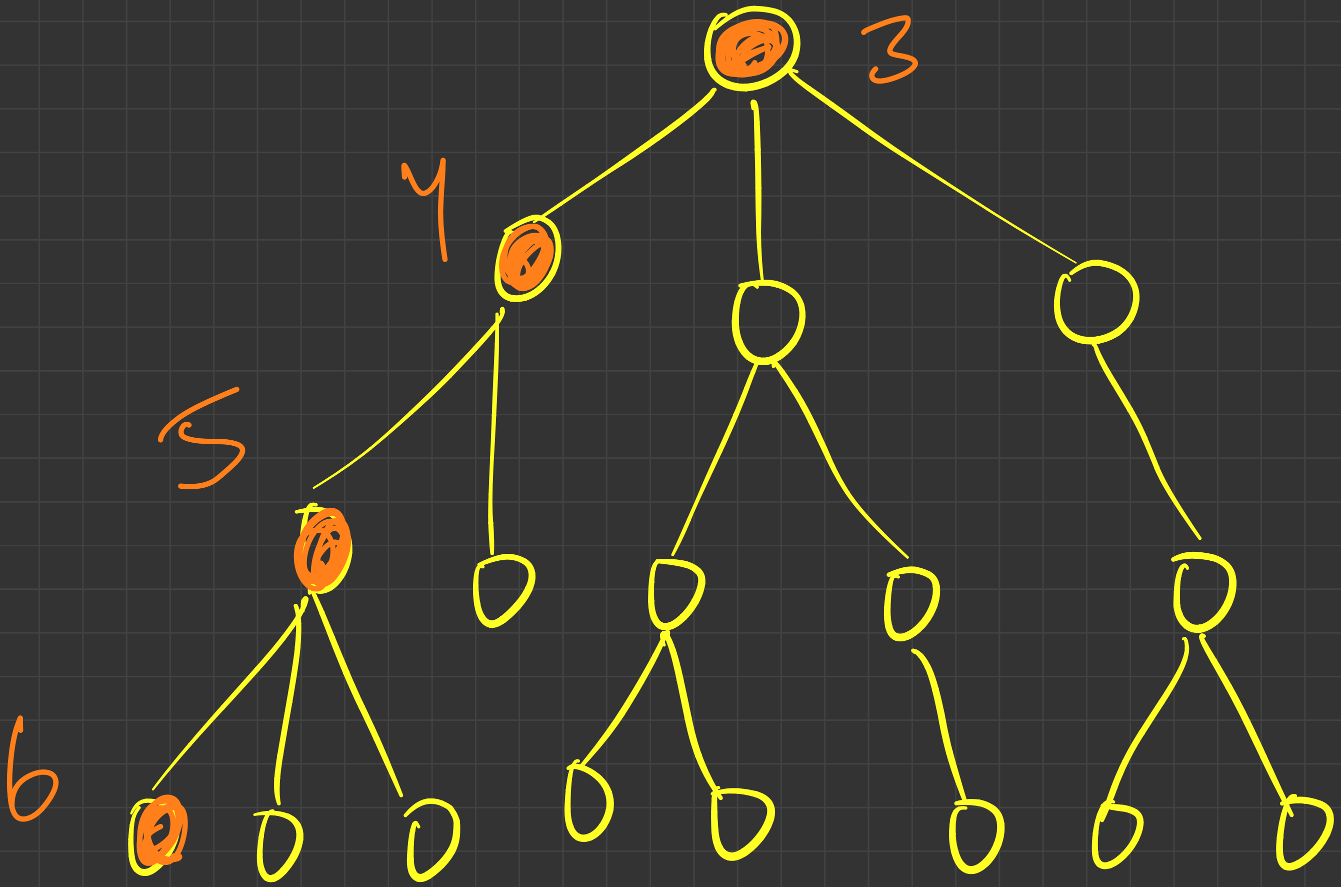

x
y

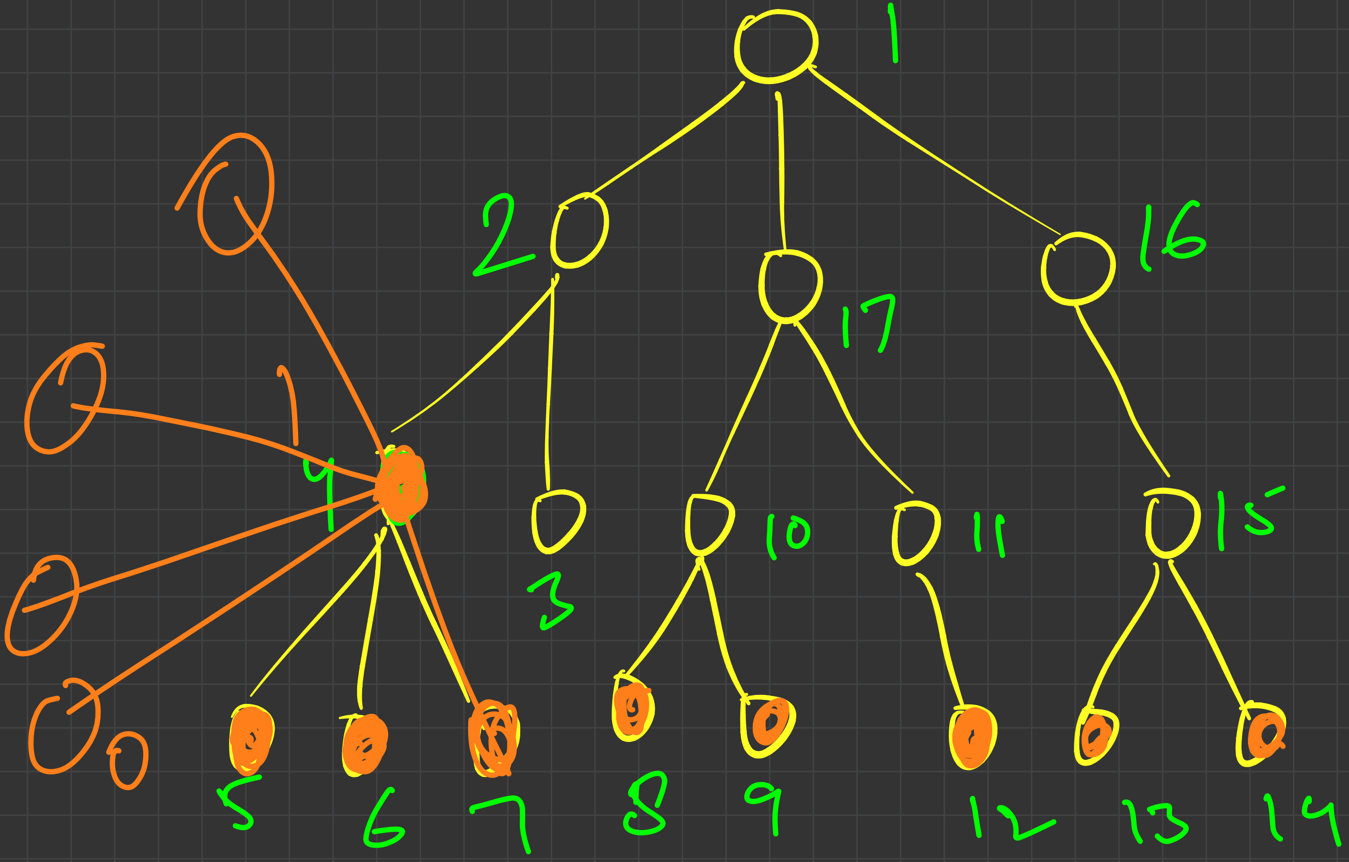
from every node the farthest node will be one of the end point of the diameter



O(n) dimeter
O(n)
O(n)

of a diameter the tree


3
4
5
6



$\underline{\underline{O(n)}}$

Ancestor - Descendant Problem

Given a rooted tree with N nodes and Q queries.

For each query of the form X, Y check whether X is an ancestor of Y or not

N <= 1e5, Q <= 1e5

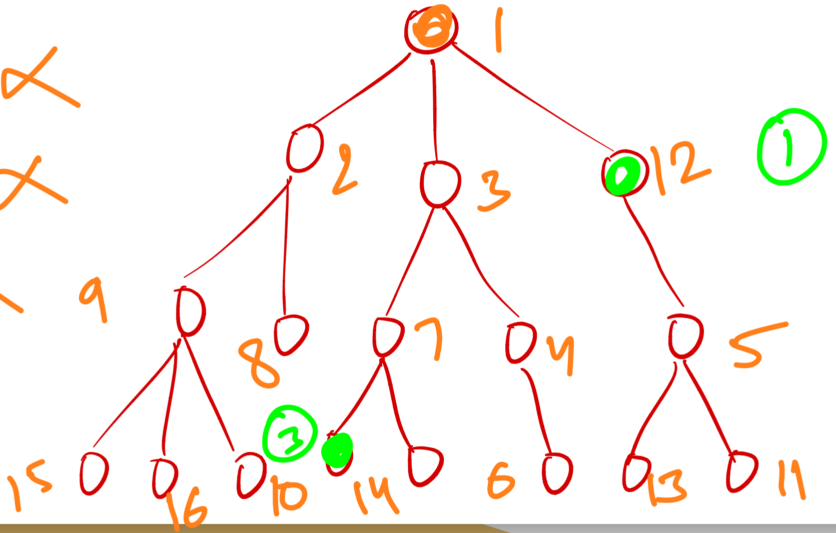
$N \cdot Q \; \times$

$Q^2 \; \times$

$N^2 \; \times \; 9$

X, Y

$=$

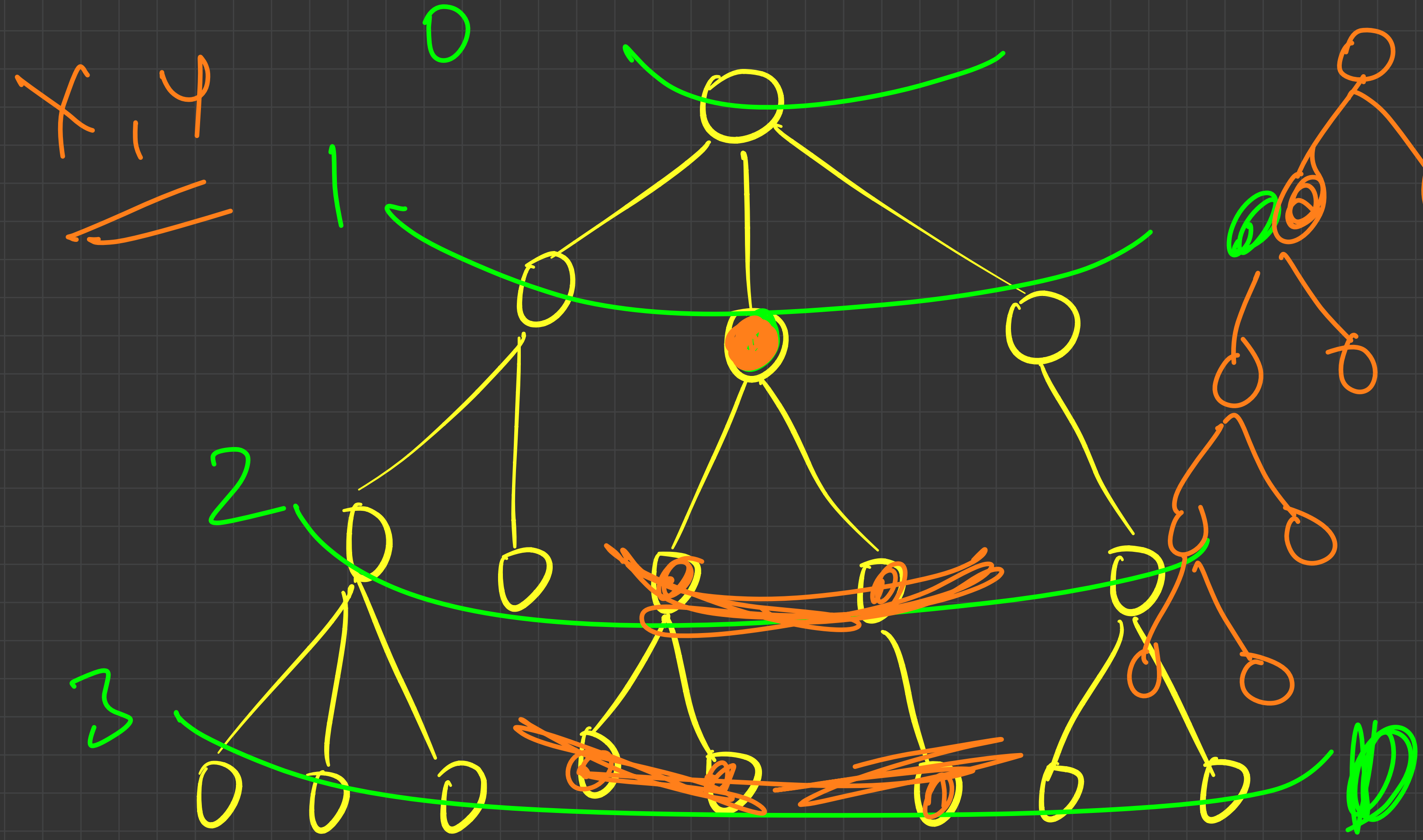


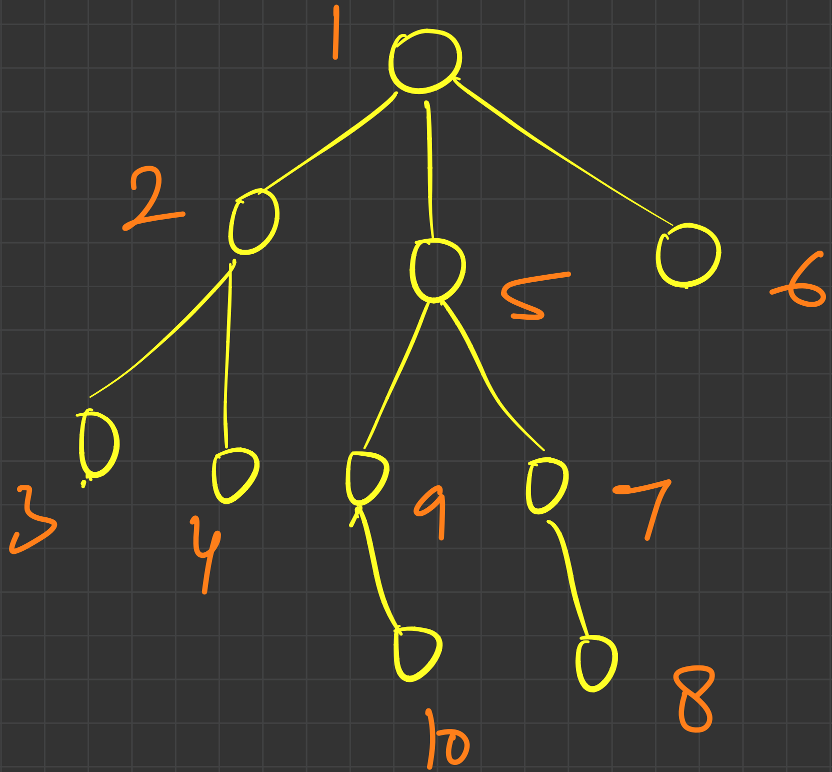
X, Y

X is an ancestor of Y

if $lca(X, Y) = X$

lowest common ancestor


X, Y
0
1
2
3



X, Y $O(N)$

$O(1)$

9 → (10)

7 → (8)

1 → (2, 3, 4, 5, 6, 7, 8, 9

5 → (9, 10, 7, 8)

Random Terms

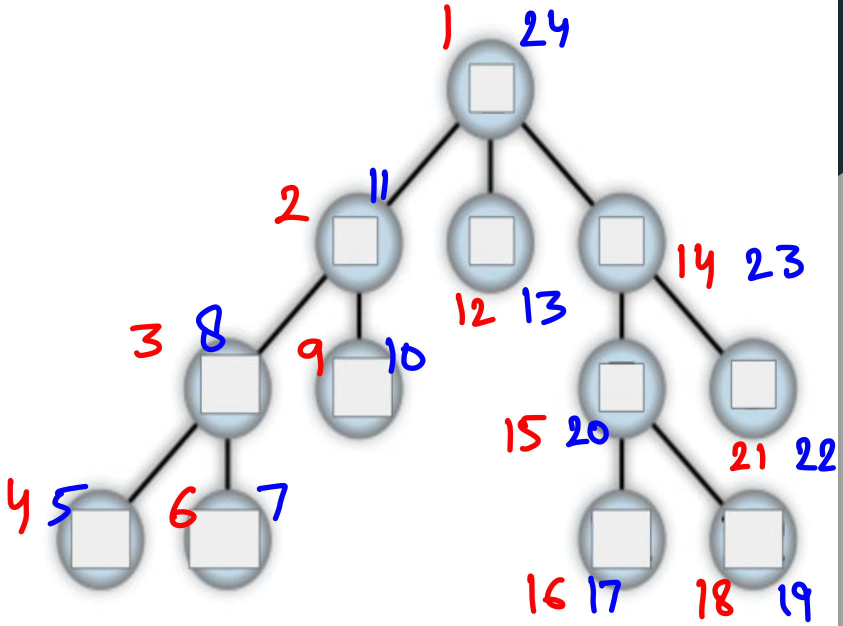
In - Out Time trick

Do a DFS traversal.

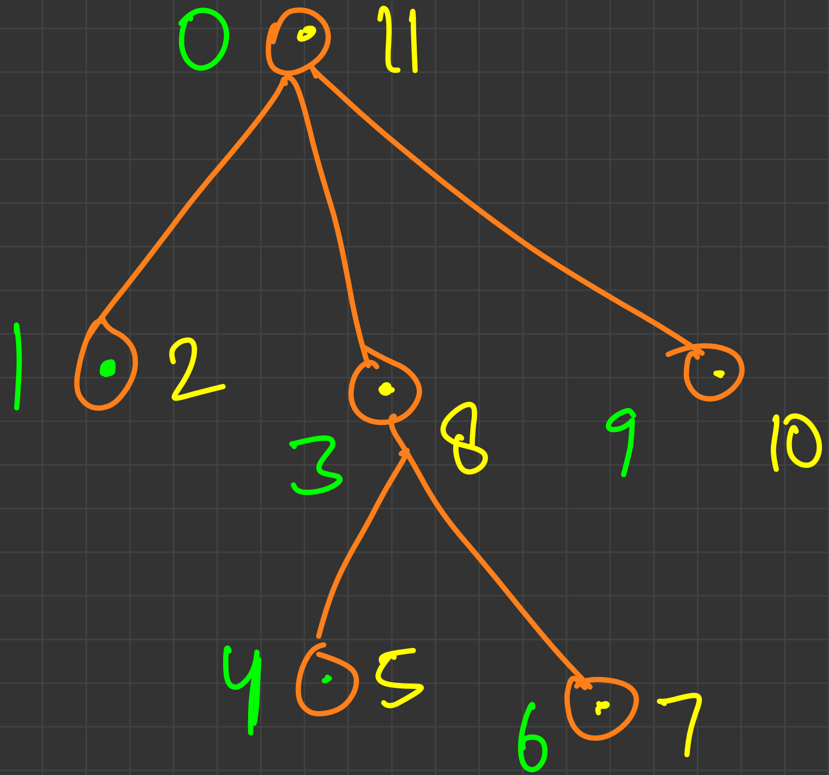
Store the following information for each node:

First visited time = In time

Last visited time = out time



Can you solve the ancestor descendant problem now?


0
11
1
2
3
8
9
10
4
5
6
7

In - Out Time trick

Solving the ancestor - descendant problem:

If X is an ancestor of Y

$$X_{\text{in time}} < Y_{\text{in time}} < Y_{\text{out time}} < X_{\text{out time}}$$

✓ ✓