★ **Height of a Tree is very imp** ★

```
int height ( node ) {
    if (node = null) return 0;
    lh = height (node→l);
    rh = haght (node→r);
    return 1+ max (lh + rh);
}
```

⇒ **Diamete of a Tree:**
↓
length of longest path
b/w any two nodes

Just add maxi = max (maxi, lh + rh) here.

⇒ **Max Path sum:**
↳ Insted of taking max (lh, rh) +1, do root→val + max (l, r)
r, l → Path len

insted of returning height, return max ~~height~~ pathsum

★ **Vertical Order Traversal** (Level order)

↳ Maintain a map < int, map < int, multiset<int> st > >.
↑ x-axis ↑ y-axis ↑ multiple values at each coordinate

↳ Maintain a queue < TreeNode*, < int, int > > for level order

⇒ **Top View:** you just need x-axis; i.e dist from root, and only stor the first node which is on x-axis at top.

↳ **Bottom view:** Exactly sam, just in the map store the last node on x-axis (i.e that ~~[illegible]~~ line)
• basically remove the condition.

↳ **Right / Left View:** The map key stores now lvl insted of line no. like previous once, rest same.