Lecture 21

Trees Code → Binary trees → Quiz

Traversals:
1. Pre | 2. Post | 3. In | 4. level

Data structure → Queue

level Order:
{50, 25, 75, 12, 68, 89, 49, 17}

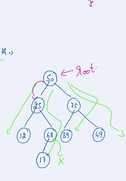
```
public void levelOrder() {
    Queue<Node> q = new LinkedList<>();
    q.addLast(this.root);
    while (!q.isEmpty()) {
        Node rv = q.removeFirst();
        System.out.print(rv.data);
        if (rv.left != null) {
            q.addLast(rv.left);
        }
        if (rv.right != null) {
            q.addLast(rv.right);
        }
    }
}
```

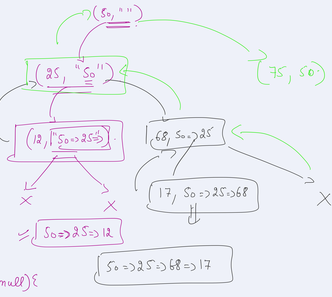
⇒ q → 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14

Output ⇒ $\boxed{50, 25, 75, 12, 68, 89, 49, 17}$

Q→ Root to leaf path →

50 ⇒ 25 ⇒ 12
50 ⇒ 25 ⇒ 68 ⇒ 17
50 ⇒ 75 ⇒ 89
50 ⇒ 75 ⇒ 49






```
public void Root to leafPath() {
    this.Root to leafPath(this.root, "");
}
private void Root to leafPath(Node node, String ogf) {
    if (node == null) {
        return;
    }
    if (node.left == null && node.right == null) {
        ogf = ogf + node.data;
        System.out(ogf);
        return;
    }
    Root to leafPath(node.left, ogf + node.data + "⇒");
    Root to leafPath(node.right, ogf + node.data + "⇒");
}
```

⇒ 50⇒25⇒12

50⇒25⇒68⇒17

Q→ Sum of all leaf Nodes

12+17+89+49 = 167

```
public int SumOfallleafNodes() {
    return SumOfallleafNodes(this.root);
}
private int SumOfallleafNodes(Node node) {
    if (node == null) {
        return 0;
    }
    if (node.left == null && node.right == null) {
        return node.data;
    }
    int lsum = SumOfallleafNodes(node.left);
    int rsum = ——————— (node.right);
    int mysum = lsum + rsum;
    return mysum;
}
```

Q→ Diameter of Tree

ans = 5

```
public int diameter() {
    return this.diameter(this.root);
}
private int diameter(Node node) {
    if (node == null) {
        return 0;
    }
    int lheight = this.height(node.left);
    int rheight = ——————— (node.right);
    int f1 = lheight + rheight + 3;
    int f2 = this.diameter(node.left);
    int f3 = ——————— (node.right);
    int myans = Math.max(f1, Math.max(f2, f3));
    return myans;
}
```

$O(n^3)$

$O(n)$

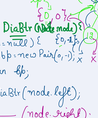
① → $\boxed{f_1 = lheight + rheight + 3;}$

② → $\boxed{f_2 = lhs\ subtree.}$

③ → $\boxed{f_3 = rhs\ subtree.}$

$\boxed{Max(f_1, Max(f_2, f_3));}$

Q→ $\boxed{\text{Diameter (↳ O(n))}}$ Using {3, 2}



```
public Pair DiaBtr(Node node) {
    if (node == null) {
        Pair p = new Pair(0, -1);
        return p;
    }
    Pair lp = DiaBtr(node.left);
    Pair rp = ——————— (node.right);
    Pair mp = new Pair();
    mp.height = Math.max(lp.height, rp.height) + 1;
    mp.diameter = Math.max(lp.height + rp.height + 2, Max(lp.diameter, rp.diameter));
    return mp;
}
```

$\boxed{0 + -1 + 2} \Rightarrow 1$

$\boxed{0, 0, 1}$

$\boxed{0 + 1 + 2}$ $\boxed{3, 0, 1}$

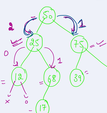
O(n)

Resume at 8:50pm

Class Pair {
int diameter;
int height;
Pair() {}
Pair(int d, int h) {
diameter = d
height = h;
}
}

Q→ isBalanced

true



$\boxed{2 - 1} = \boxed{1 <= 1}$

diff b/w height of lftbtree and rhtbtree of any node <= 1

Class Pair {
int height;
boolean isbalanced;
}

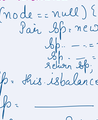
```
public boolean isbalanced() {
    return this.isbalanced(this.root).isbalanced;
}
private Pair isbalanced(Node node) {
    if (node == null) {
        Pair lp = new Pair();
        lp.height = -1;
        lp.isbalanced = true;
        return lp;
    }
    Pair lp = this.isbalanced(node.left);
    Pair rp = ——————— (node.right);
    Pair mp = new Pair();
    mp.height = Math.max(lp.height, rp.height) + 1;
    if (Math.abs(lp.height - rp.height) <= 1 && lp.isbalanced
            && rp.isbalanced) {
        mp.isbalanced = true;
    } else {
        mp.isbalanced = false;
    }
    return mp;
}
```

O(n)

⇒ {50, 25, 12, 17}

Q→ Left View of the tree

{50, 75, 49, 17}



Sol 1
[50]
[25 75]
[12, 68, 39] = 49
[17]

50, 75, 49, 17

```
public void rightView(Node node, int level) {
    if (node == null) {
        return;
    }
    if (level > max_level) {
        System.out(node.data);
        max_level = level;
    }
    rightView(node.right, level+1);
    rightView(node.left, level+1);
}
```

leftView

Q→ ↳ Question of trees

BST ⇒ Wednesday