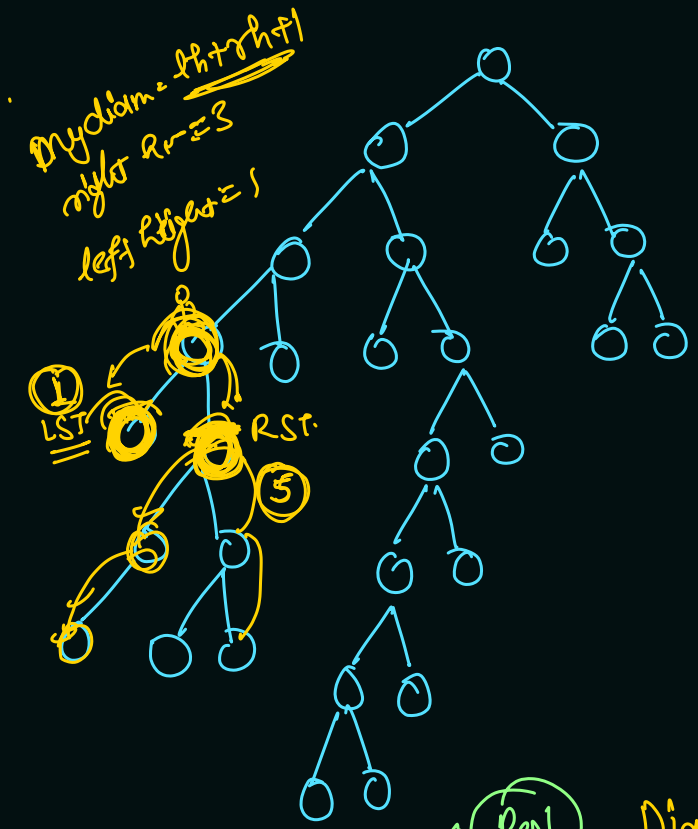
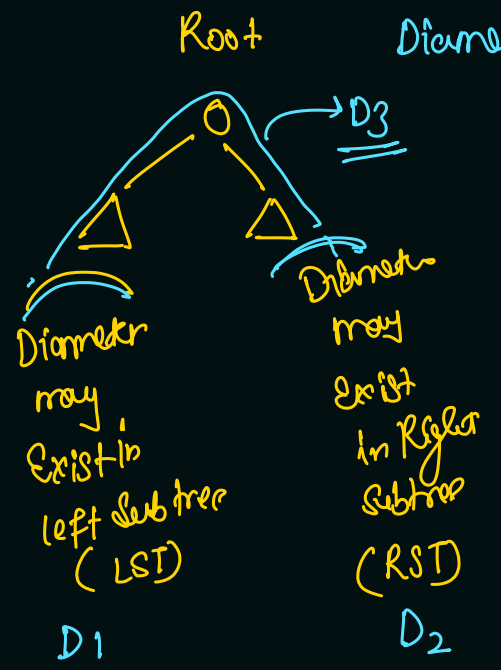
Diameter of Binary Tree :

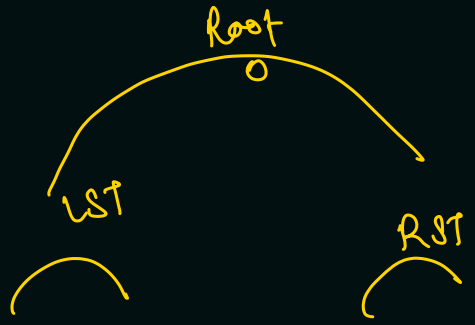
Diameter → Distance between two farthest node is known as diameter of a tree.

↳ Structure based problem





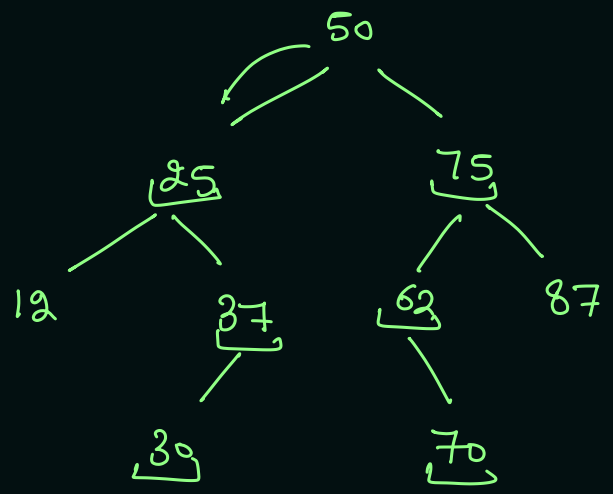
Diameter may pass through Root

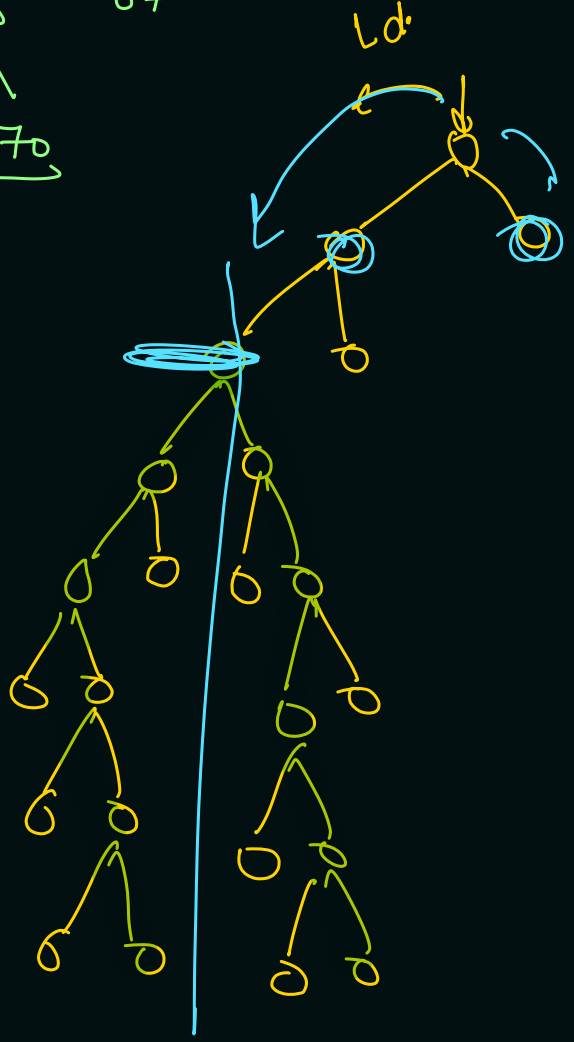


Diameter Result =

- left Diameter
  - vs
- Right Diameter
  - vs
- Diameter through Root = lh + rh + 1

left height + Right height






```java
public static int diameter1(Node node) {
    if(node == null) return 0;

    int ld = diameter1(node.left);
    int rd = diameter1(node.right);

    int lh = height(node.left);
    int rh = height(node.right);
    int md = lh + rh + 1;

    return Math.max(md, Math.max(ld, rd));
}
```

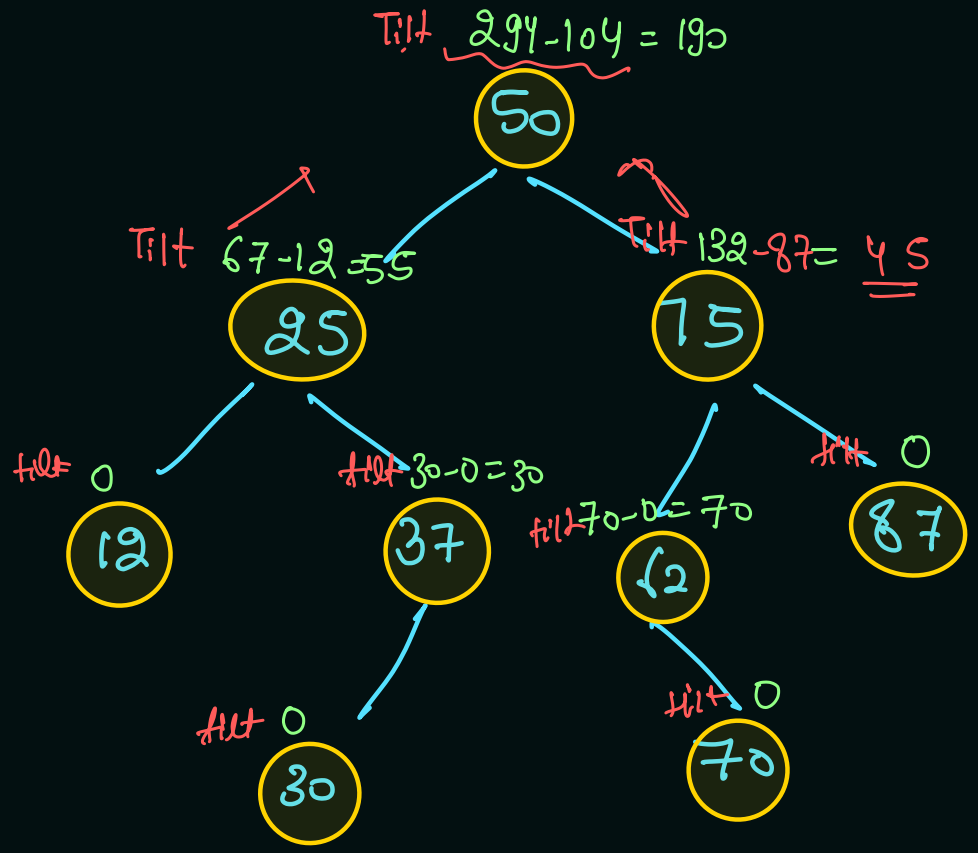
ld Diameter + height

rd Diameter + height

→ O(n) for Every Single Iteration

Time Complexity → $O(n^2)$

Tilt of a binary Tree :

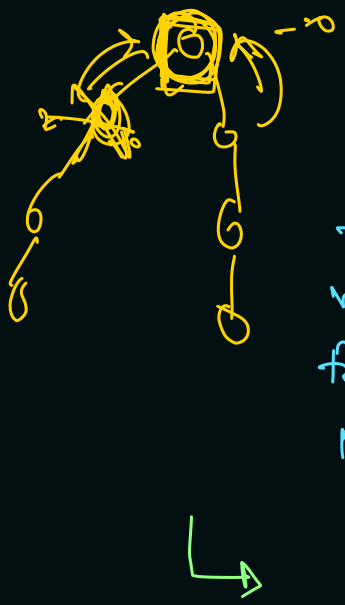


tilt of tree = 190 + 55 + 45 + 0 + 30 + 70 + 0

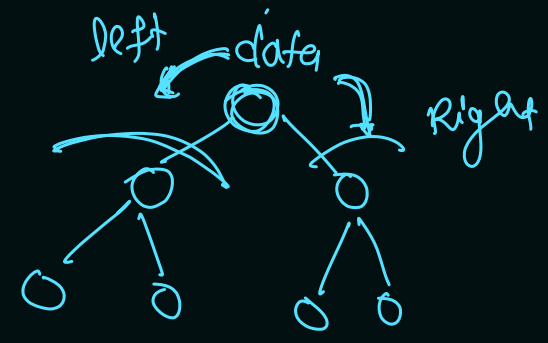
= 390

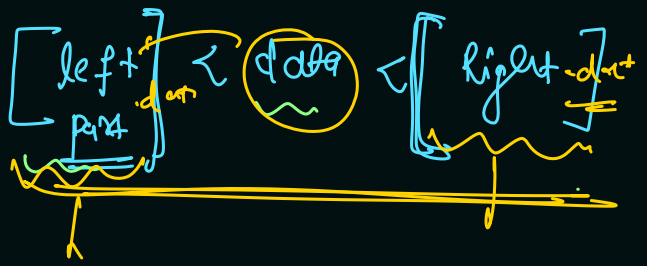
Sum of tree

Is a Binary Search Tree;



this is valid for all Node





max < data < min

Binary Search

→ Data sorted

→ mid, compare.

data < mid
↳ left

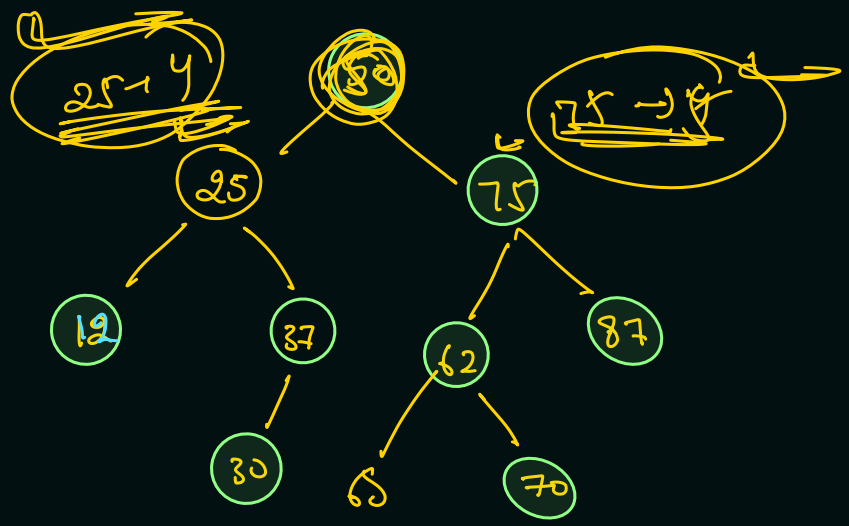
data > mid
↳ Right

In Single Traversal

Multisolver

- max → ht
- min → isBST
- size → diameter
- sum

→ MultiSolver




```java
public static boolean isBST(Node node) {
    if(node == null) return true;

    boolean lres = isBST(node.left);
    boolean rres = isBST(node.right);

    boolean res = true;
    if(node.left != null && node.right != null)
    res = (node.left.data < node.data && node.data < node.right.data);

    return lres && rres && res;
}
```

false

10:00 BST