Agenda

- Diameter of a BT
- is Balanced BT
- level order traversal
- left view
- right view
- zig zag traversal

Diameter of a Binary Tree.

{Max^{m} distance b/w any two leaf Nodes}

{40, 80} → 4

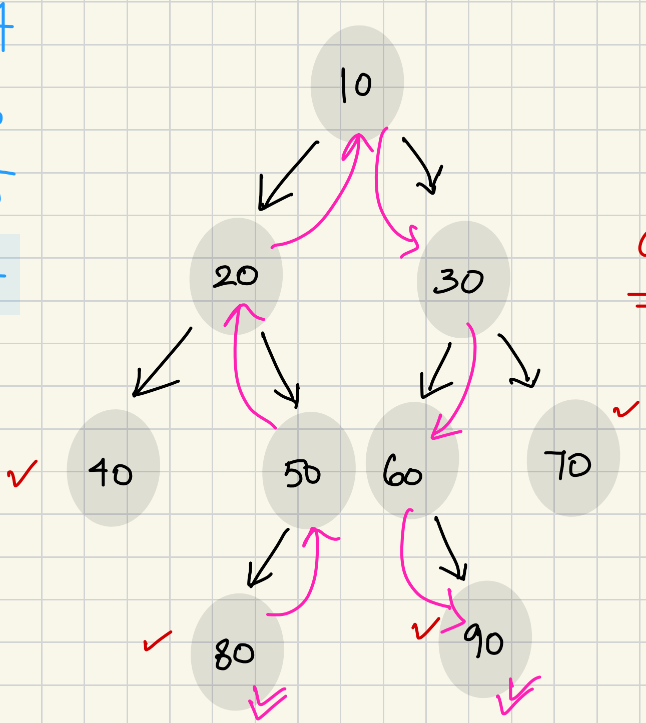
{40, 90} → 6

{40, 70} → 5

{80, 90} → 7

{80, 70} → 6

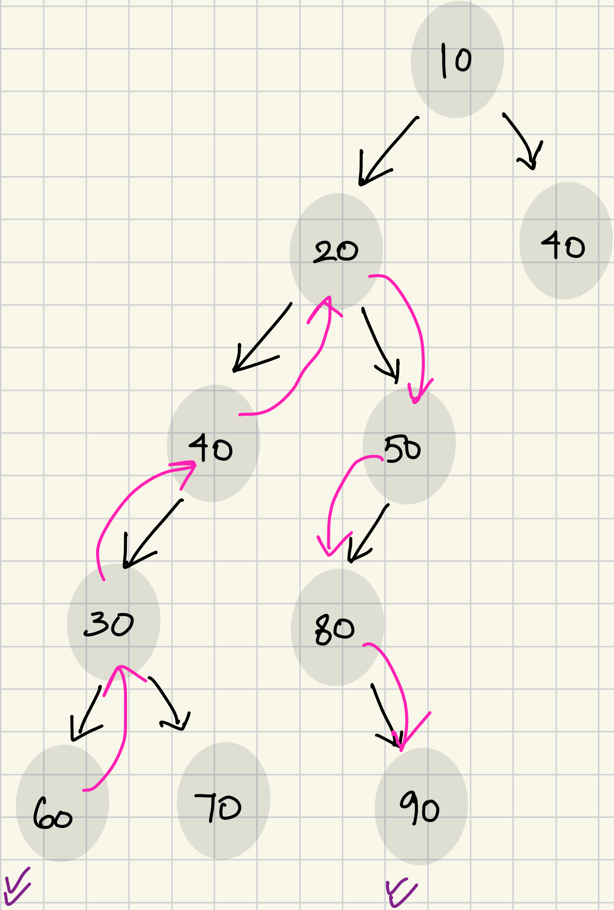
{90, 70} → 4



diameter = 7

~~diameter = hLST + hRST + 1~~

Wrong



~~diameter =~~ 4 + 1 + 1 = 6

NOTE: diameter not passing through root }

diameter = max { diameter through each node } faith: returns diameter of the tree

```
int diameter(Node root)
{

    int diaLST = diameter(root.left)
    int diaRST = diameter(root.right)
    int diaRoot = h(LST) + h(RST) + 1;

  return max{diaLST, diaRST, diaRoot};
}
```

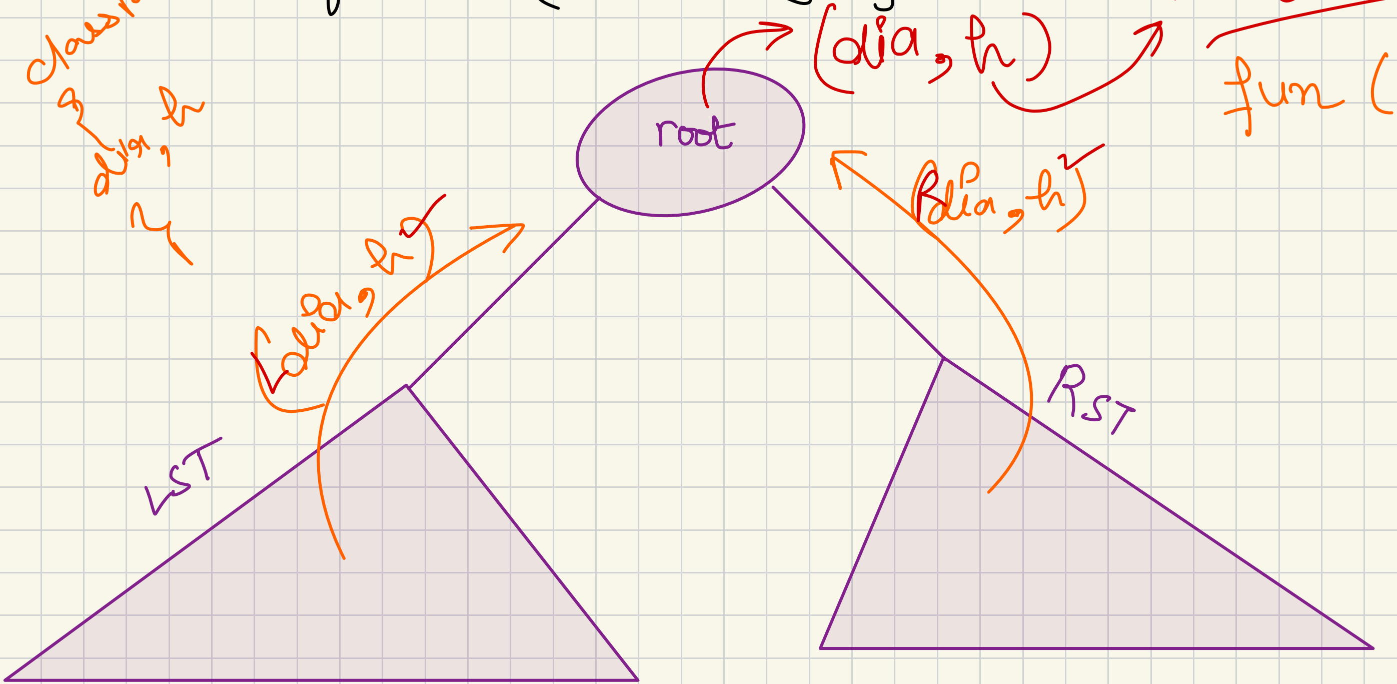
TC: $O(N^2)$

faith : {dia, height} returns

max(LST_h, RST_h)+1

fun (root)

class faith
{
dia,
h
}



is Balanced Binary Tree.

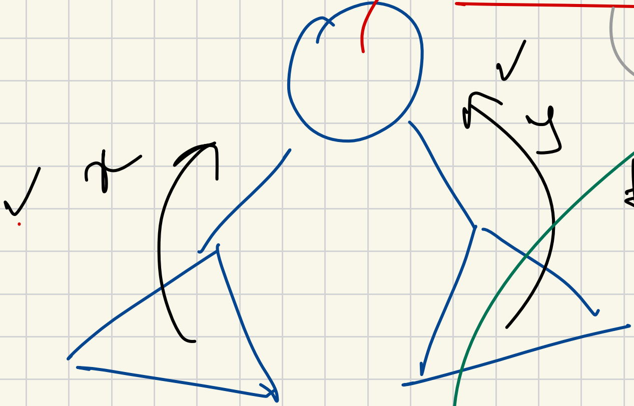
- Each Node is Balanced

is Node Balanced

Balanced Node

$$|h_{LST} - h_{RST}| \le 1$$

Math.abs(h(LST) - h(RST)) <= 1.



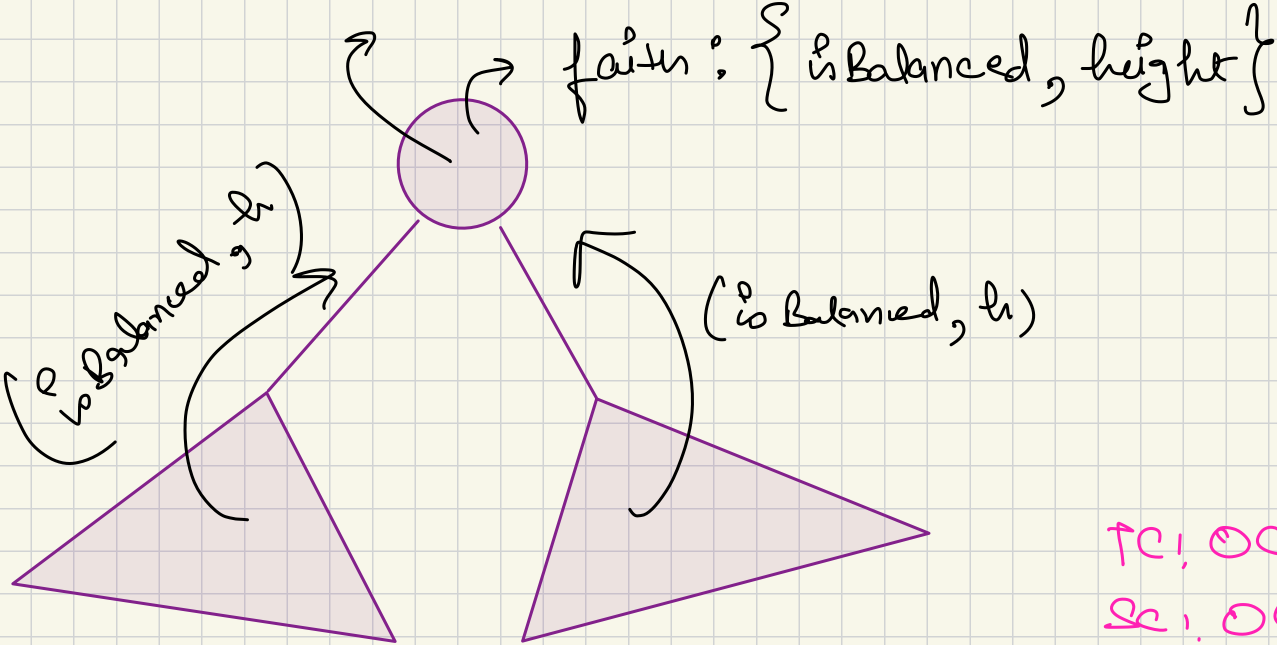
faith: returns is BT Balanced?

```
boolean isBalanced(Node root)
{
    boolean x = isBalanced(root.left)   ✓
    boolean y = isBalanced(root.right)  ✓

    return ______________ true ? false
}
```

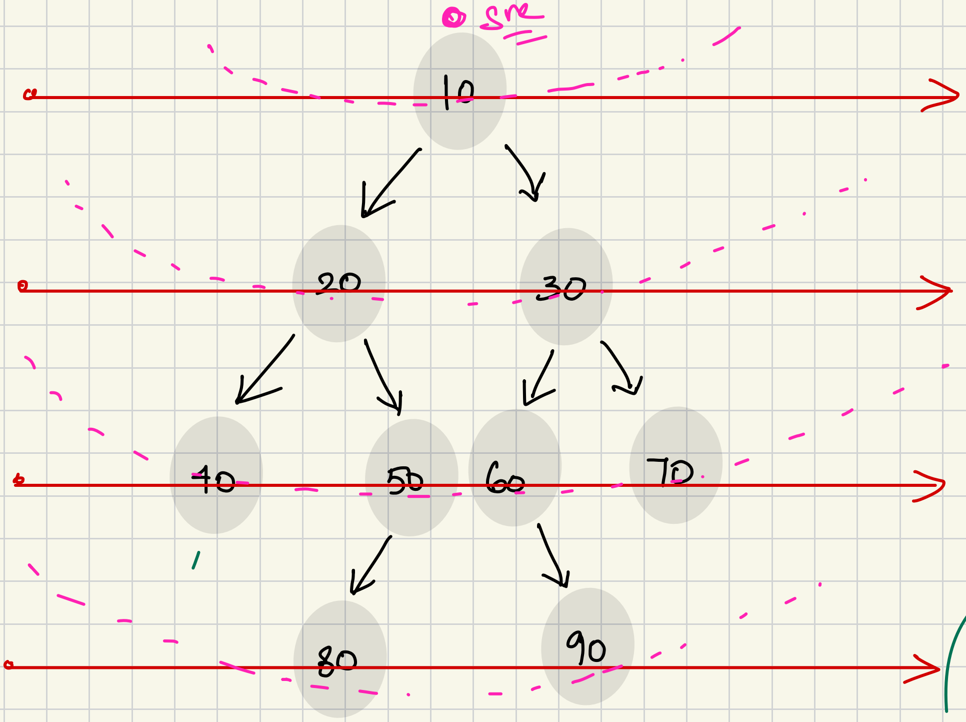
TC: $O(N^2)$

SC: $O(H)$



TC: O(N)
SC: O(H)

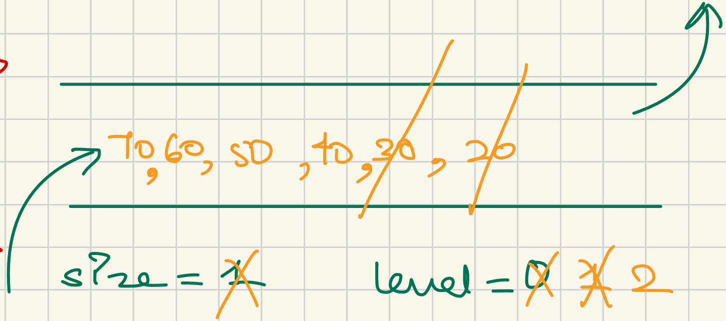
Level Order traversal . (BFS)



o/p

TC: O(N) SC: O(2^h)

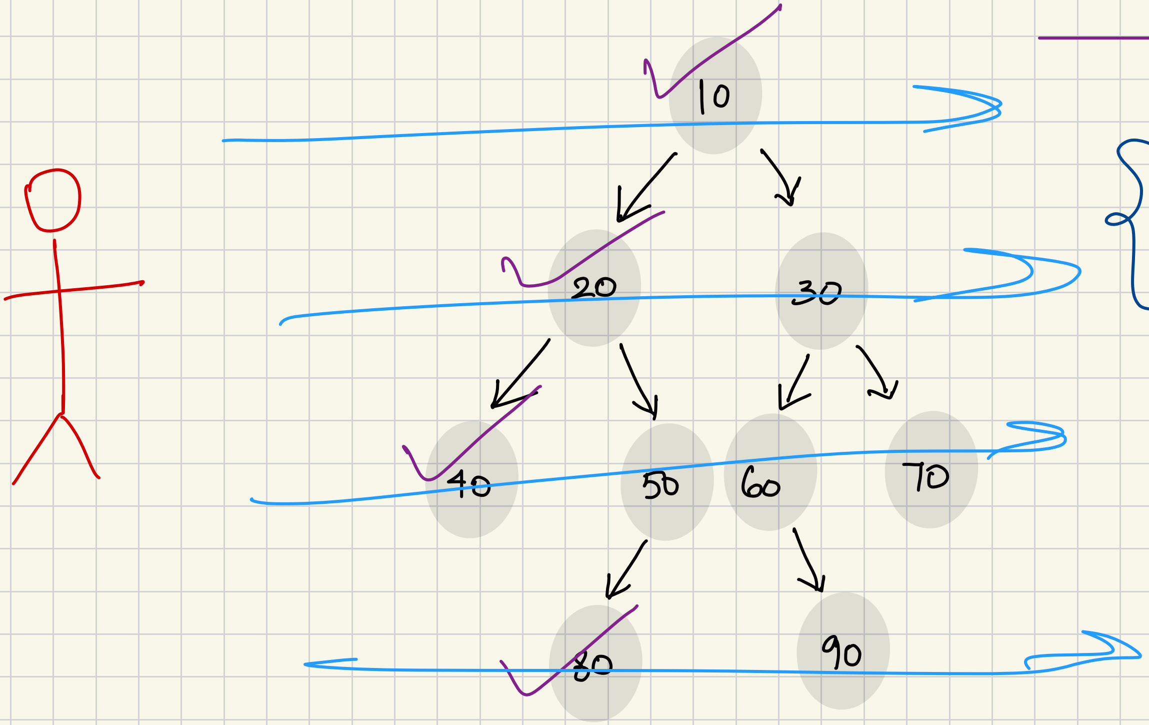
- Level 0 : 10 ✓
- level 1 : 20 ✓ 30 ✓
- level 2 . 40 50 60 70
- level 3 80 90



size = ~~1~~ ~~0~~ ~~2~~ ~~1~~ ~~0~~ 4

level = ~~0~~ ~~1~~ 2

Left View in a binary tree.

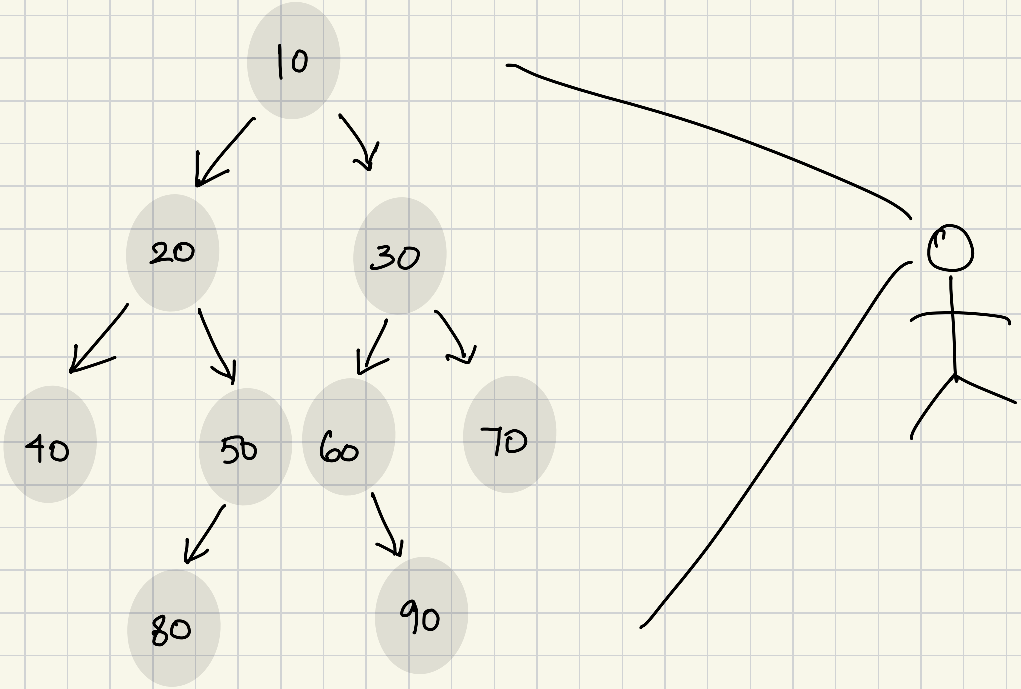


Left View : 10,20,40,80

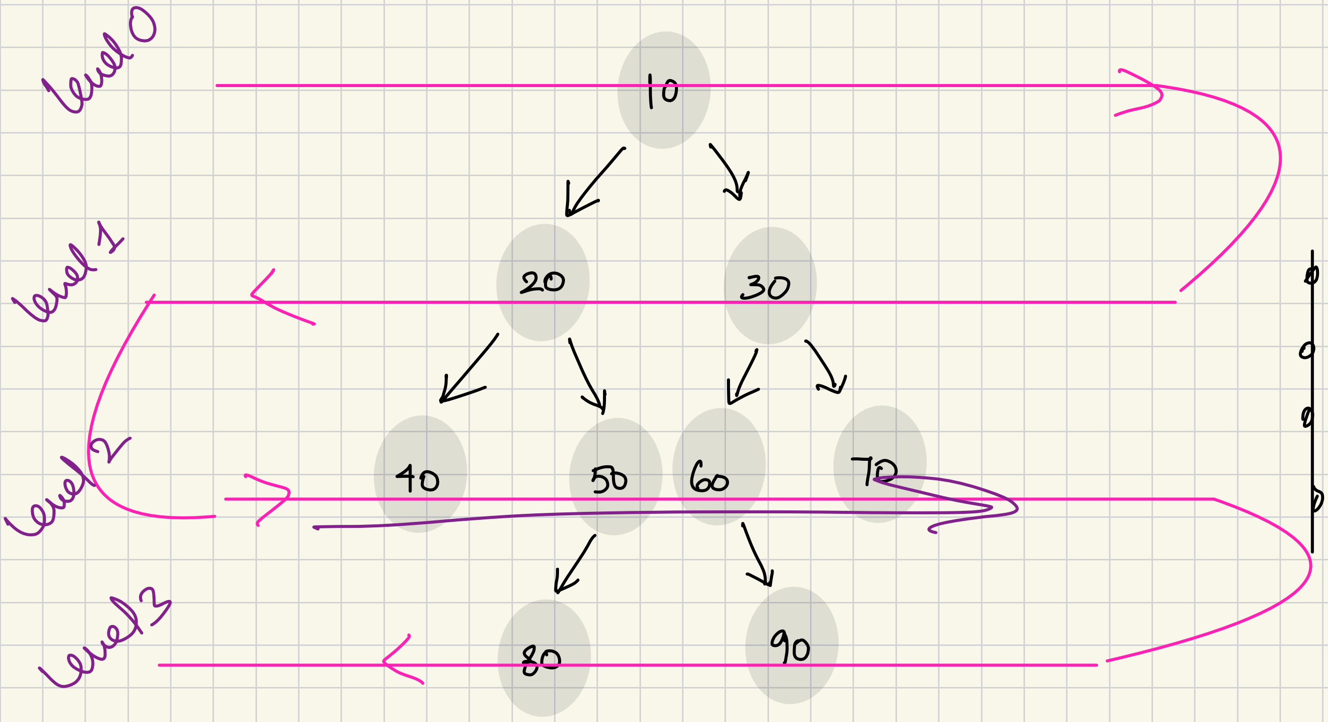
{first node of level}

Right View

{10, 30, 70, 90}



Zig zag traversal.



10

30 20

40 50 60 70

90 80