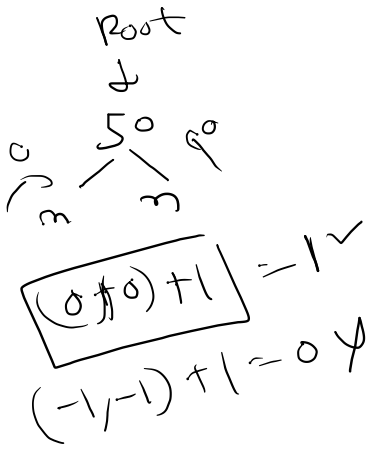
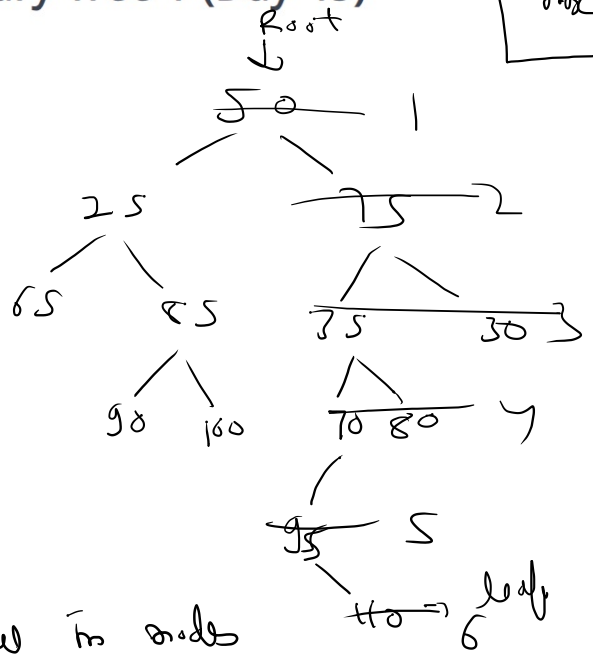
Maximum Depth of Binary Tree 1 (Day 43)





max (left, Right) + 1

base case => root == null

return 0

=> height of tree in node

ans = 6

Sum of Root To Leaf Binary Numbers (Day 43)

1 0 0 = 4

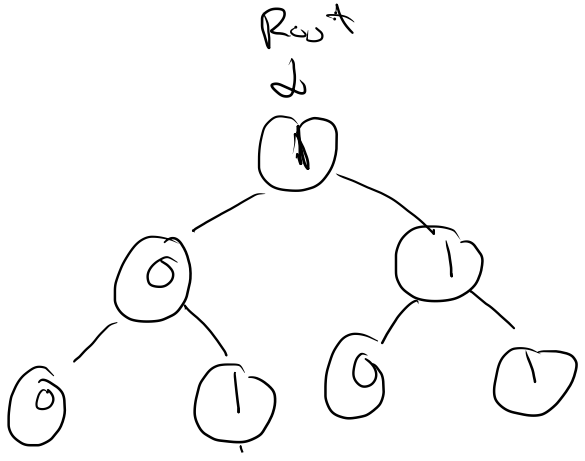
1 0 1 = 5

1 1 1 = 7

1 1 0 = 6

\+ 22

1 0 0 n n 1 n n 1 0 n n 1 n n

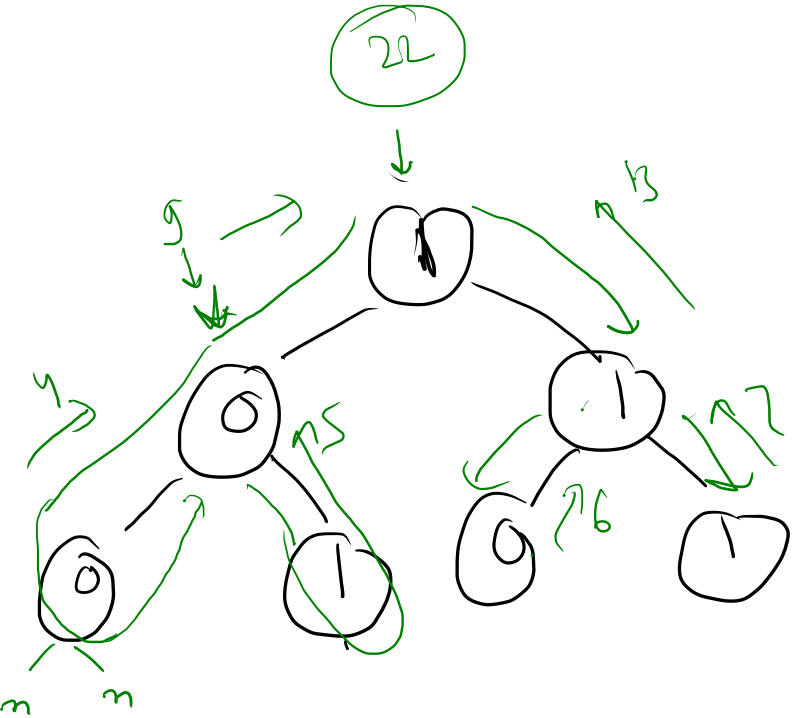


4 leaf's - Root to leaf Path

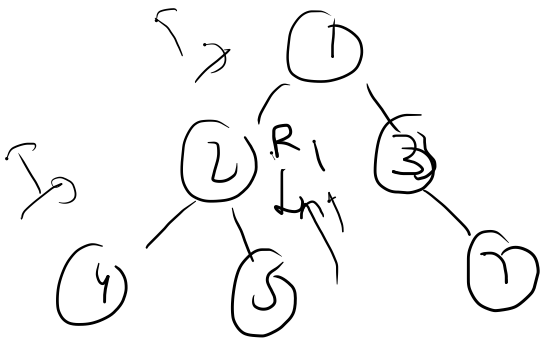
Store top = 1 1 1

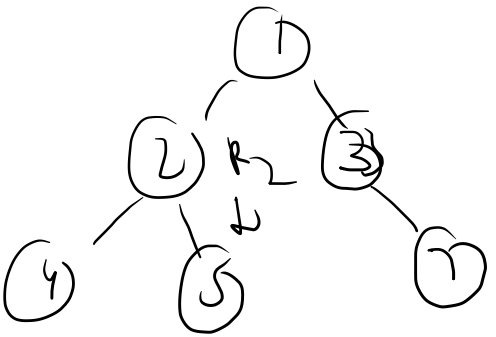
↓ Convert it to decimal

4 + 5 + 6 + 7

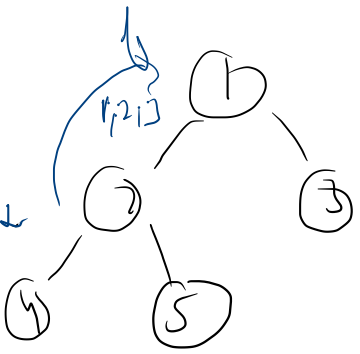


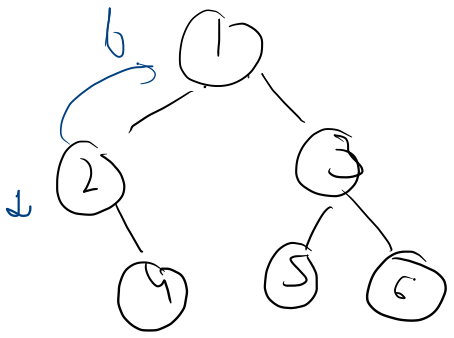
. Same Tree (Day 43)



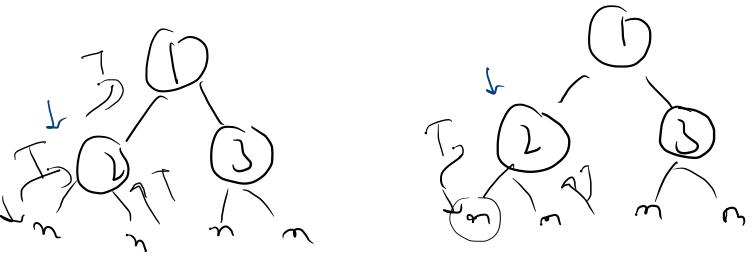


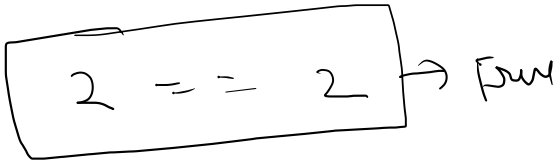
Structure wise diff





```java
if (node1 == null && node2 == null) {
    return true;
}

// Structure is not same
if (node1 == null || node2 == null) {
    return false;
}

int isLeftSubtreeSame = isSameTrees(node1.left, node2.left);
if (! isLeftSubtreeSame) {
    return false;
}

int isRightSubtreeSame = isSameTrees(node1.right, node2.right);
if (! isRightSubtreeSame) {
    return false;
}
return (node1.val == node2.val);
}
```

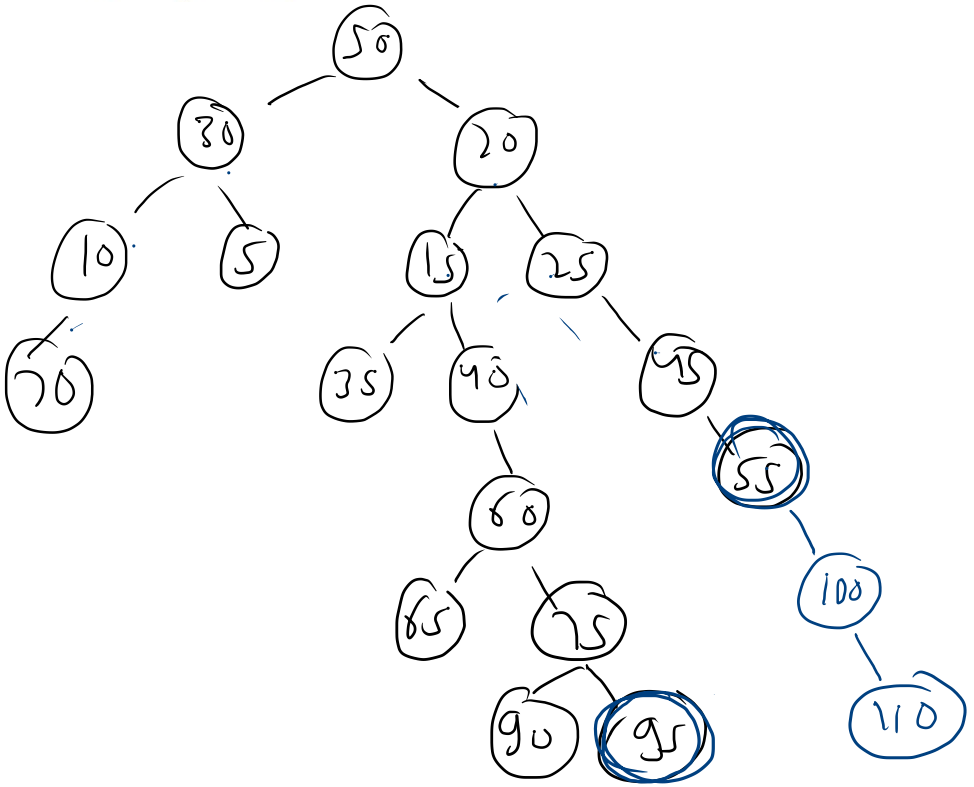


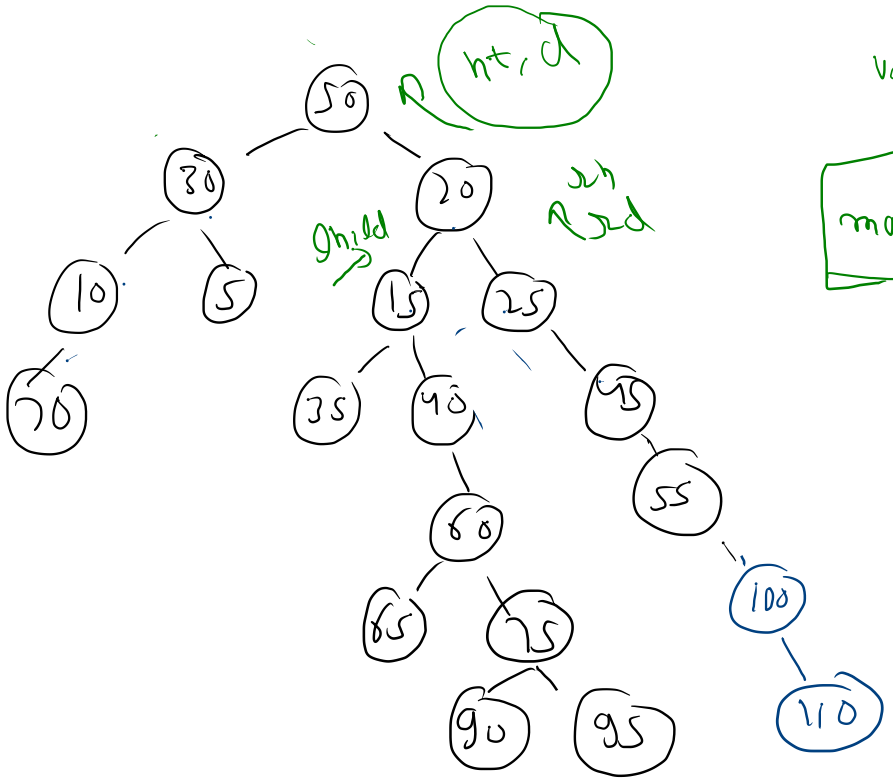


2 == 3 false

```java
if (node1 == null && node2 == null) {
    return true;
}

// Structure is not same
if (node1 == null || node2 == null) {
    return false;
}

int isLeftSubtreeSame = isSameTrees(node1.left, node2.left);
if (! isLeftSubtreeSame) {
    return false;
}

int isRightSubtreeSame = isSameTrees(node1.right, node2.right);
if (! isRightSubtreeSame) {
    return false;
}
return (node1.val == node2.val);
}
```

Diameter of Binary Tree (Day 44)





ht, d

val = (lh + rh + 2)

max(val, ld, rd) ans

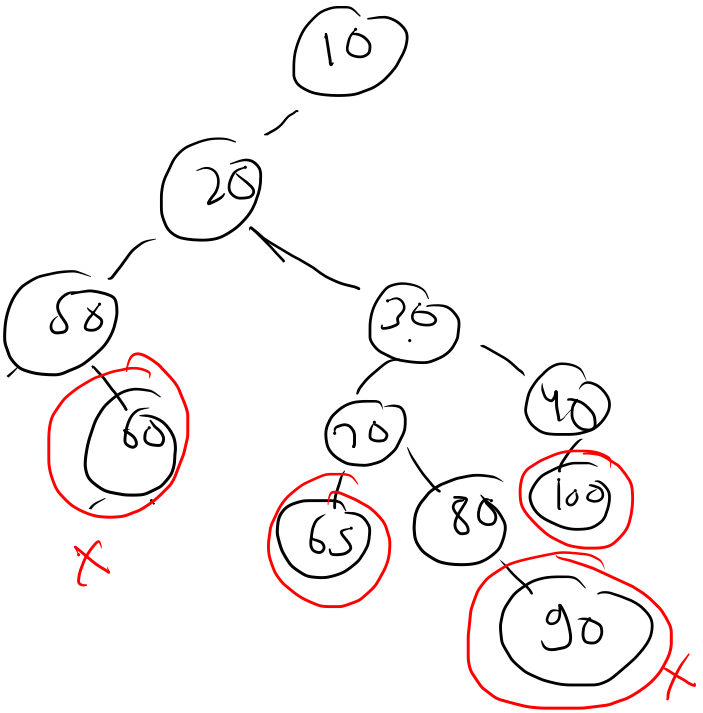
↳ return

Pair

→ height
dia

Sum of Left Leaves 1 (Day 44)

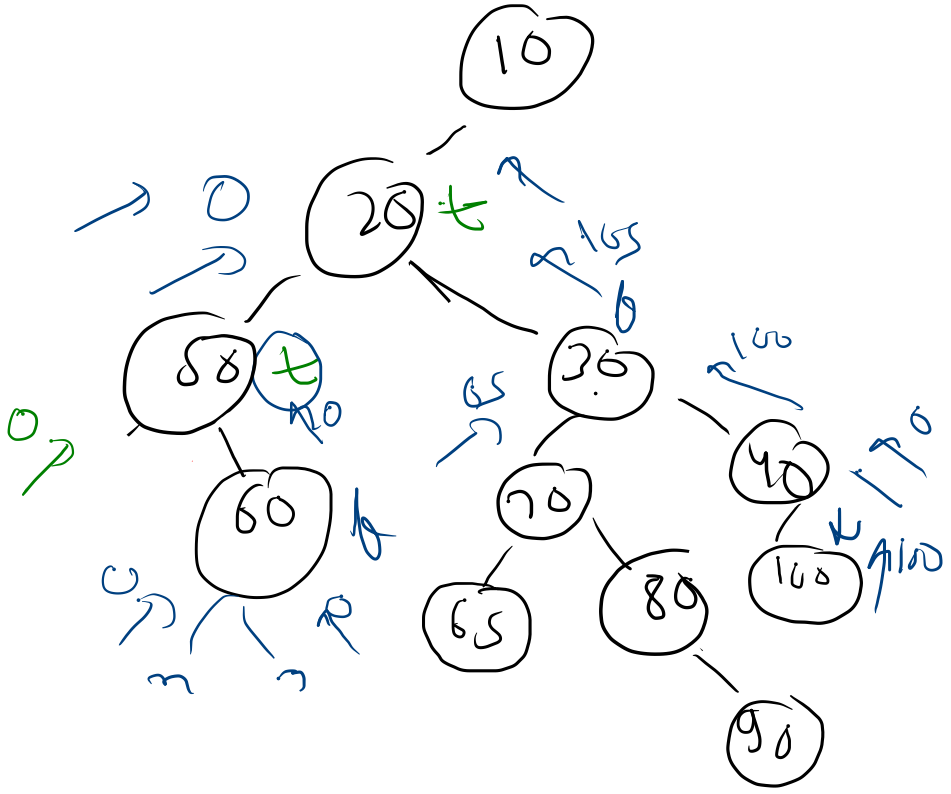
```
10 20 50 n 60 n n 30 70 65 n n 80 n 90 n n 40 100 n n n n
```



100 65

100+65

= 165


10
20
30
40
50
60
70
80
90
100
65