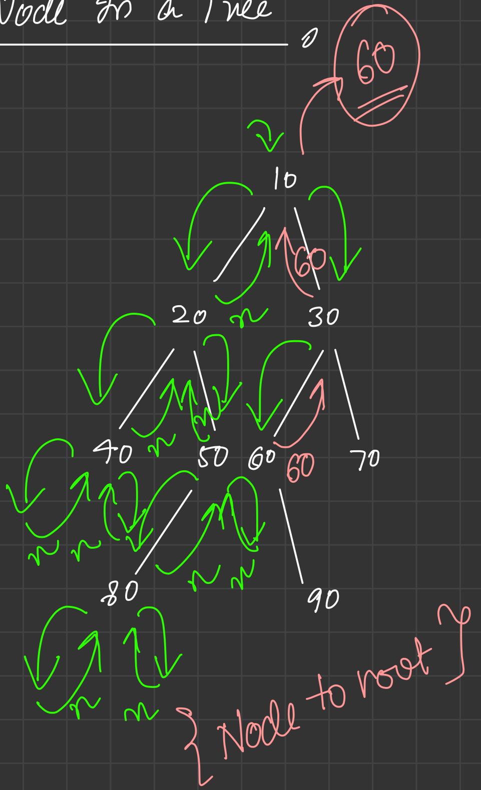
find Node in a Tree

tar = 60



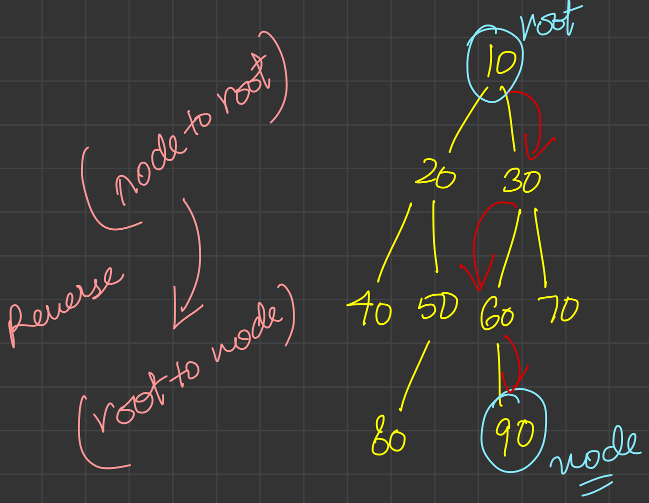
faith: returns address of Node if found in tree.

```
Node find(Node node, int tar)
{
    if (node = = null) return null;
    if (node.data = = tar) return node;

    Node filc = find (node.left, tar);
    if (filc != null) return filc

    Node firc = find (node right, tar);
    if (firc != null) return firc;

    return null;
}
```

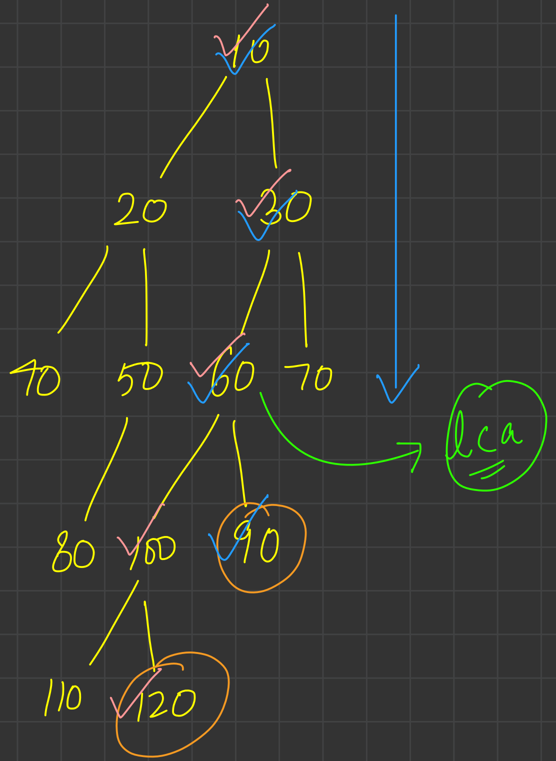
Path to Given Node

{Root to Node Path}

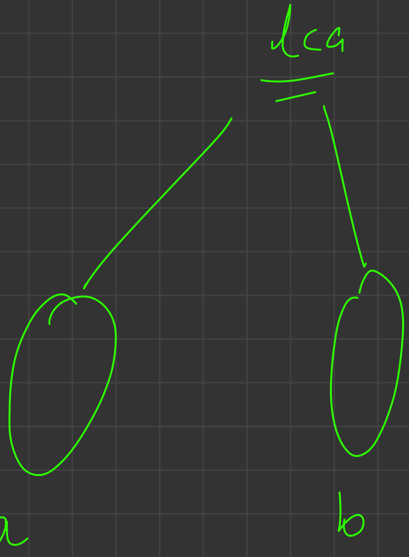
10 → 30 → 60 → 90



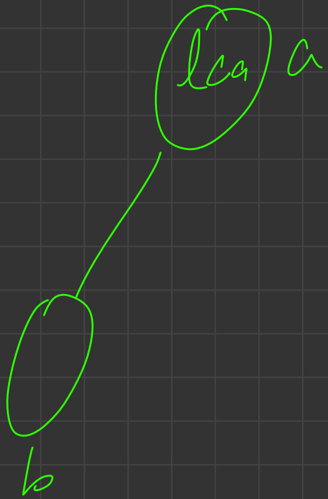
Lowest Common Ancestor. (lca)



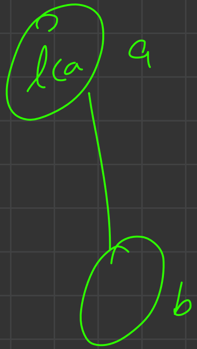
Case 1

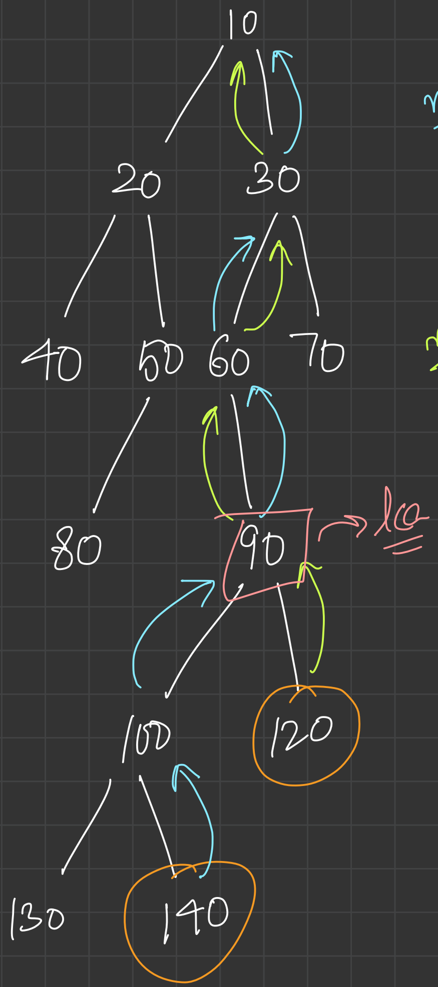


Case 2



Case 3





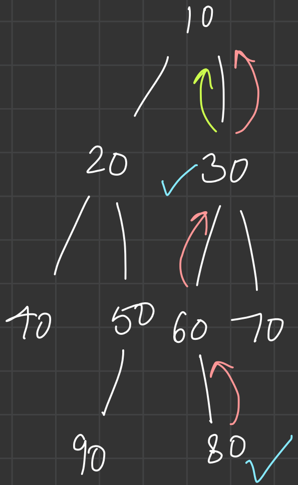
n2r_o

{140, 100, 90, ~~60~~, ~~30~~, ~~10~~}

n2r_o

{120, 90, ~~60~~, ~~30~~, ~~10~~}

lca = ~~10~~ ~~30~~ ~~60~~ 90 ✓



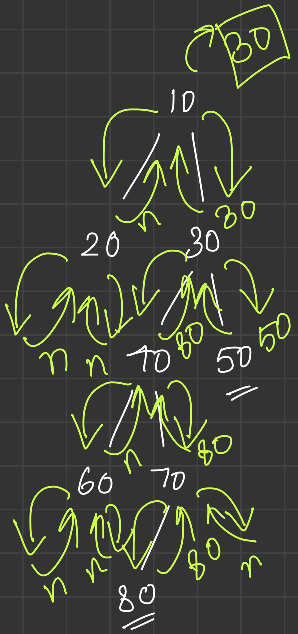
n2R

{80, 60, 30, 10}

n2R

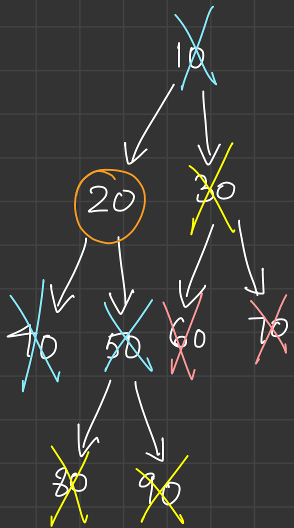
{30, 10}

lca: ~~10~~ 30



Node lca(Node root, int n1, int n2)

Burning of trees



t = ~~0~~ ~~1~~ ~~2~~ 3

✓ time = 3 unit

HashMap

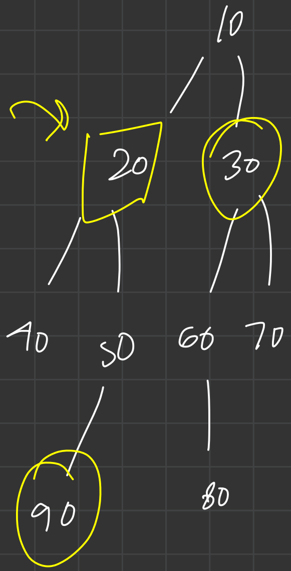
{

key ↓ child

value ↓ parent

K distance away

k = 2


10
20
30
40
50
60
70
90
80