CS & IT ENGINEERING

Data Structures

Tree

Lecture No.- 04

By- Pankaj Sharma Sir

Recap of Previous Lecture

Topic Tree Part-03

unlabelled binary trees

labelled binary tree

Pre →

Post →

In →

Pre, In

Post, In →

Topics to be Covered

Topic Tree Part-04

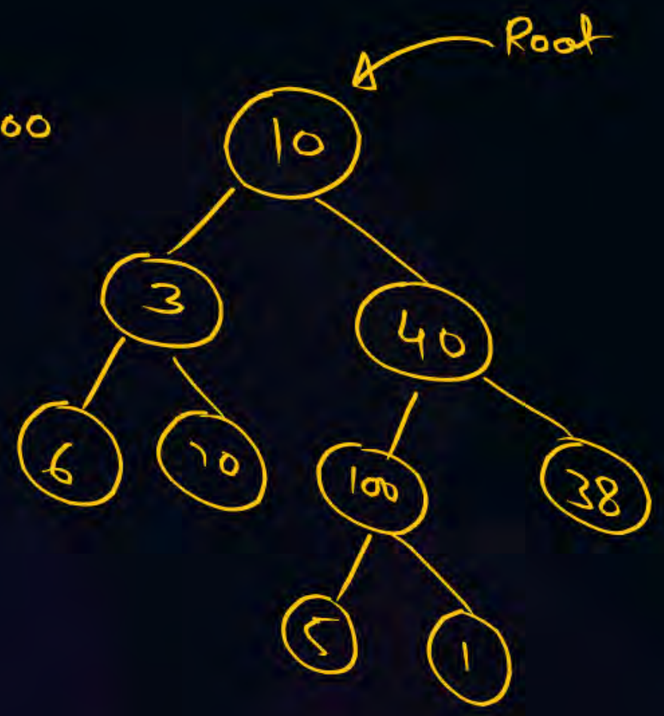
BST

Topic : Tree

BST

① why?

key = 100



Given a key & a binary tree, find whether the key is present in the tree or not?

↓

Traverse → O(n) time

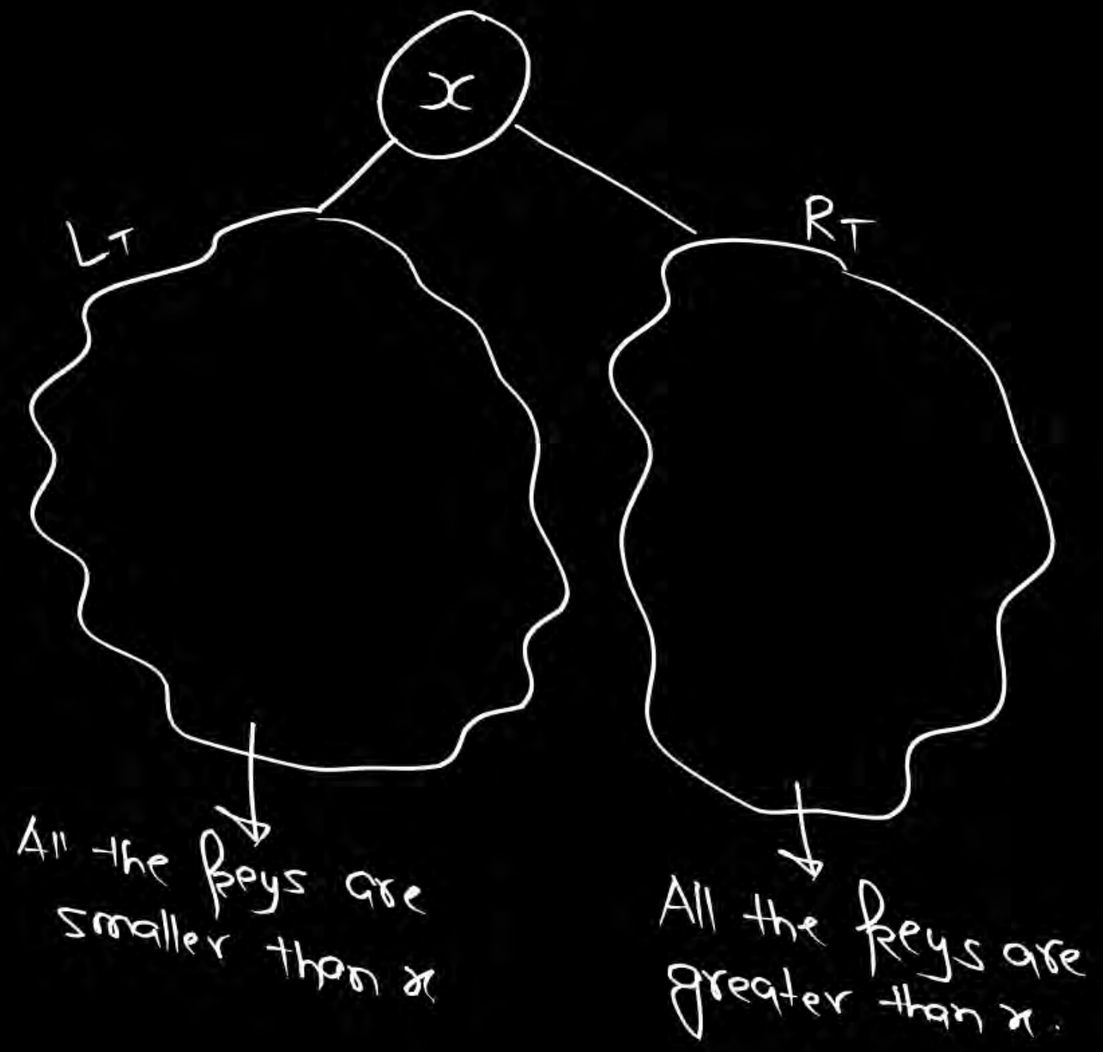
BST

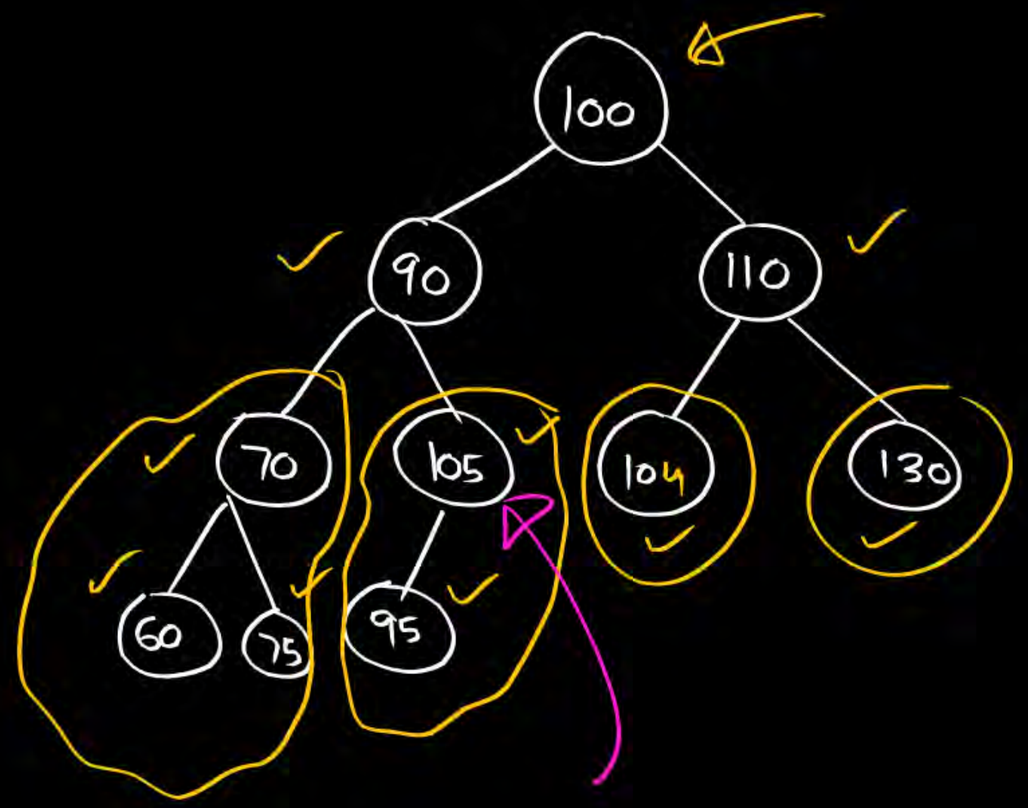
A binary search tree is a binary tree in which every node satisfies the following property:

All the keys in the left subtree of a node are smaller than the key in the node.

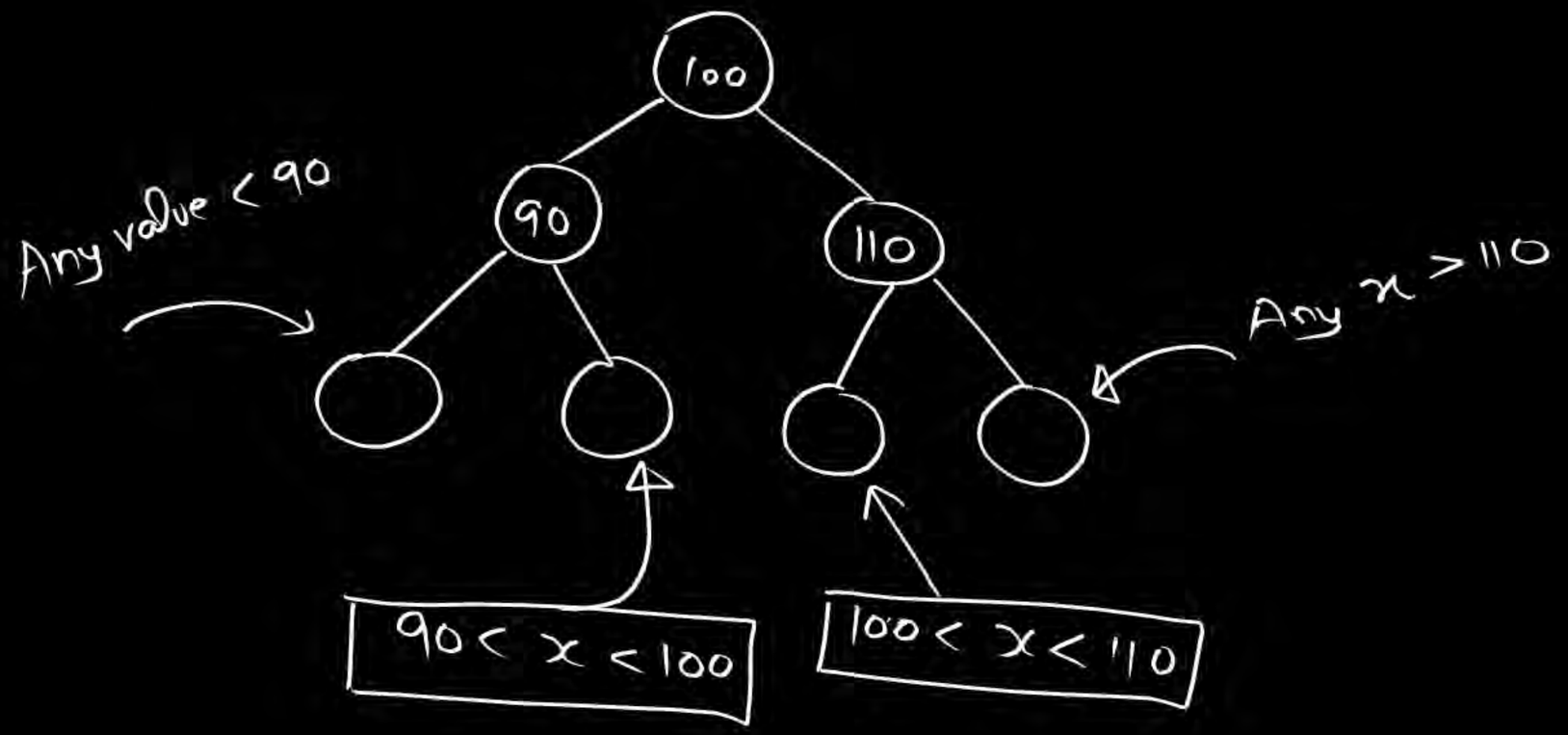
&

All the keys in the right subtree of a node are greater than the key in the node.





Binary tree
BST

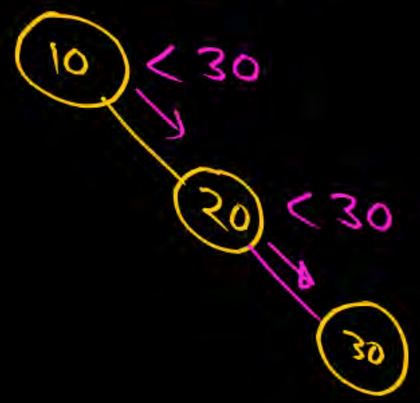

100
90
110
Any value < 90
Any x > 110
90 < x < 100
100 < x < 110

Q Construct a BST by inserting keys $\overrightarrow{10, 20, 30}$ in order into initially empty BST?

(i) Insert 10

(ii) Insert 20

(iii) Insert 30

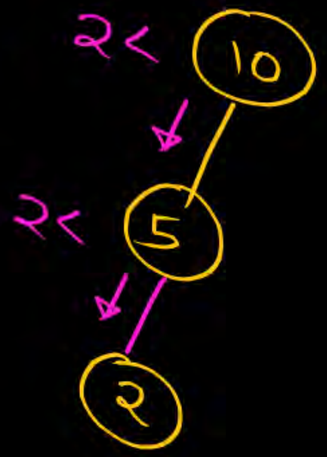


Q Construct a BST by inserting keys 10, 5, 2, 9, 3, 4 in order.

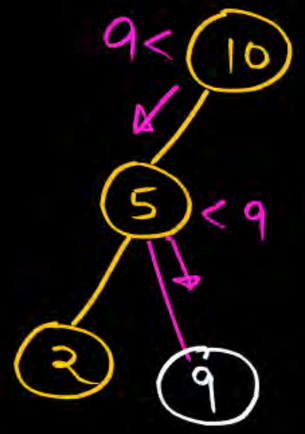
(i) Insert 10

(ii) Insert 5

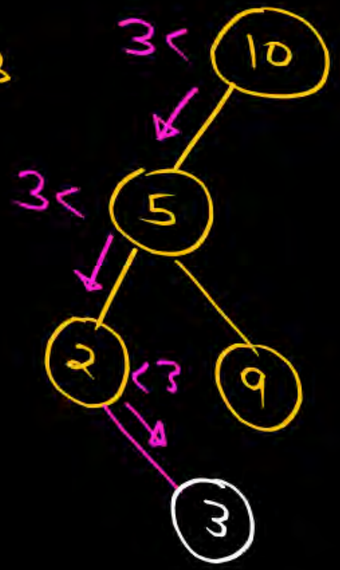
(iii) Insert 2



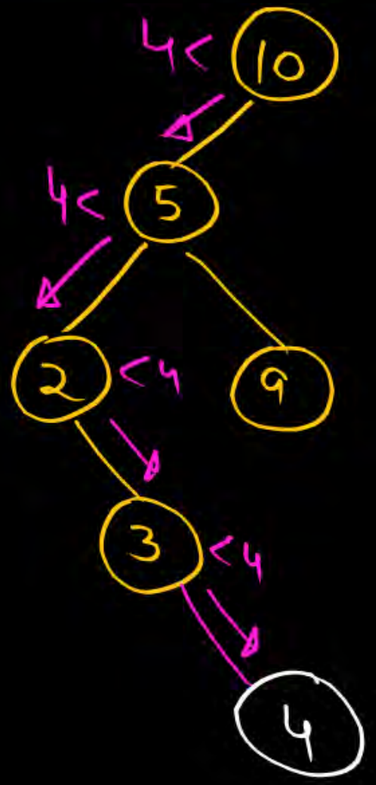
(iv) Insert 9



(v) Insert 3



vi) Insert 4



No. of BST, when the insertion order of keys is fixed $\longrightarrow$ 1

a) Const. BST by inserting keys 10, 20, 30 $\longrightarrow$ 1

b) Const. BST by inserting keys 10, 5, 2, 9, 3, 4 $\longrightarrow$ 1

Const. BST by inserting keys 1,2,3? (in any order) → 5 BST

$\boxed{n=3}$ keys → 5 BSTs

order could be

a) 1, 2, 3

b) 1, 3, 2

c) 2, 1, 3

d) 2, 3, 1

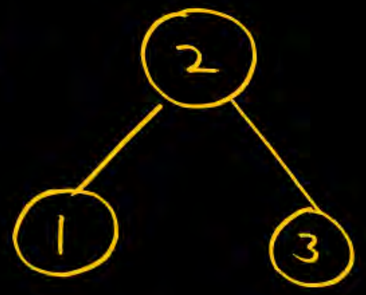
e) 3, 1, 2

f) 3, 2, 1

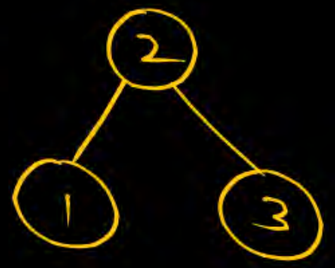
a)

b)

c)



d)



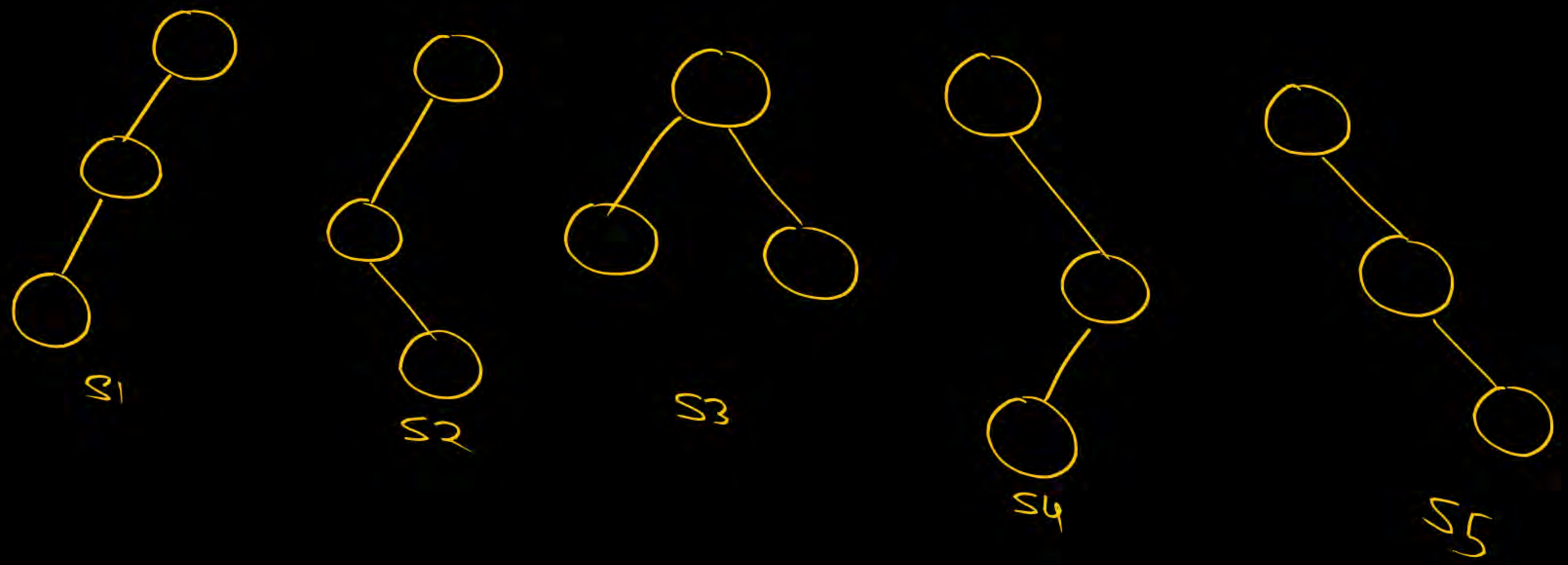
e)

f)

$$\boxed{\#\text{BSTs with } n \text{ keys} = \frac{{}^{2n}C_n}{n+1}}$$

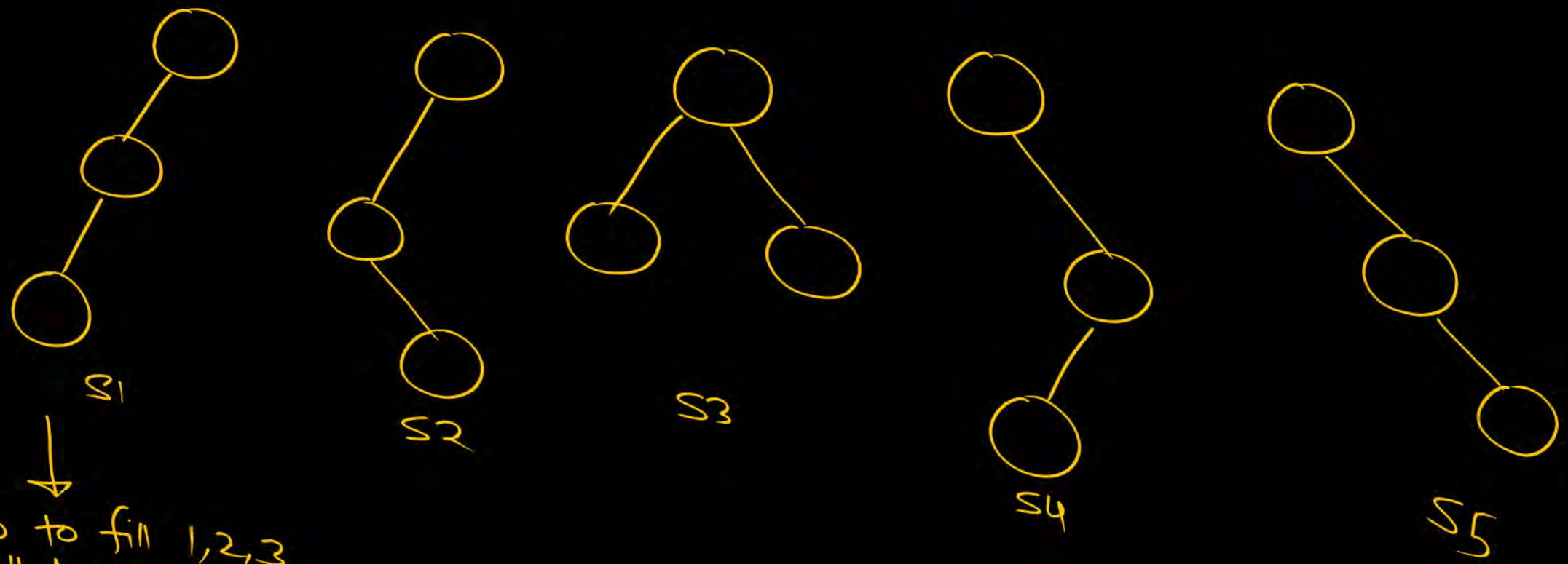
$n = 3$ Structure $\Rightarrow \frac{{}^{2n}C_n}{n+1} = 5$

Keys $\Rightarrow 1, 2, 3$



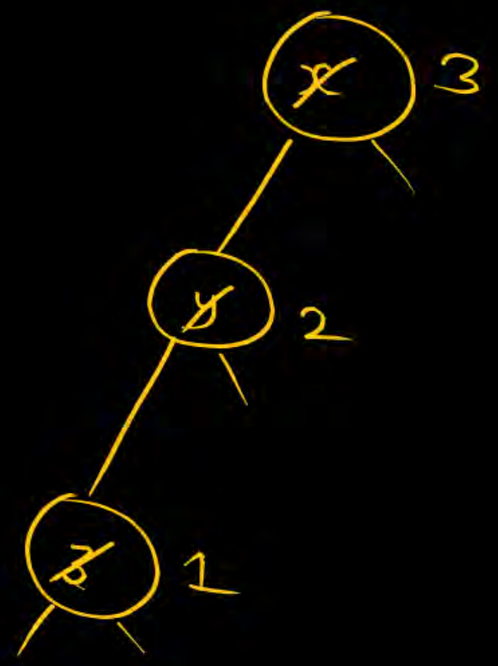
$n = 3$ Structure $\Rightarrow \frac{{}^{2n}C_n}{n+1} = 5$

Keys $\Rightarrow$ 1, 2, 3



How to fill 1, 2, 3 so that this structure will become a BST

1, 2, 3



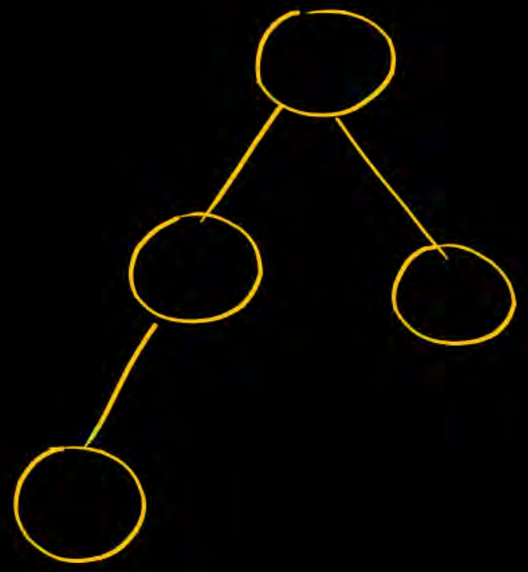
$z < y < x$

Given a n-node unlabelled structure

&

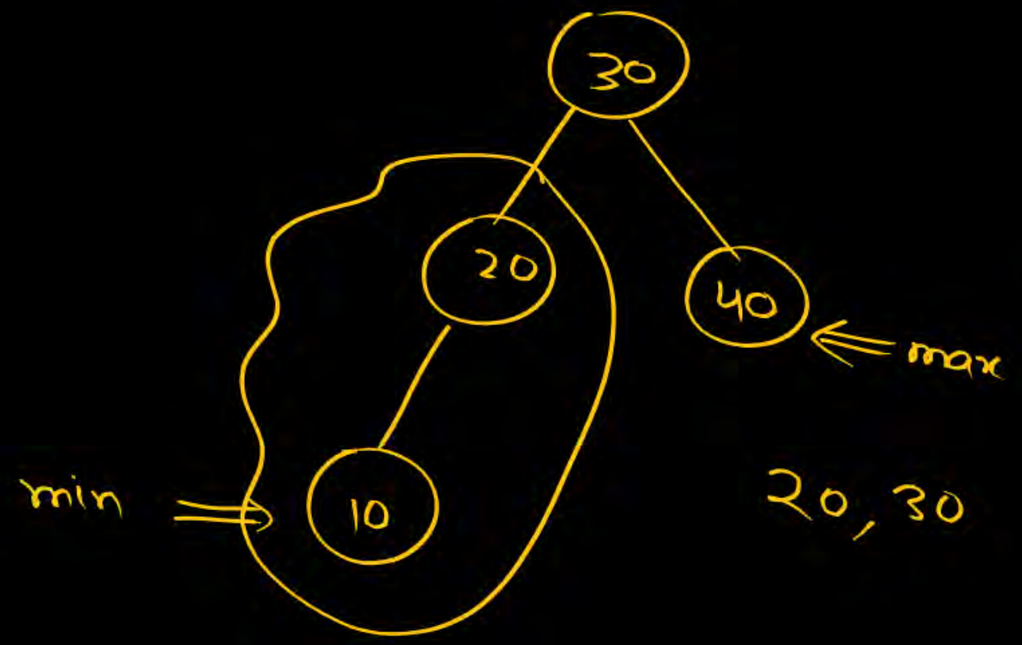
also n-distinct keys are given.

ways to fill the structure so that we get BST



Keys

10, 20, 30, 40



Keys

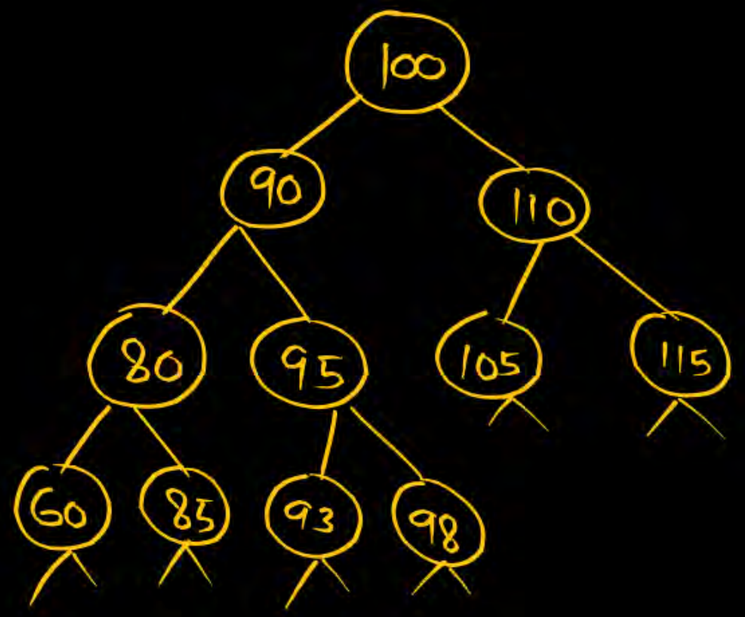
10, 20, 30, 40

20, 30

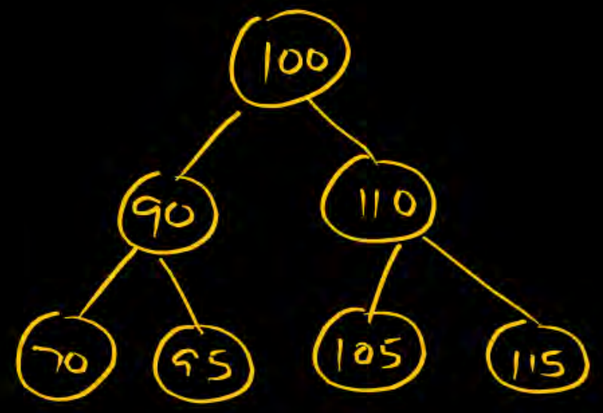
Q # BSTs with n keys $\Rightarrow$ $\begin{pmatrix}\text{structure with} \\ n \text{ nodes}\end{pmatrix} \times 1$

$$= \frac{{}^{2n}C_n}{n+1}$$

2 The inorder traversal of a BST is always increasing order of keys.



In: 60, 80, 85, 90, 93, 95, 98, 100, 105, 110, 115



Pre: 100, 90, 70, 95, 110, 105, 115

Pre : Given

In : 70, 90, 95, 100, 105, 110, 115 → Unique BST → Post-order

Given the preorder of some BST :

Pre : 100, 90, 70, 95, 110, 105, 115

then the postorder traversal of the BST is : _____

a)

b)

c)

d)

Preorder $\longrightarrow$ unique tree

Pankaj sir PW

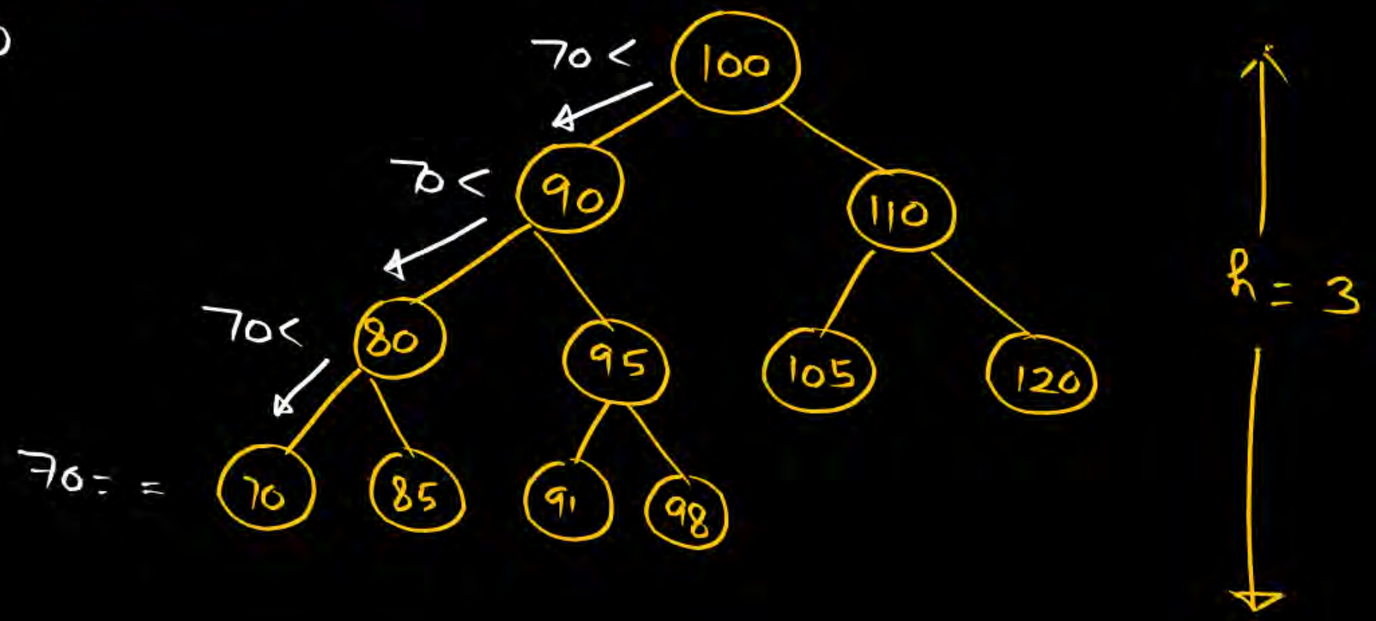
Search in a BST

Key = 70

$\#comp = 4$

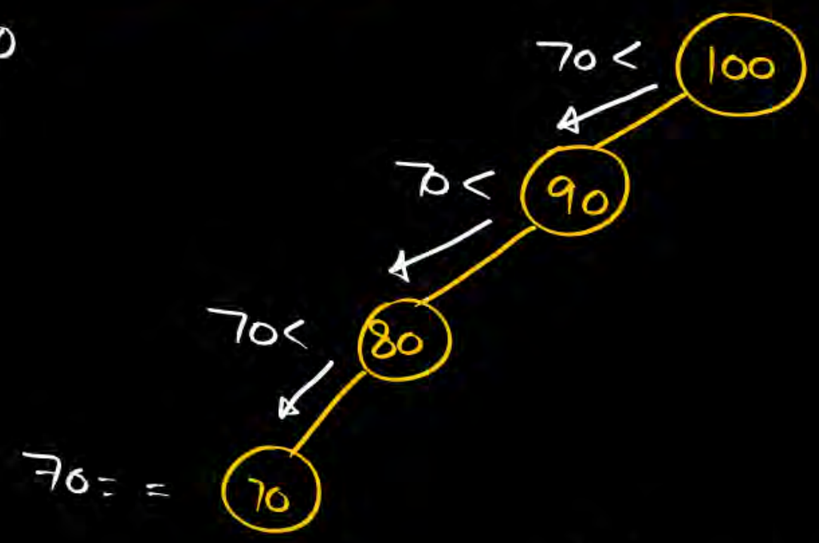
$= (h+1)$

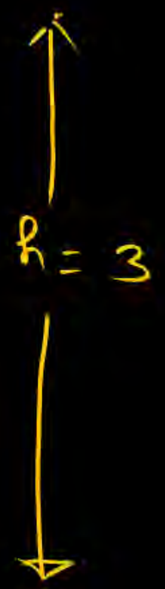
$= O(h)$



Search in a BST

Key = 70





$\#\text{comp} = 4$

$= (h+1)$

$= O(h)$

Skewed binary tree

Search

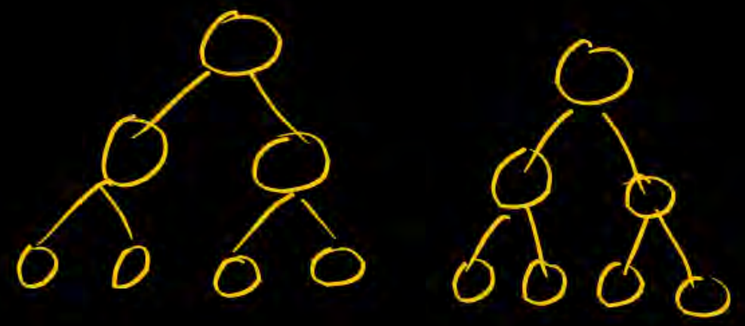
$O(h)$

FBT
CBT

$O(\log_2 n)$ ✓

Skewed

$O(n)$ ✓



Insertion in a BST

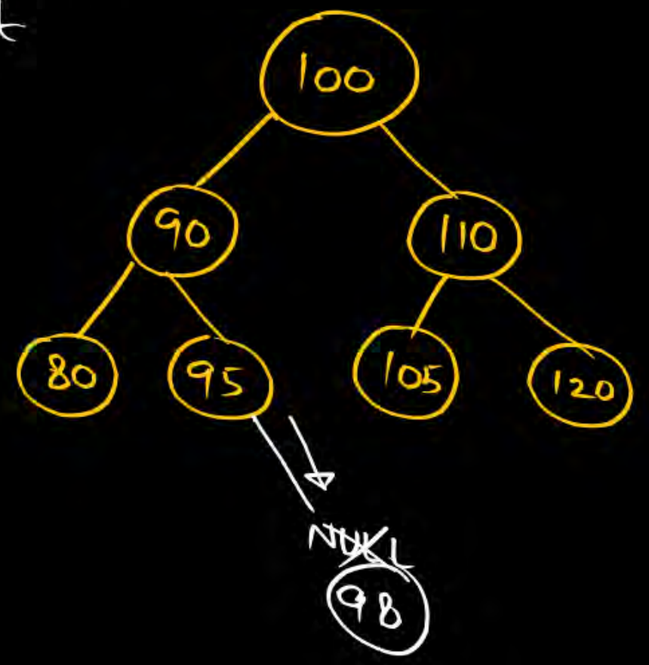
insert : 98

a) Search

b) Insert

right of 95

Complexity



Deletion from BST

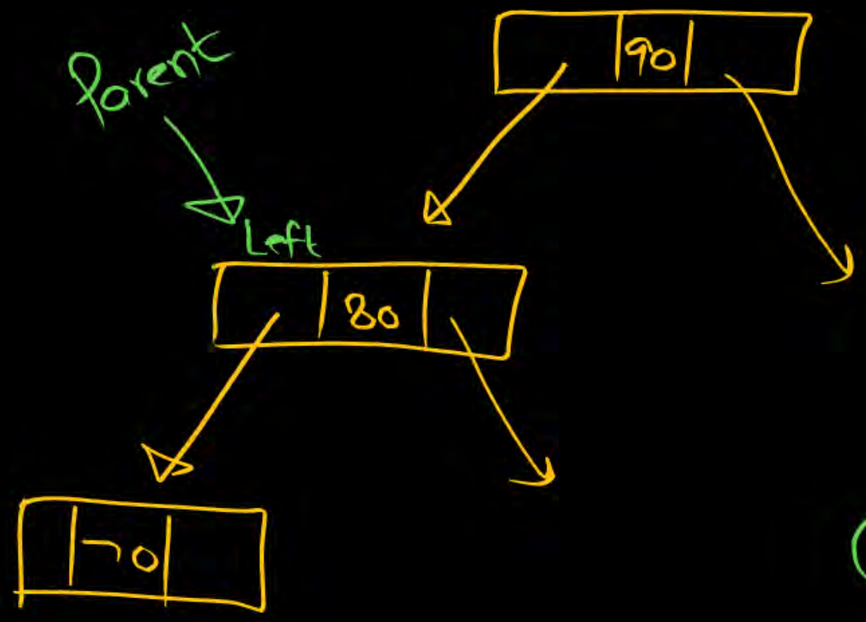
a) Delete a node with 0-child (leaf node)

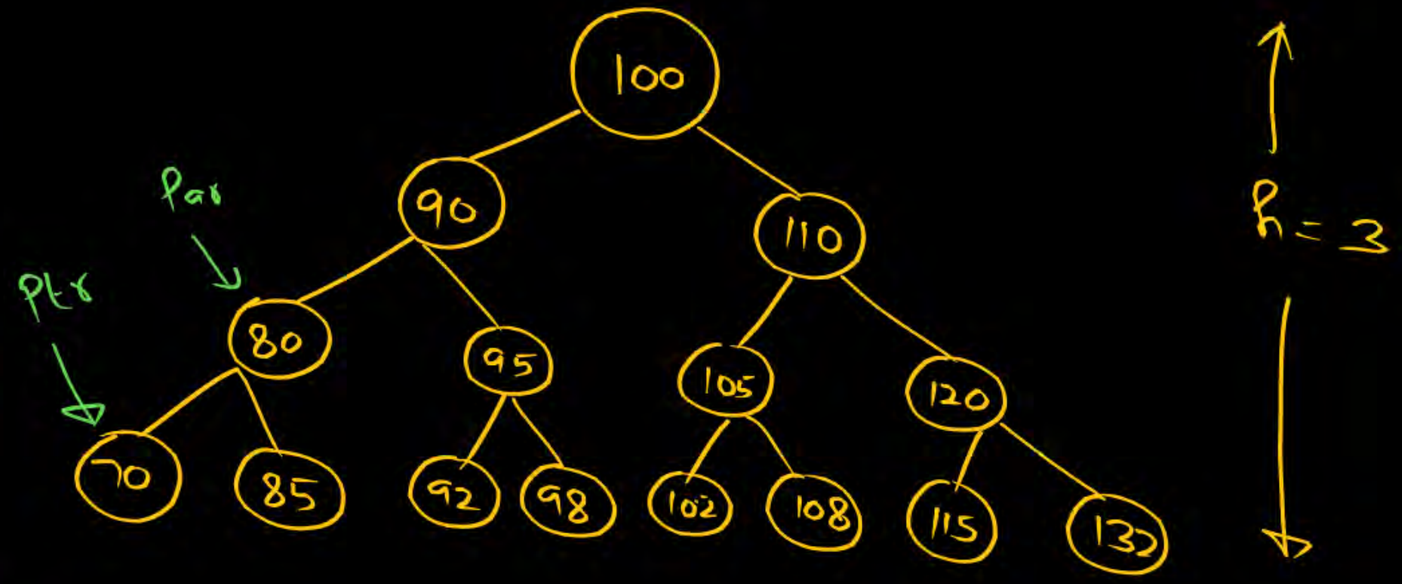
b) Delete a node with 1-child

c) Delete a node with 2-child

Deletion of leaf node

delete 70

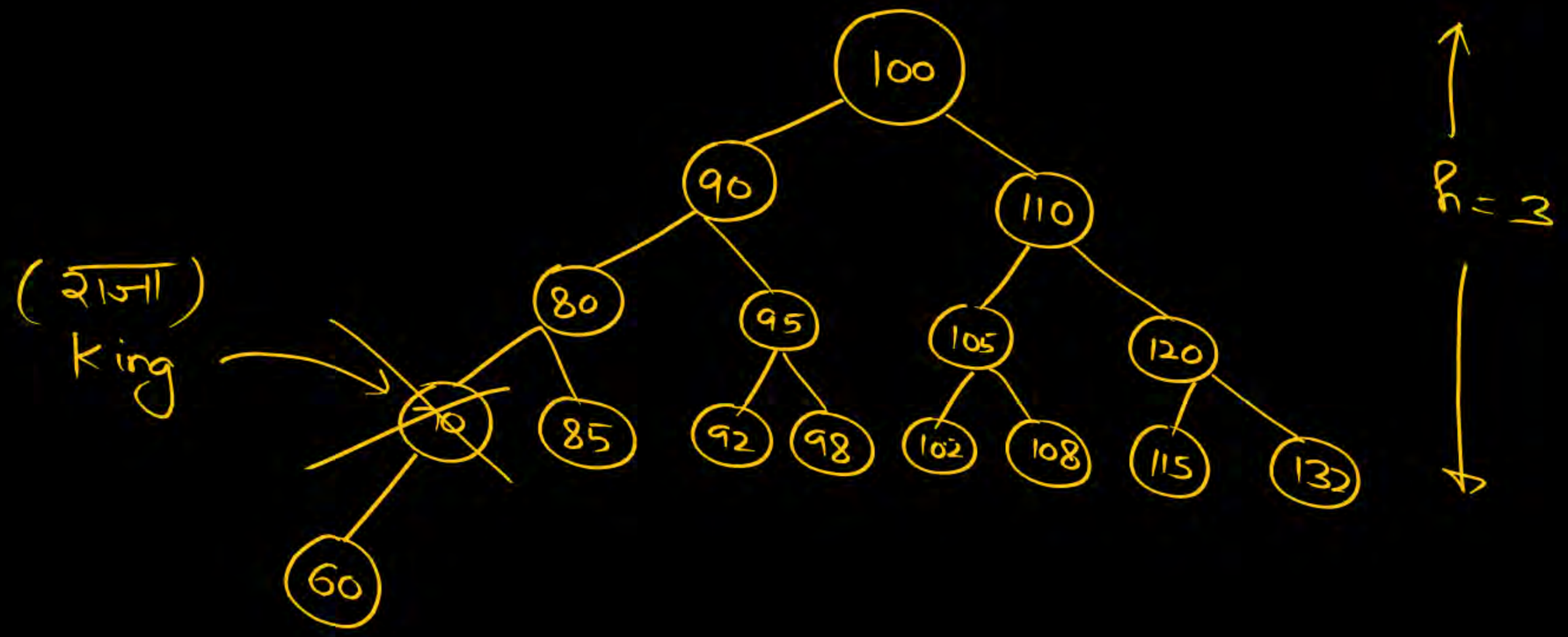




Node to be deleted

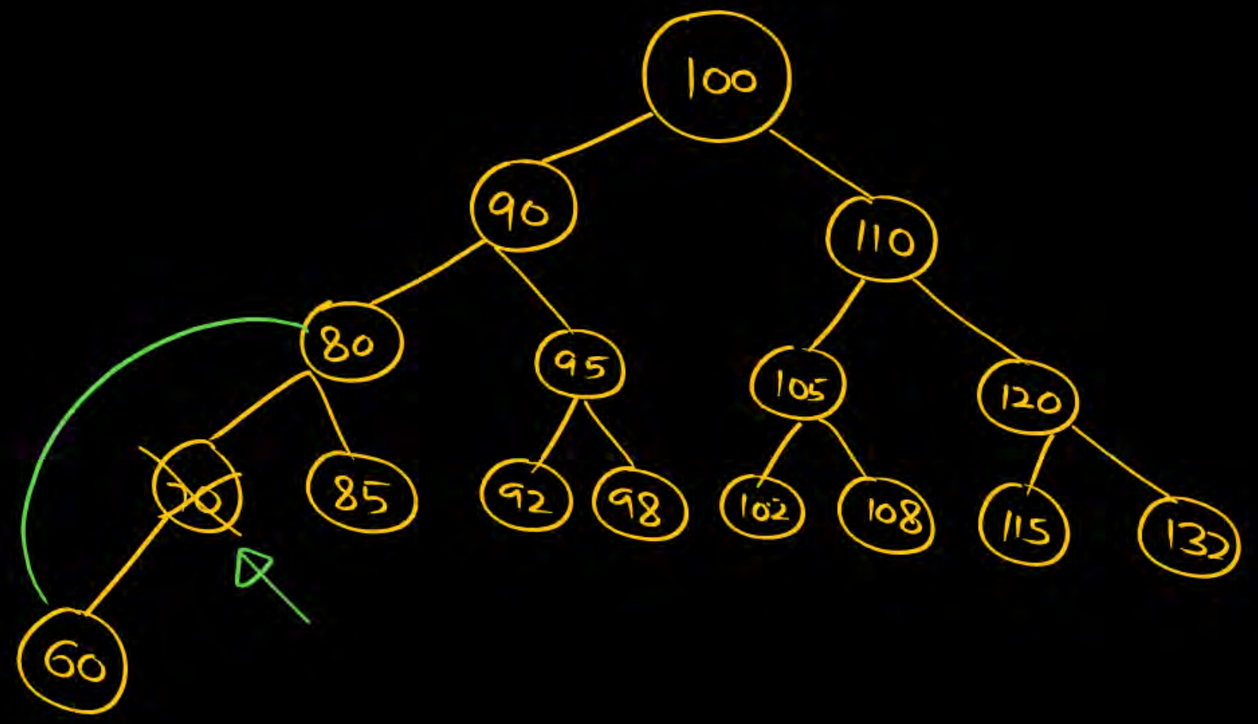
1. We need to identify the parent pointer (of node to be deleted), which is pointing to deleted node
2. Set this parent pointer to NULL.

Deletion of node with exactly 1-child



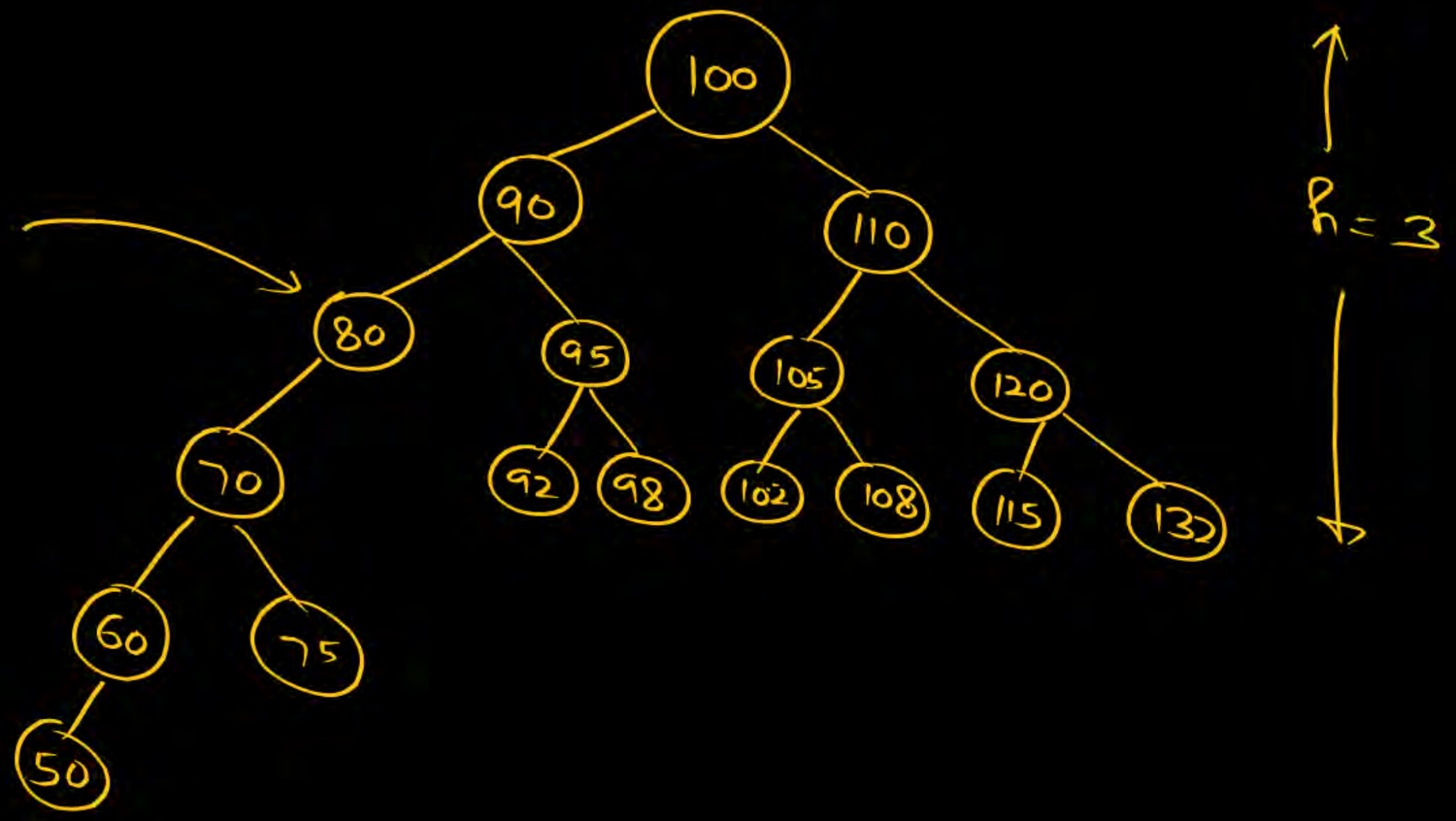
Deletion of node with exactly 1-child

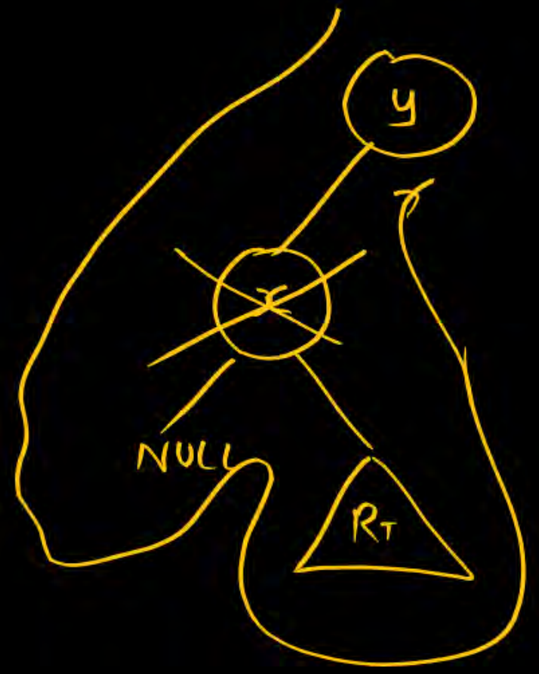
(राजा)
King

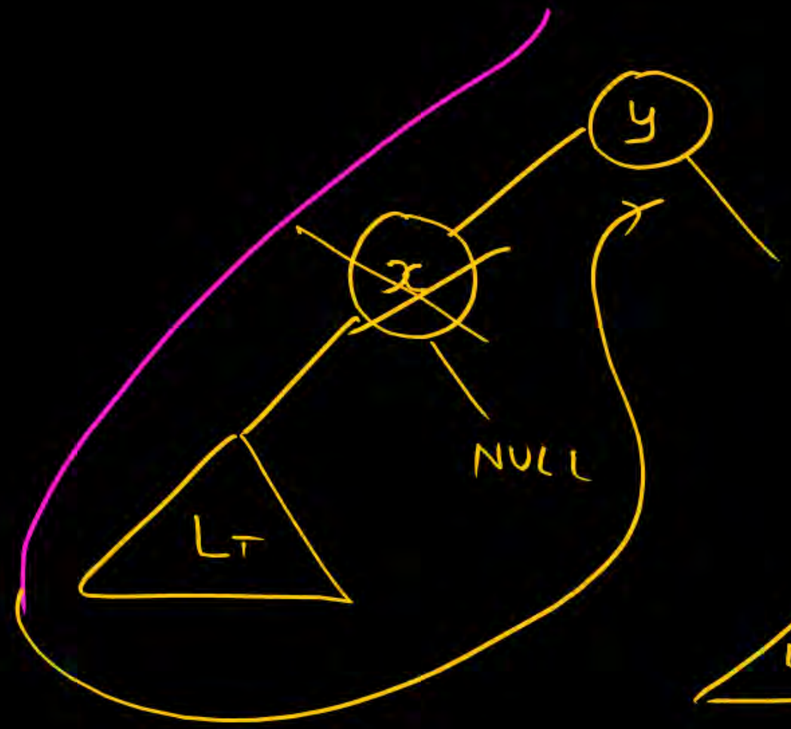


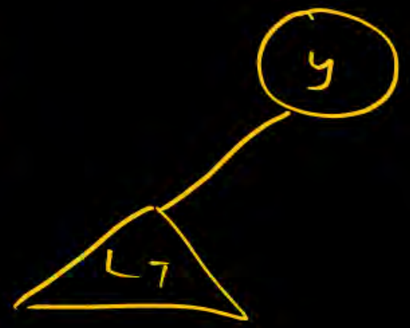
$h = 3$

Deletion of node with exactly 1-child

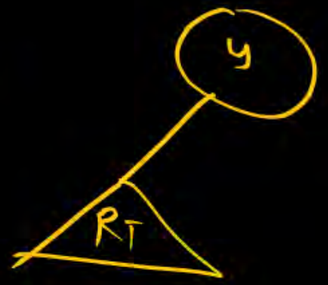






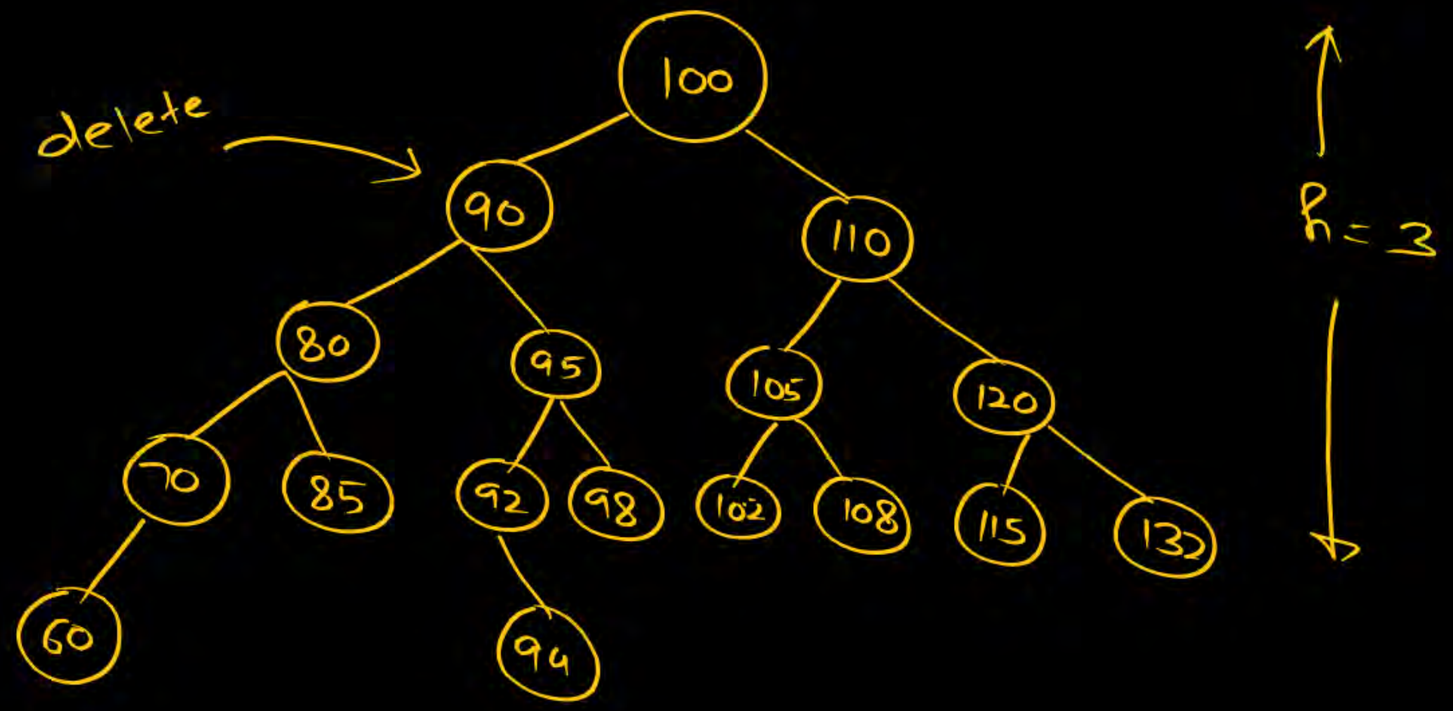


$\cancel{x}$, R_T, y

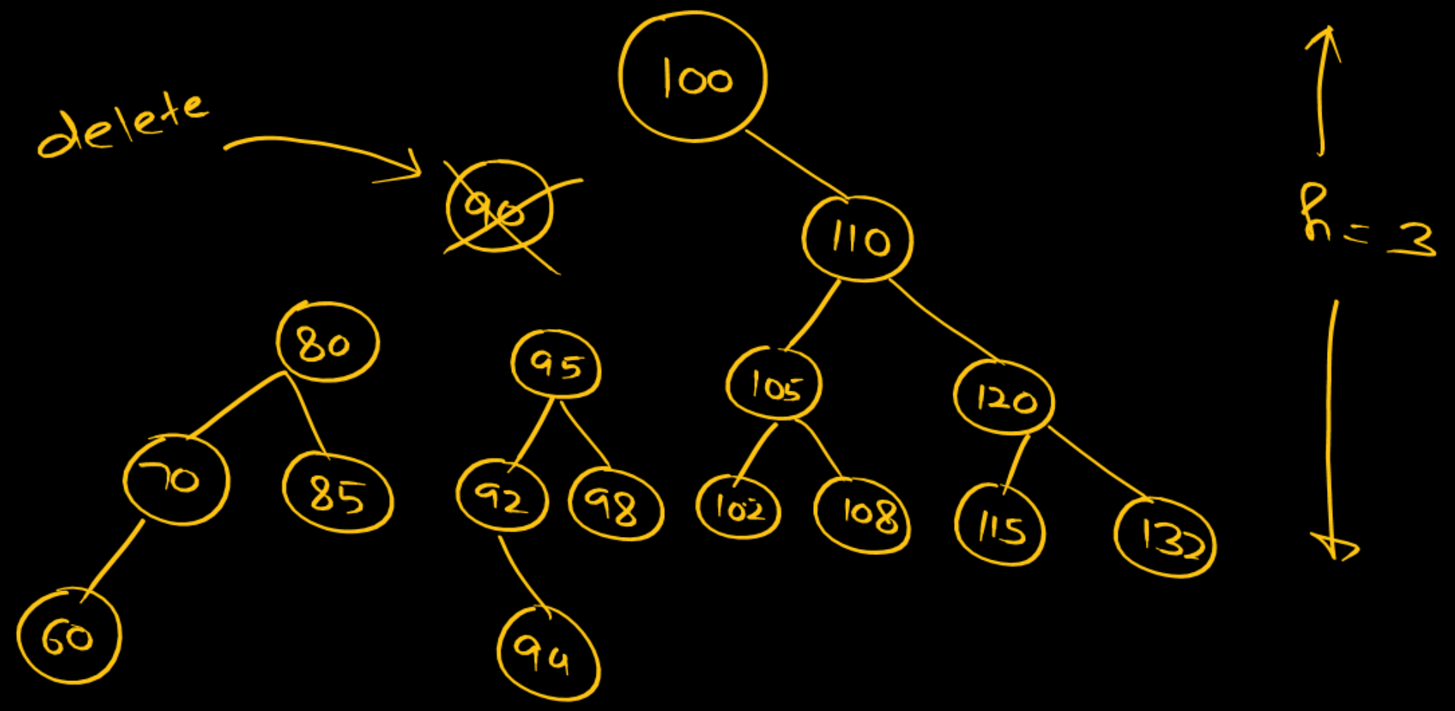


$L_T(x)$, x, y

Deletion of node with 2-childs



Deletion of node with 2-childs

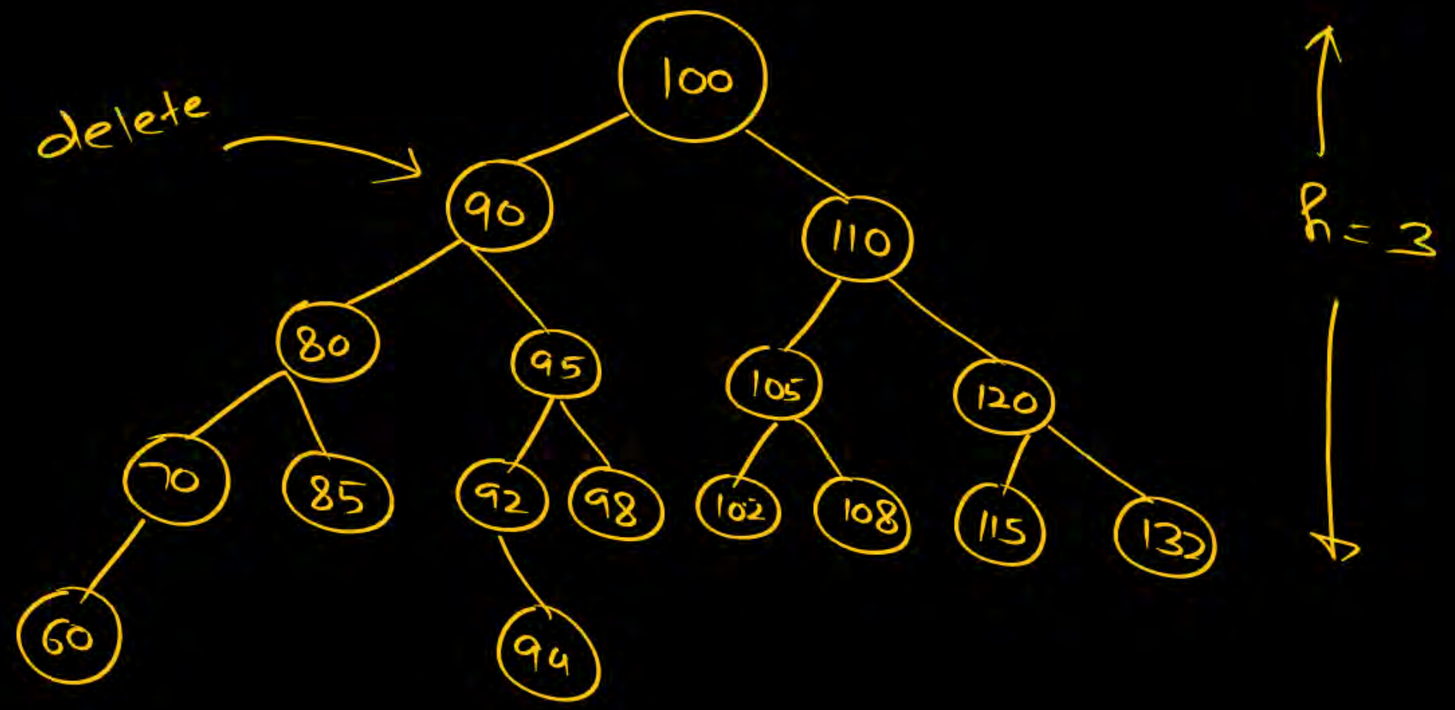


Deletion of node with 2-childs

(i) 1st way: Replace the key to be deleted by the largest key in Left subtree of node to be deleted & then perform deletion

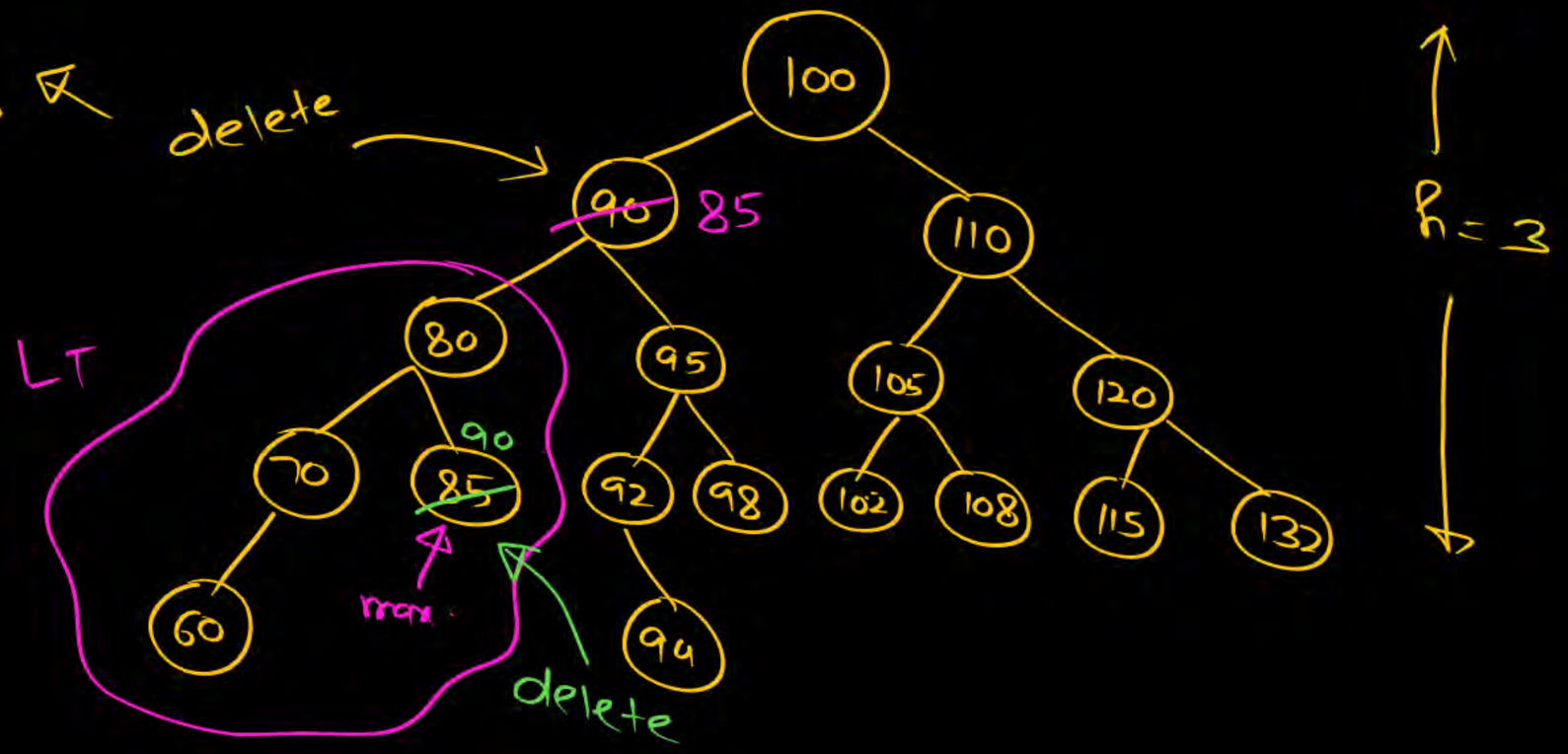
OR

ii) 2nd way: Replace the key to be deleted by the smallest key in the right subtree of node to be deleted & then perform deletion.

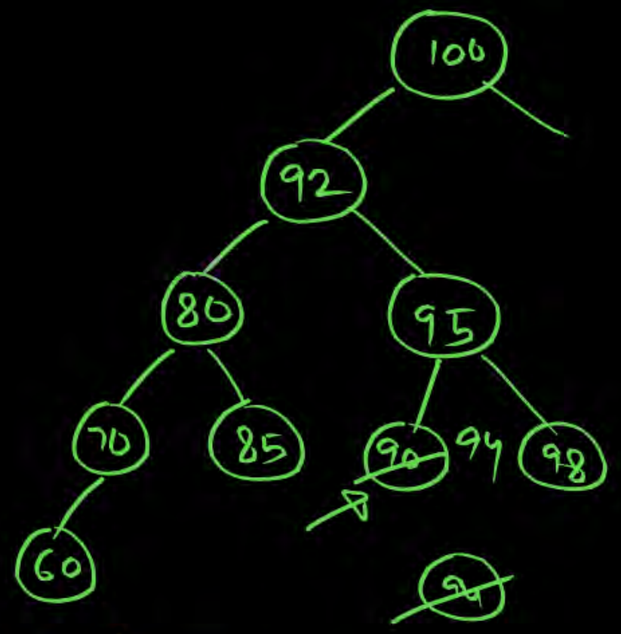


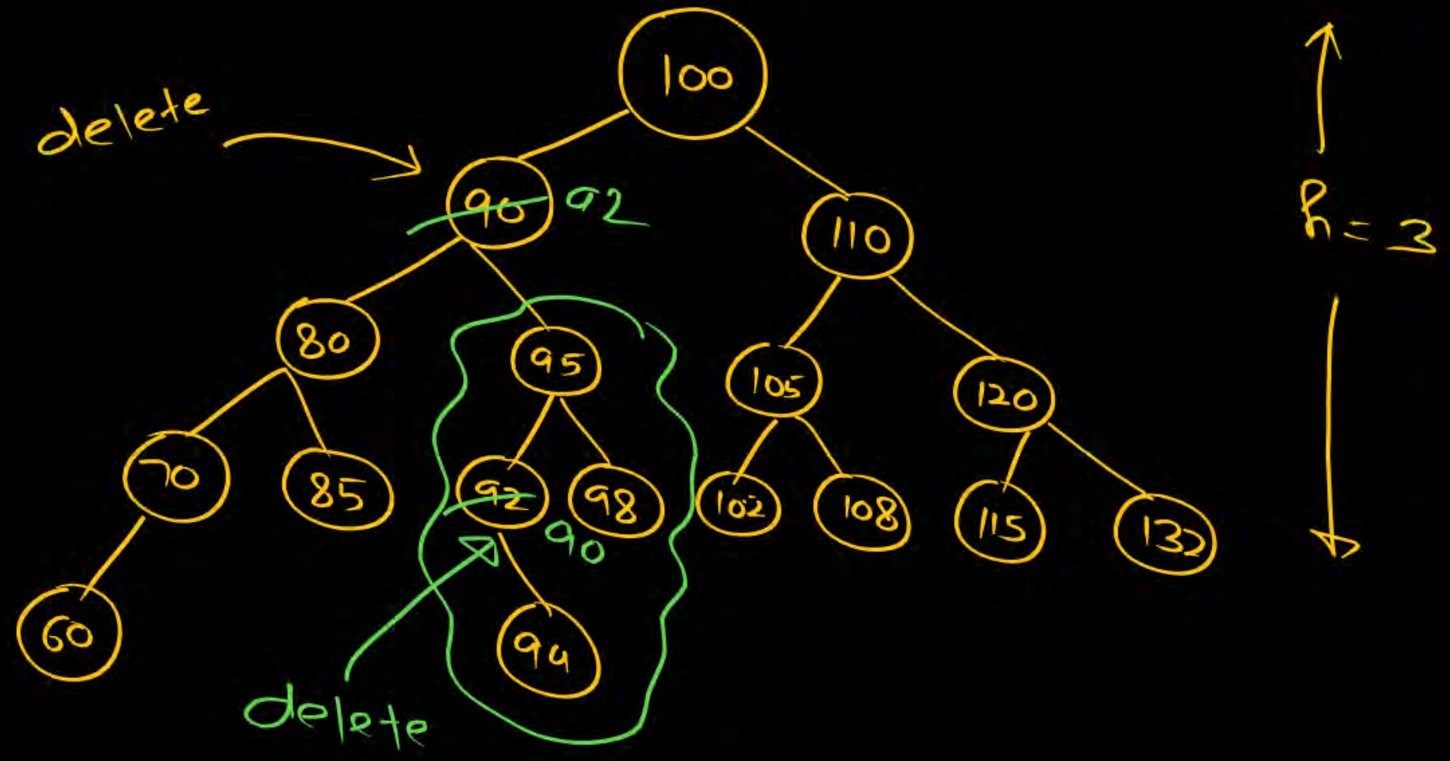
Deletion of node with 2-childs

(i) 1st way: Replace the key to be deleted by the largest key in Left subtree of node to be deleted & then perform deletion



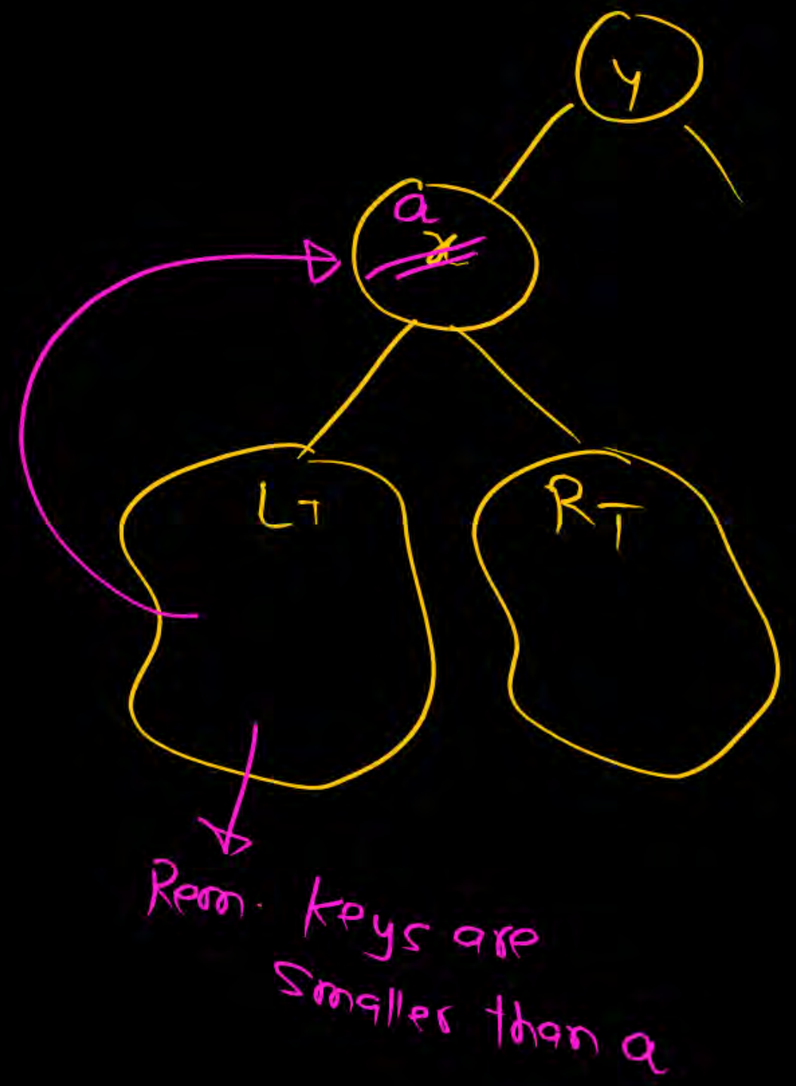
Deletion of node with 2-childs



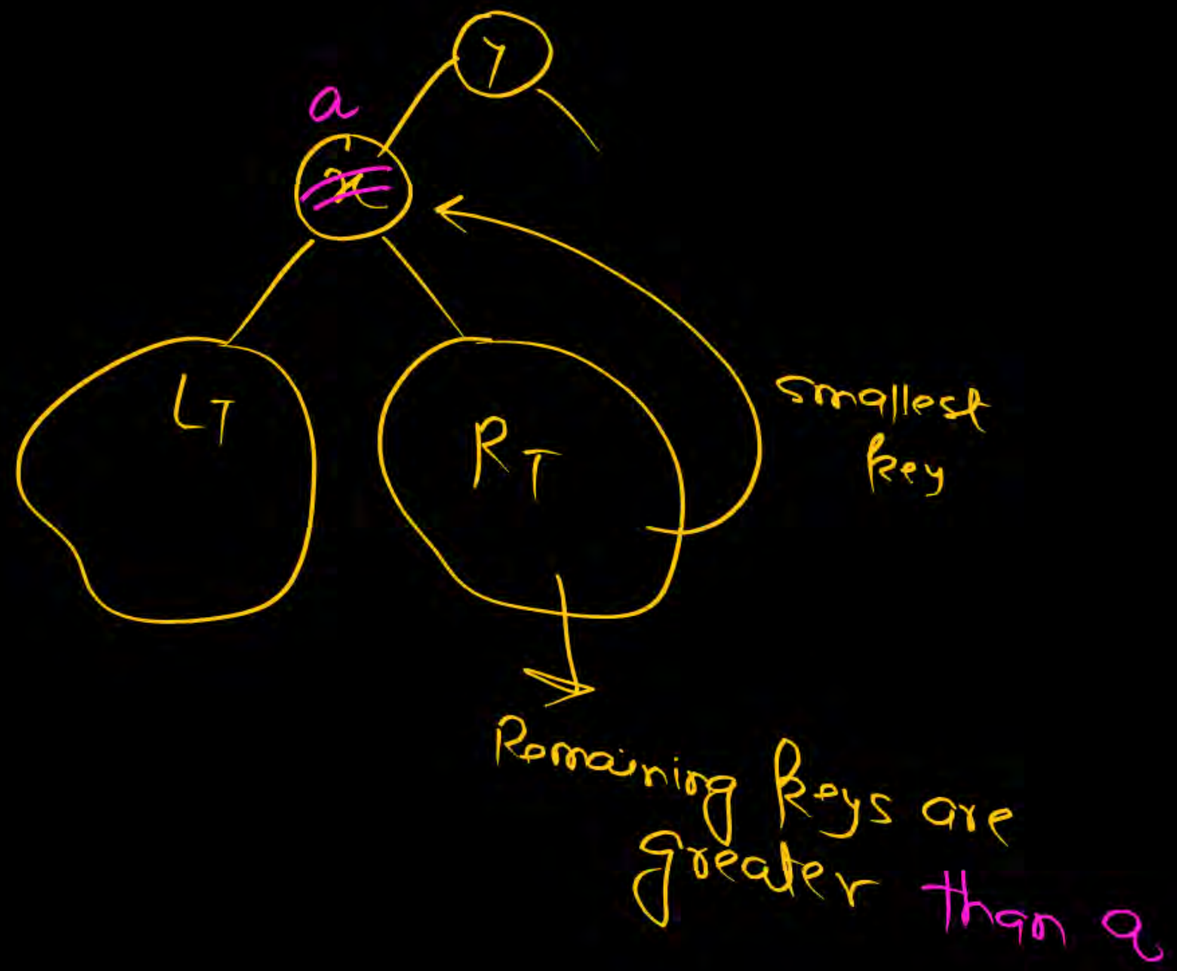


ii) 2nd way: Replace the key to be deleted by the smallest key in the right subtree of node to be deleted & then perform deletion.

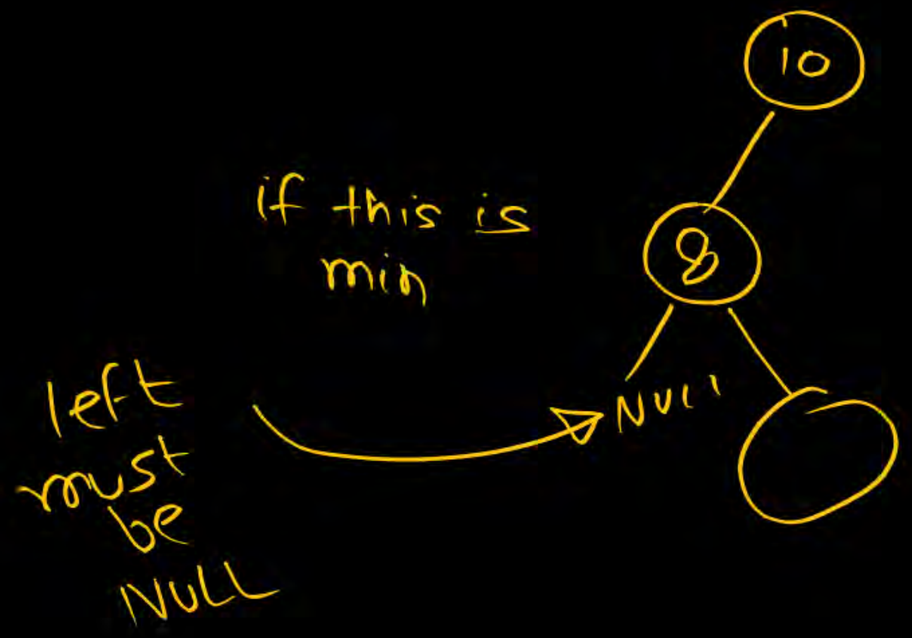
Case 1

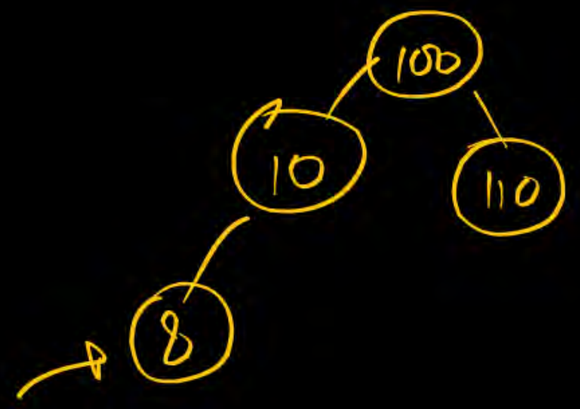


Case 2:



Node with min value can have 0 or 1 child (No child or Right child)



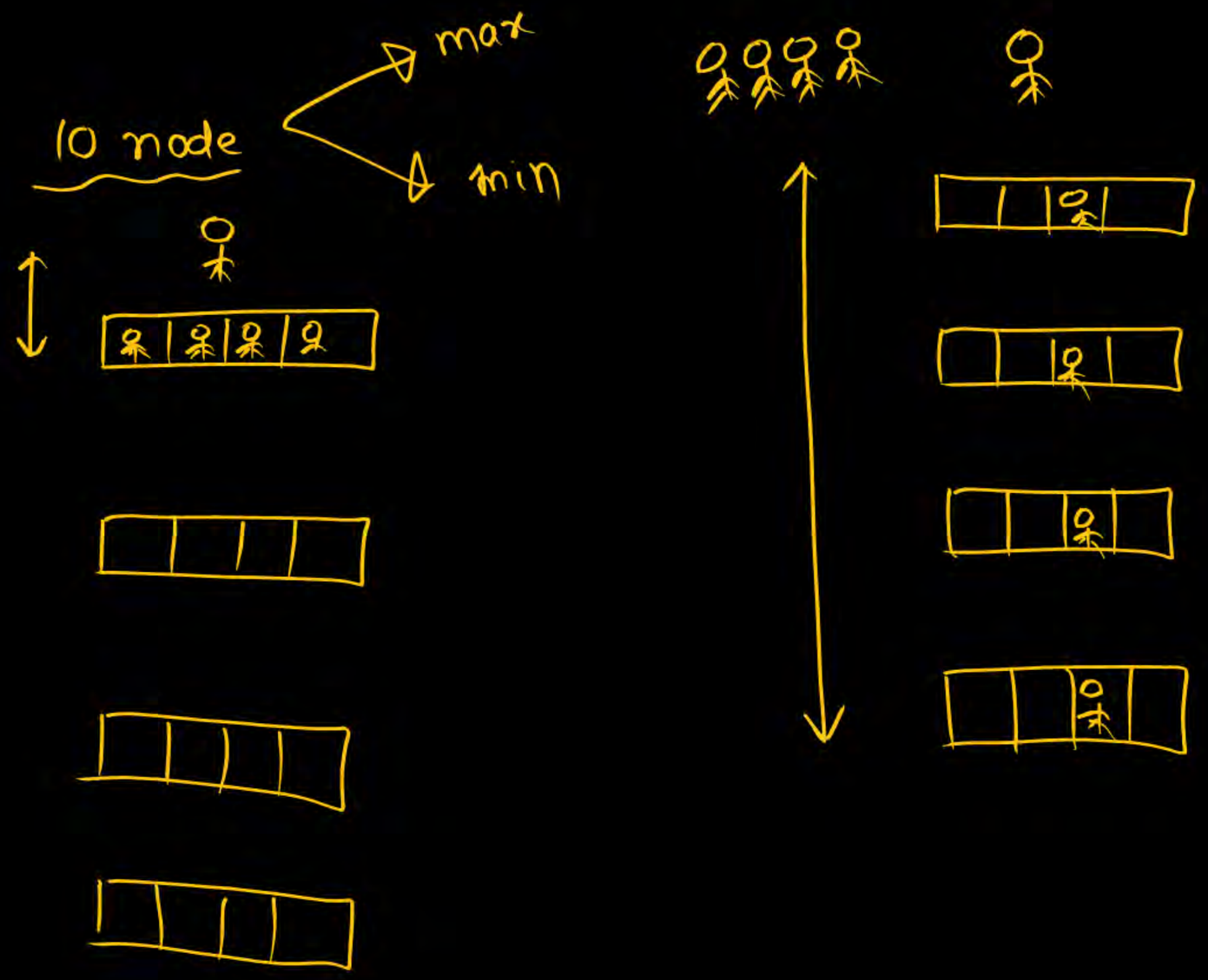


Node with max. value can have 0 or 1 child

deletion of node with 2-child

converted into deletion of node with 0/1 child

BST


10 node
max
min

C, DS

C++/STL

Python

Coding Problems 300+

review

THANK - YOU