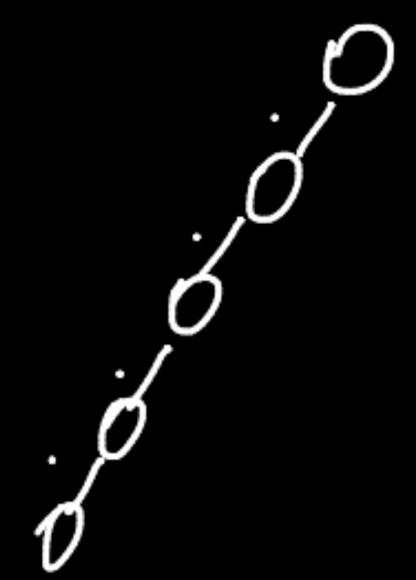
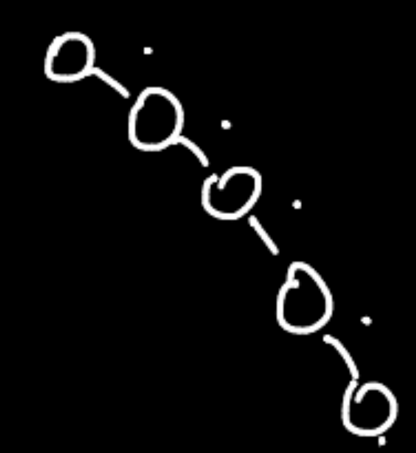
$n = 5$

Skewed



or

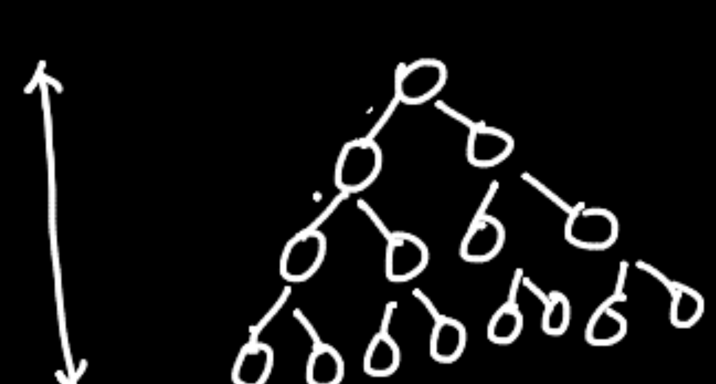


↳ atmost 2-children $n-1 \sim O(n)$

↳ perfect binary tree

every node has exactly two children

$n = 15$



$n-1$

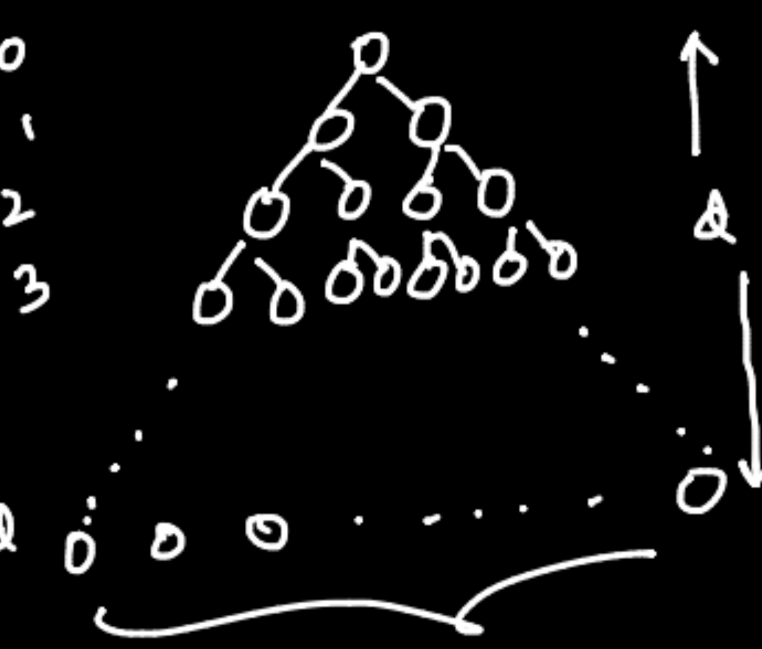
Level-0

Level-1

Level-2

Level-3

Level-d



2^0

2^1

2^2

2^3

$\vdots$

2^d

GP n

$$\underbrace{2^0 + 2^1 + 2^2 + \cdots + 2^d}_{} = n$$

$a = 1, \quad r = 2$

$N = d+1$

$$\frac{a(r^N - 1)}{r-1} = \frac{1(2^{d+1}-1)}{2-1}$$

$$n = 2^{d+1} - 1$$

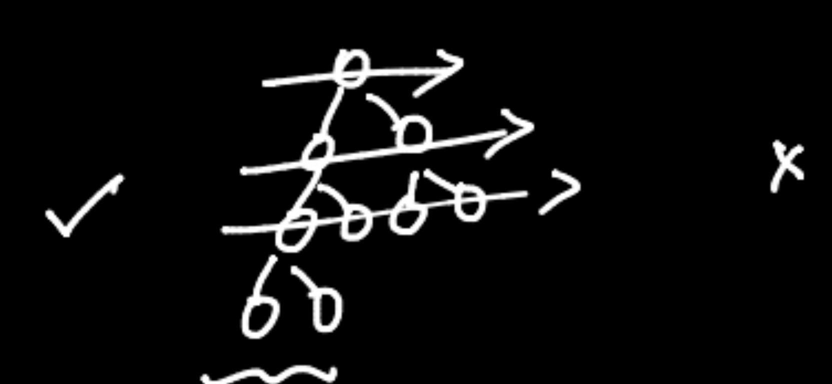
$$2^{d+1} = n+1$$

$$d+1 = \log(n+1)$$

$$d = \log_2(n+1) - 1$$

$$\boxed{h = \log_2(n+1) - 1}$$

Complete Binary Tree



may / may not

$h \propto \log_2 n$

int A[6] =

$A[i] = x;$

1. you'll save space

Binary Search Tree

	Array (fixed)	Linked List	Sorted (fixed)
insertion	$O(1)$	$O(1)$	$O(n)$
search	$O(n)$	$O(n)$	$O(\log_2 n)$
deletion	$O(n)$	$O(n)$	$O(n)$

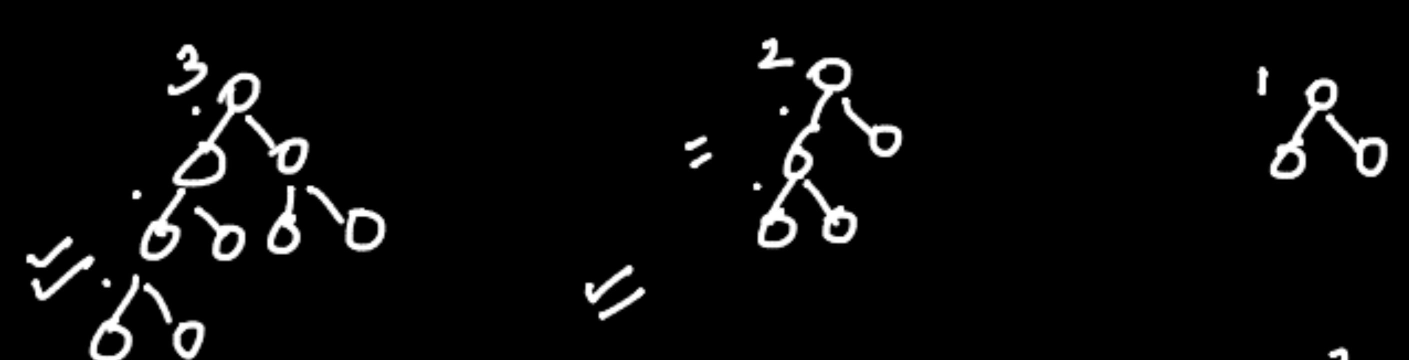
BST

on avg. $O(\log_2 n)$

⇓

almost balanced

if BST skewed $O(n)$



$n = 9$ $n = 5$ $n = 3$

$\Rightarrow \lfloor \log_2 n \rfloor \Rightarrow \lfloor \log_2 9 \rfloor = \lfloor 3.\ldots \rfloor = 3$

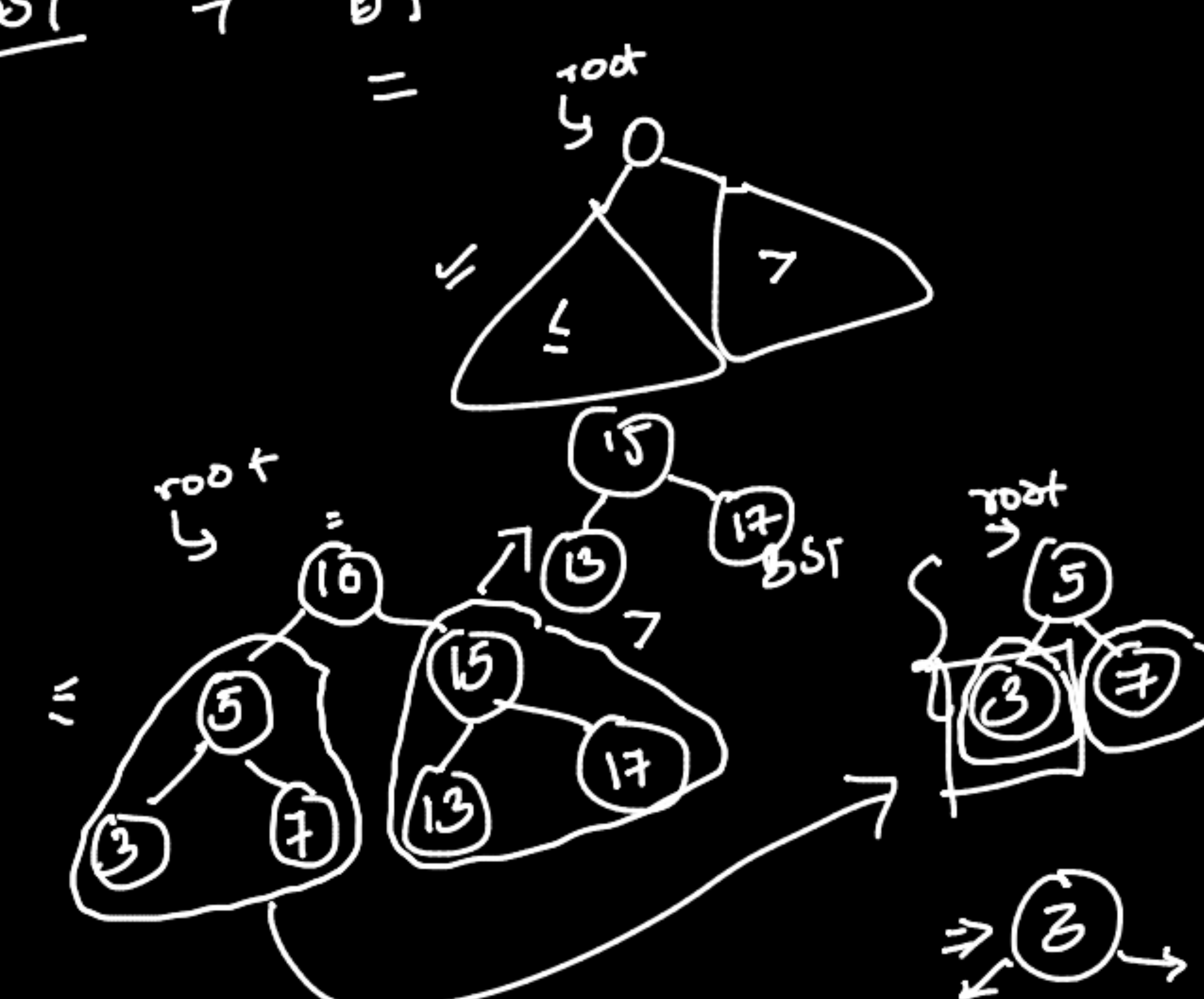
$\lfloor \log_2 5 \rfloor = \lfloor 2.\ldots \rfloor = 2$

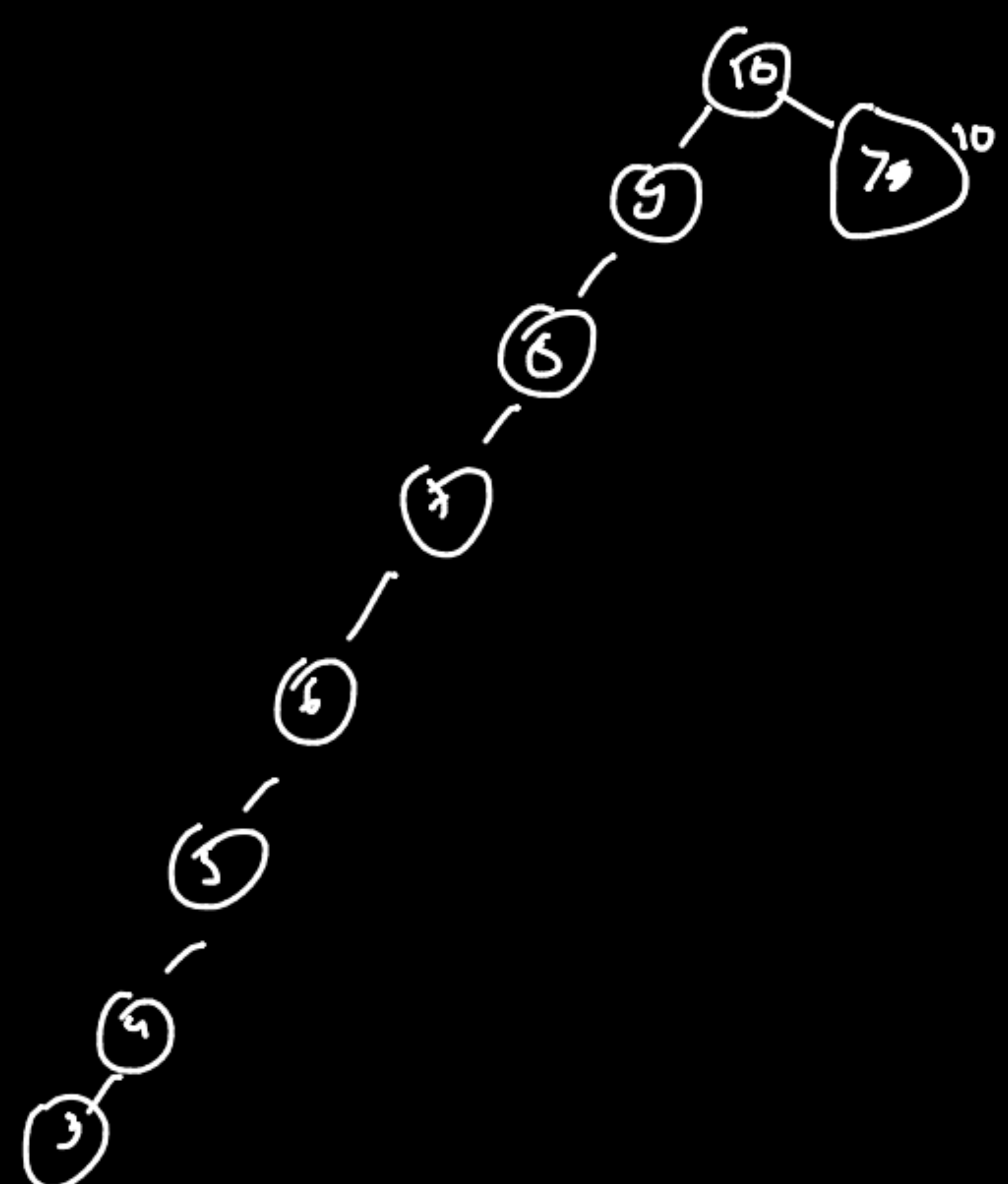
$\lfloor \log_2 3 \rfloor = \lfloor 1.(\ ) \rfloor = 1$

floor

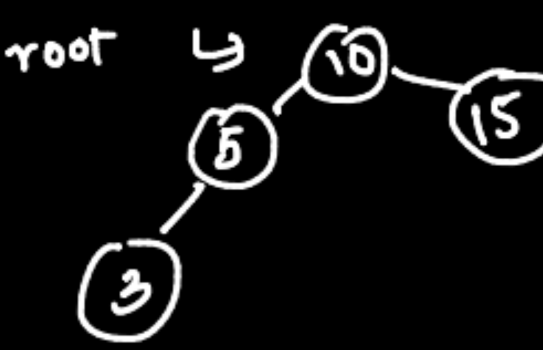
$\log_2 2 = 1$

BST → $\bar{\bar{B}}T$





root



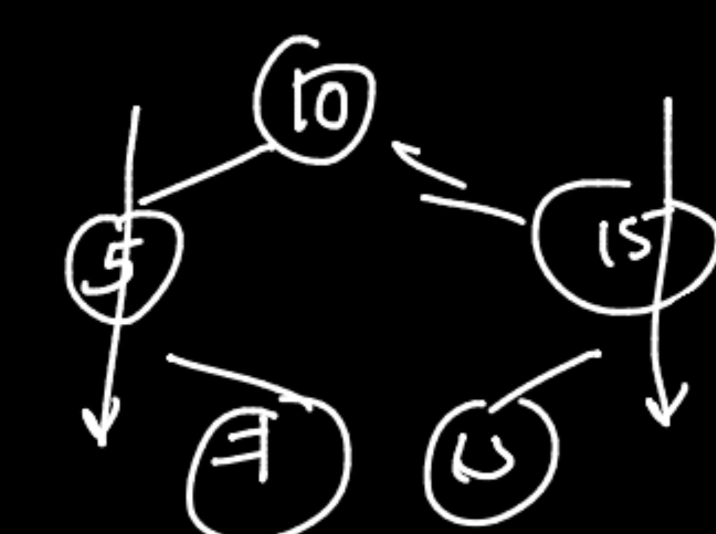
BST → insert(13) ⇒ insert into BST which is also BST

⇓

recursion

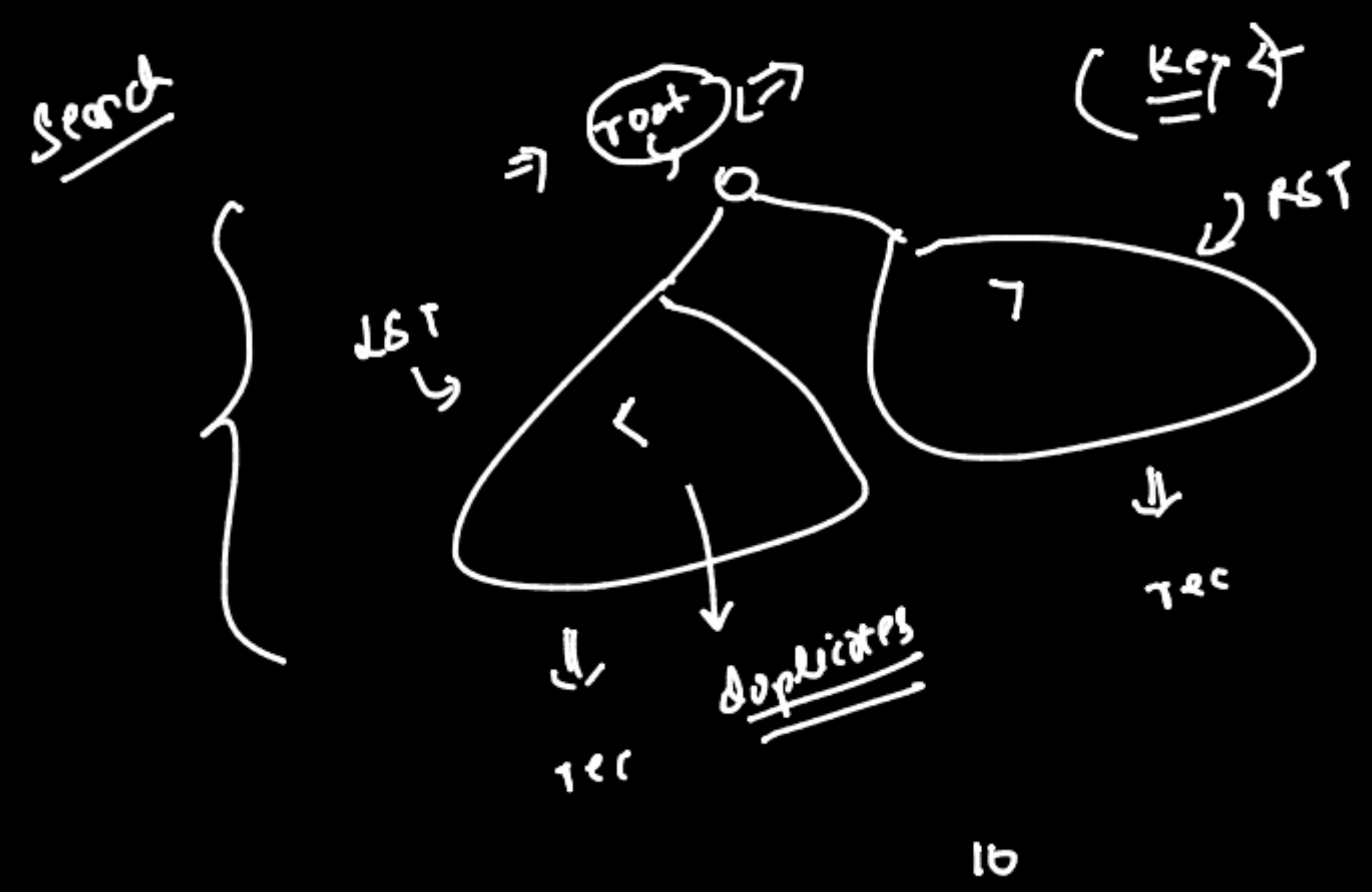
root

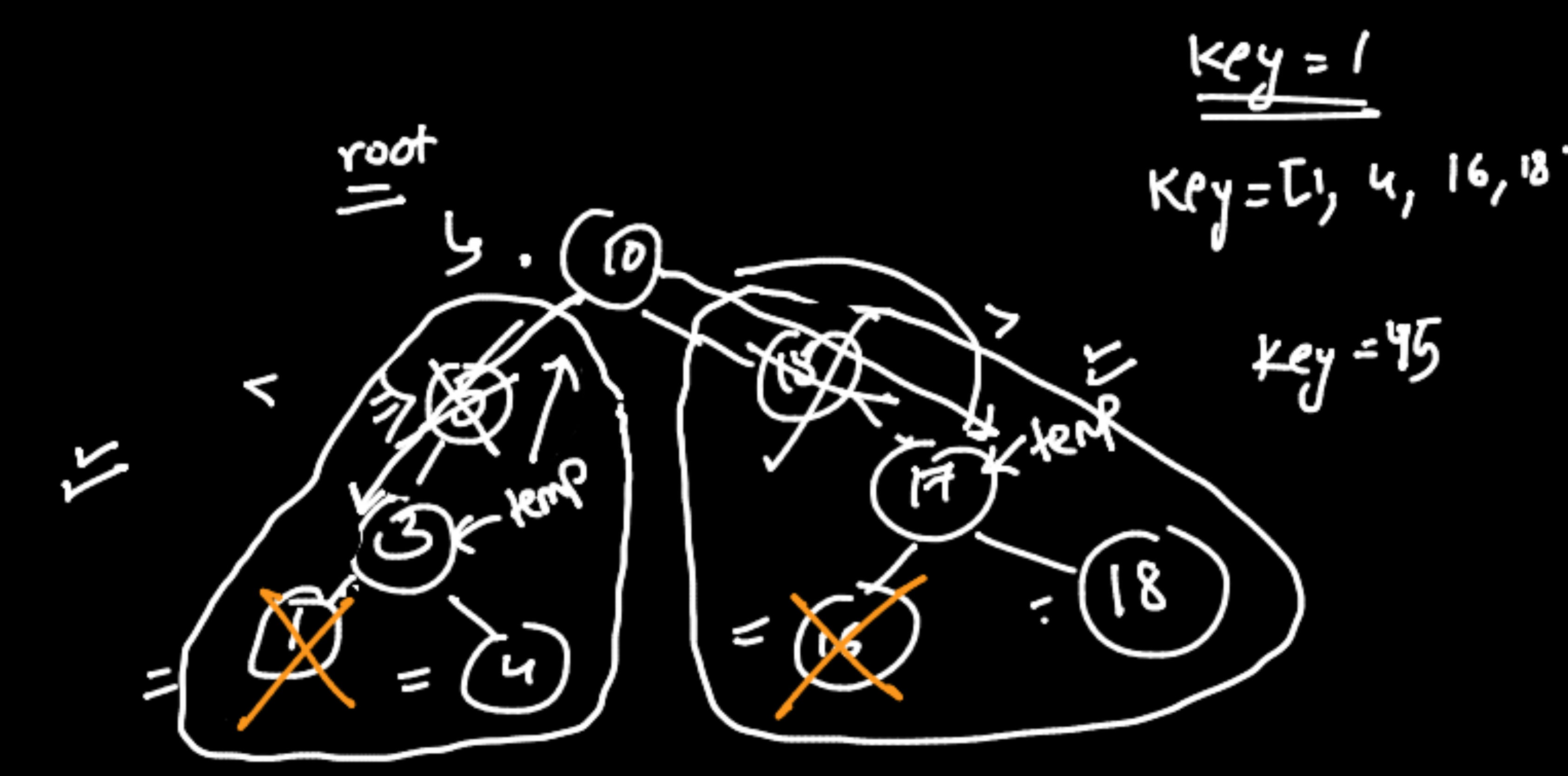
Not BST

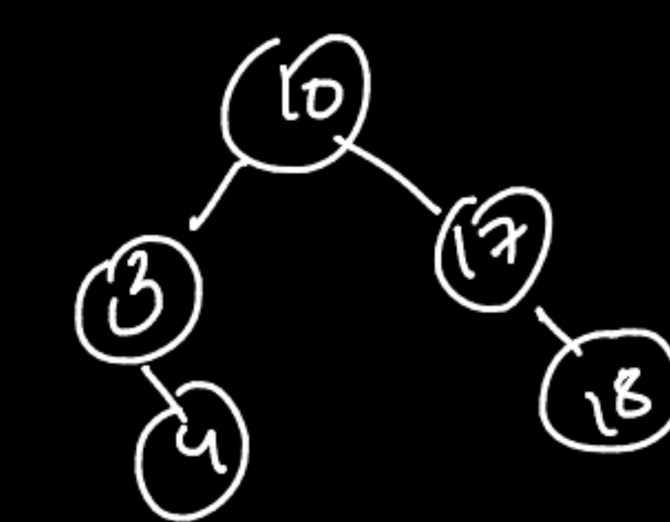


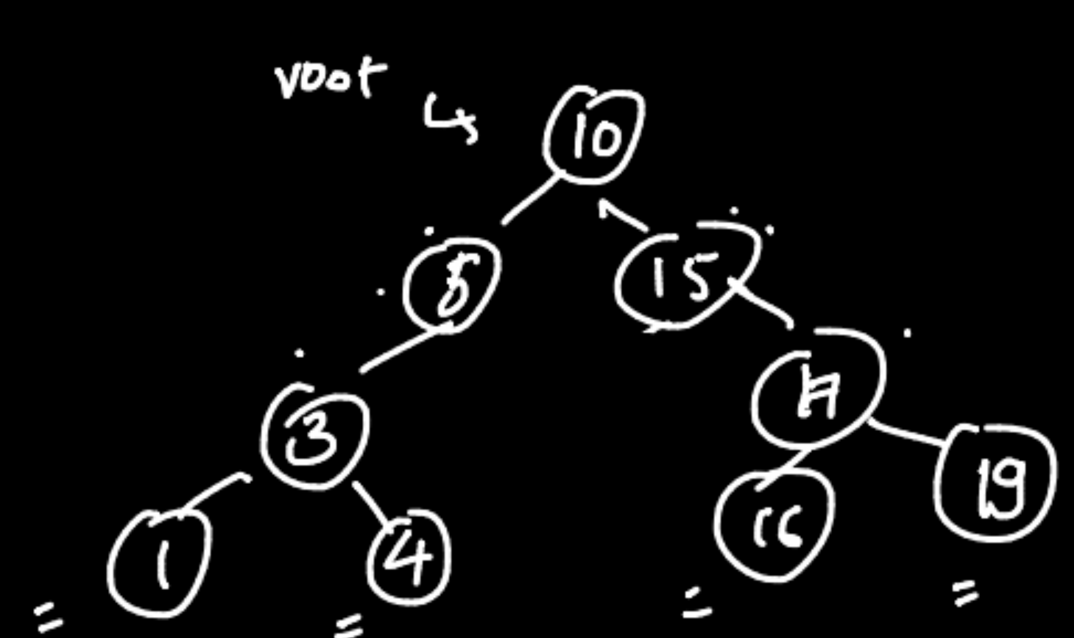
find min? ⇒ leftmost node

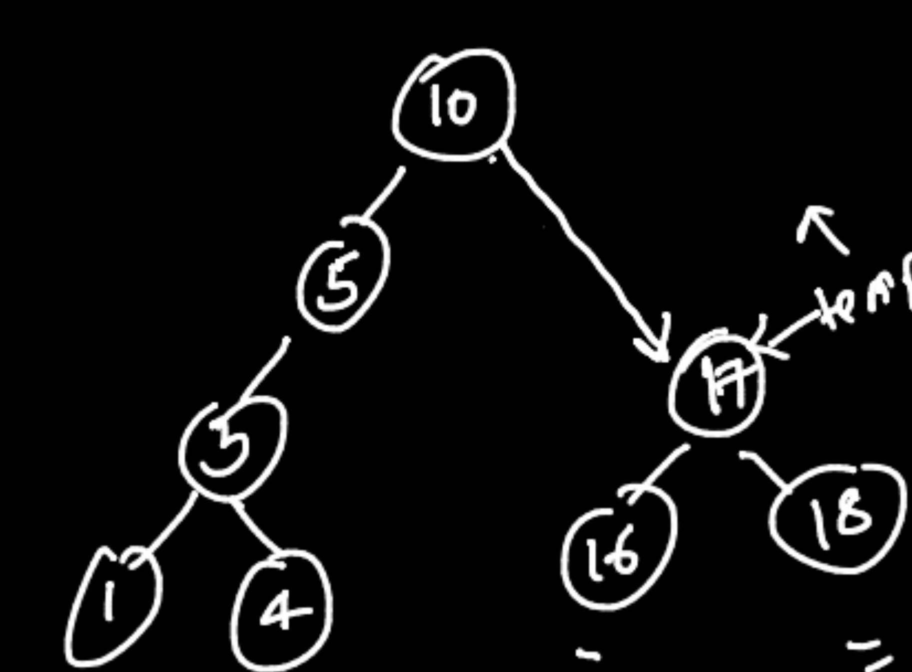
find max? ⇓ rightmost



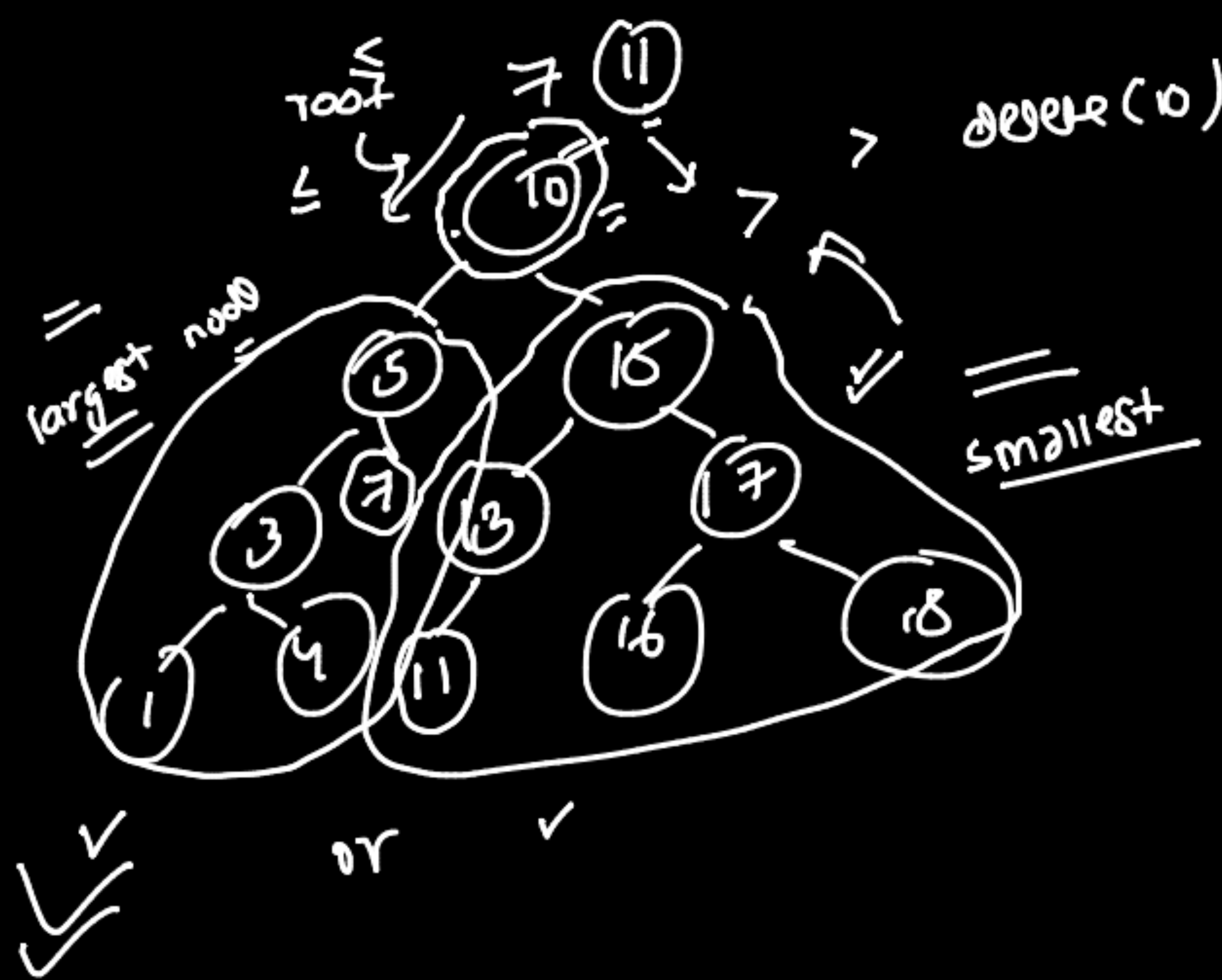


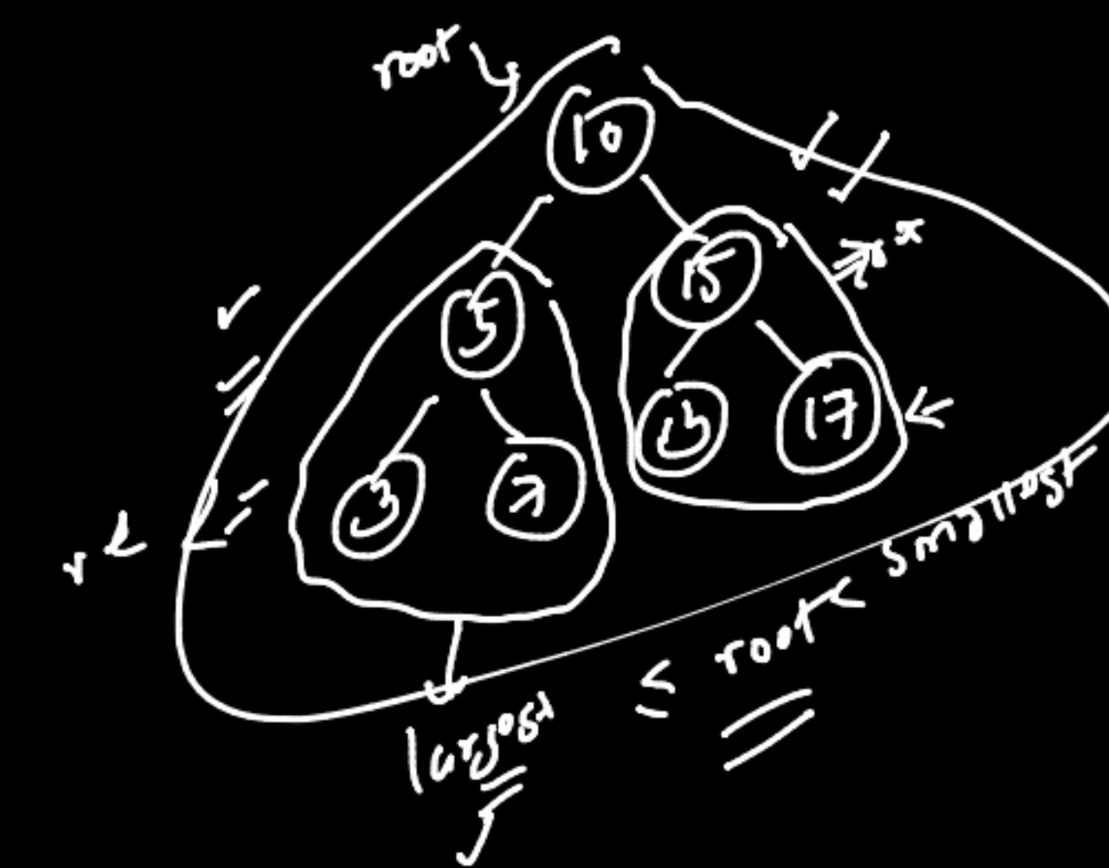






1. No child → leaf node (1, 16, 4, 19)
2. one-child → right or left (5, 15)
3. two-child → (10, 3, 17)


root
delete(10)
largest node
smallest
or


root
largest
≤ root < smallest