AVL [height Balanced Binary search tree]

Lecture 1

How height is Balanced?

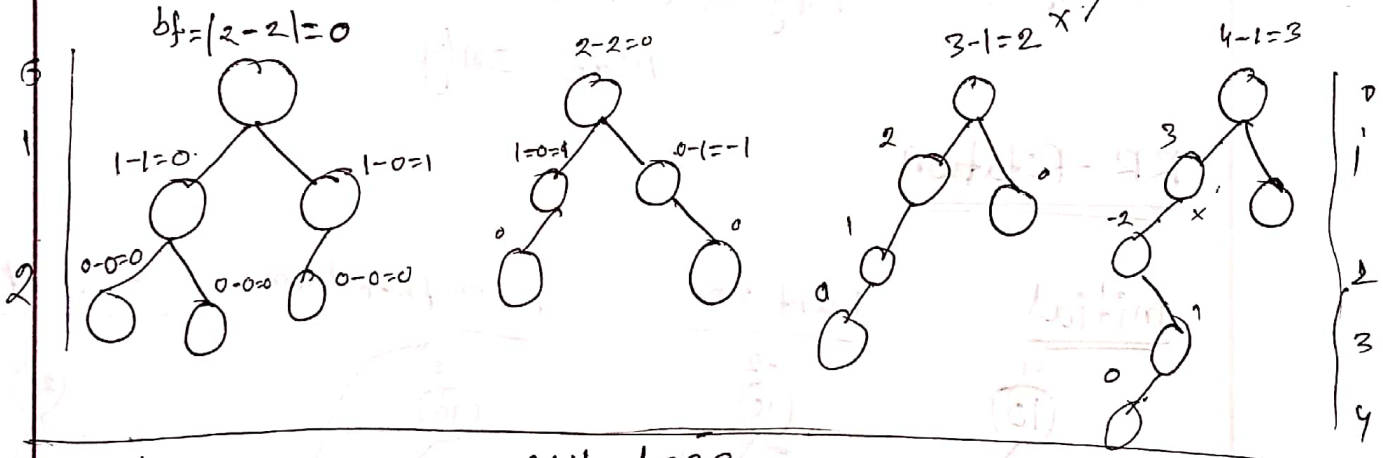
if is balanced factor = (height of left − height of Right)

$bf = (hl - hr) = \{-1, 0, 1\}$

$bf = |hl - hr| \leq 1 \rightarrow$ balanced

if $bf = |hl - hr| > 1 \rightarrow$ imbalanced

⇒ Find balanced factor of each node:



Lecture 2

AVL tree

Rotation for insertion

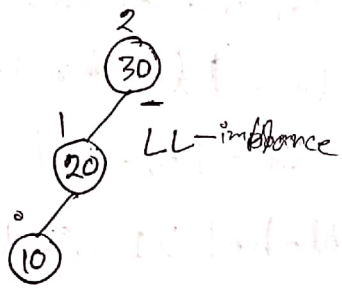
1. LL - Rotation
2. RR - Rotation
3. LR - Rotation
4. RL - Rotation

Rotation for insertion

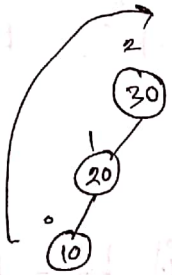
LL Rotation

Initial

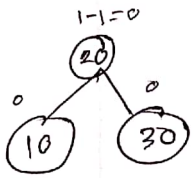
After insertion 10



perform Rotation



After Rotation



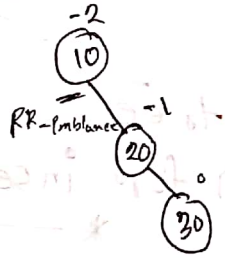
bf

if tree is bag ⇒ but Rotation done with (3) node only

RR-Rotation

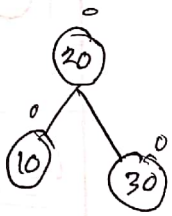
initial

insert 30



perform RR-Rotation

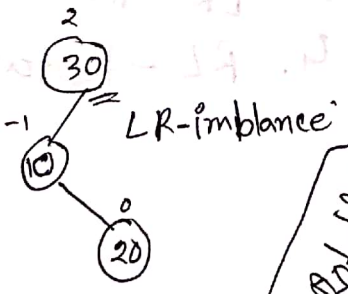
After RR-Rotation



LR-Rotation

initial

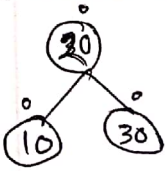
After insert 20 R



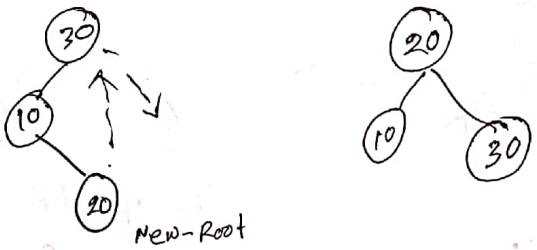
perform LR Rotation

Double Rotation

After LR Rotat



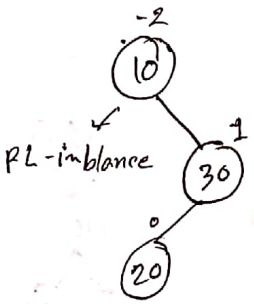
LR → direct Rotation :



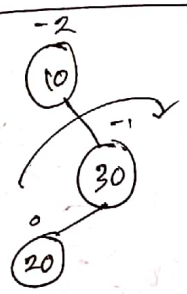
RL-Rotation

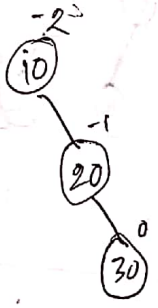
Initial

After Inserting 20

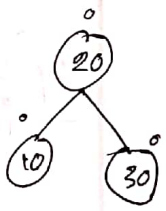


perform RL Rotation





After RL Rotat

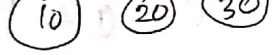


LL-Rotation
RR→Rotation
] single Rotation

LR-Rotation
RL-Rotation
] double Rotation

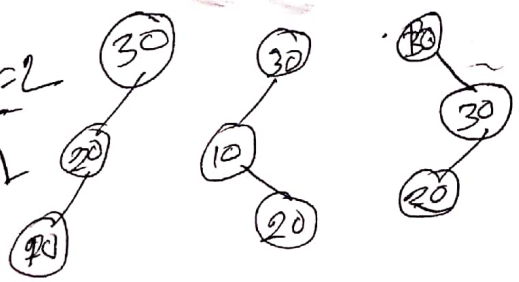
---

if n = 3 node given // we create 5 shape



max h = 2

min h = 1



AVL Tree height bbt

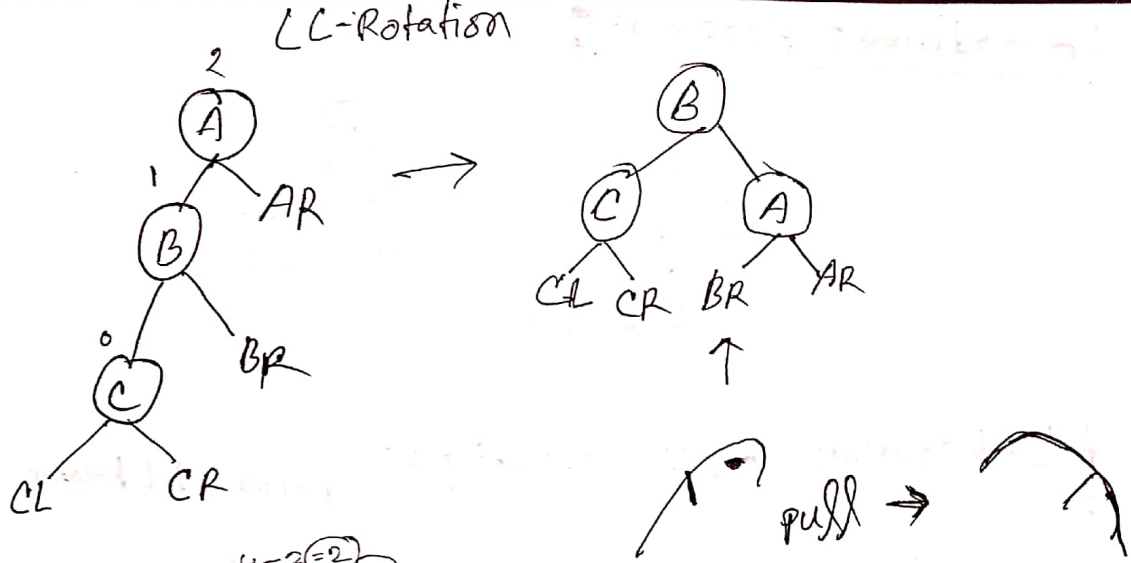
min height

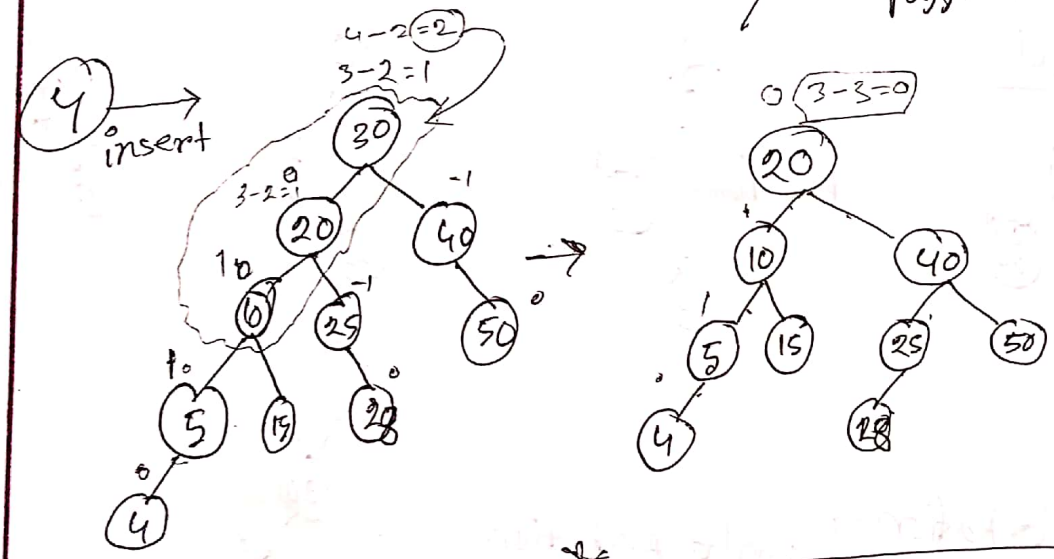
Lecture 3

AVL tree

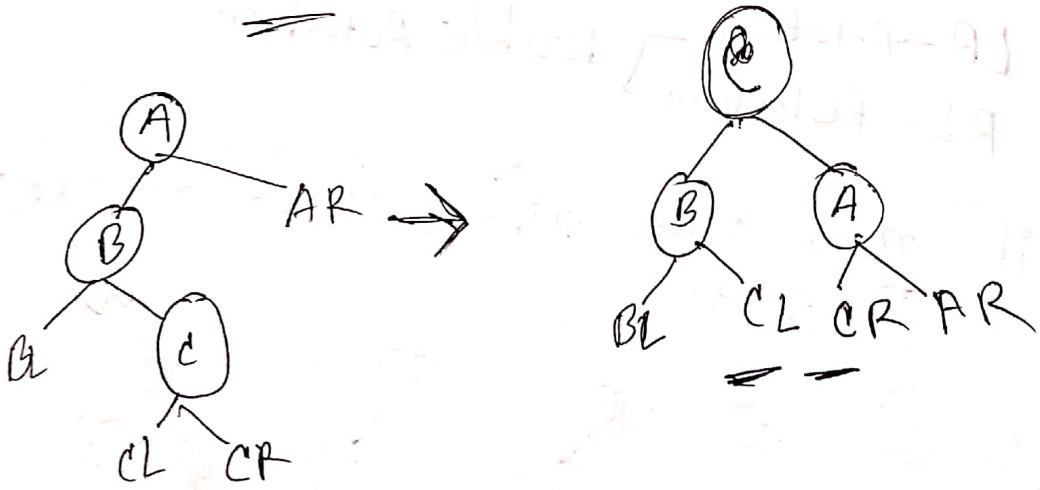
Formula of Rotation for insertion

LL-Rotation

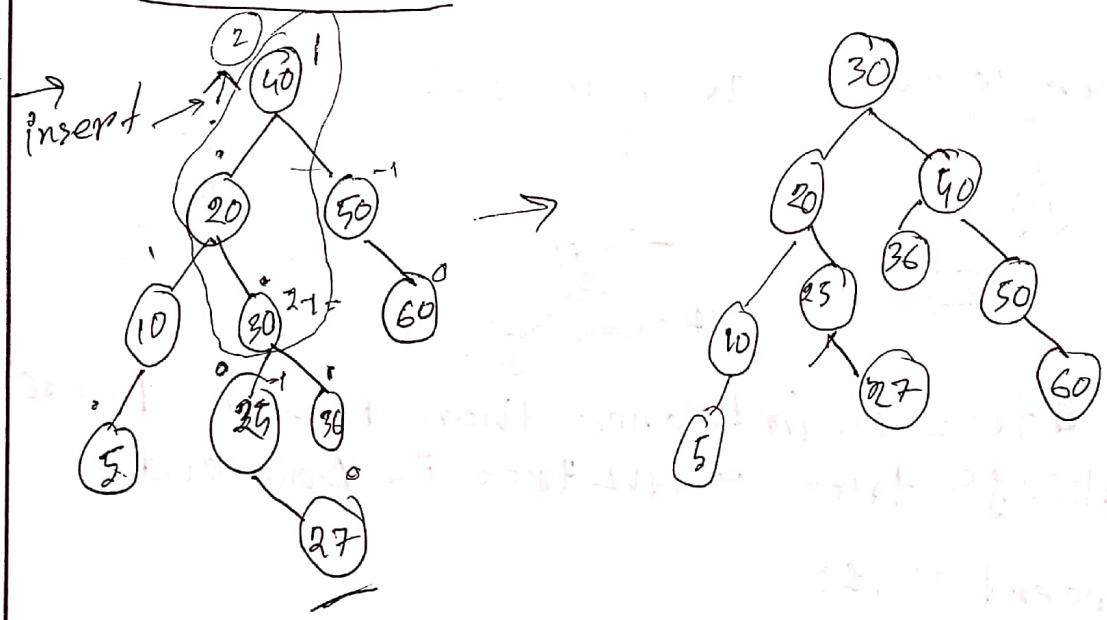




LR Rotation

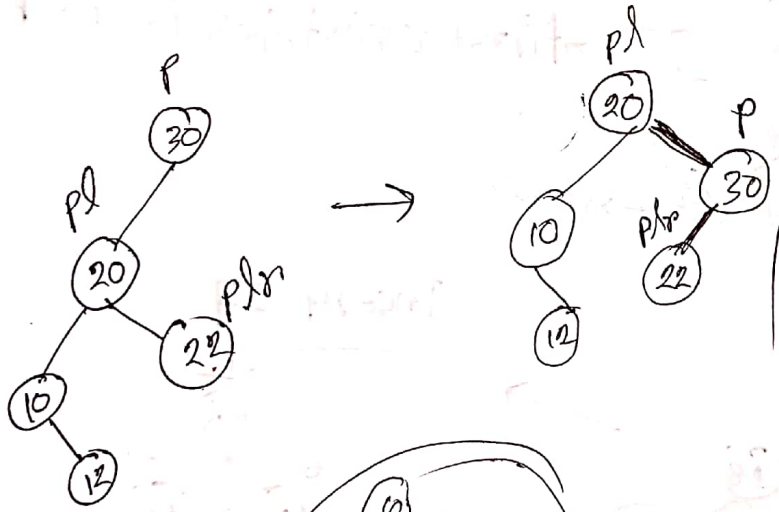


LR - Rotation



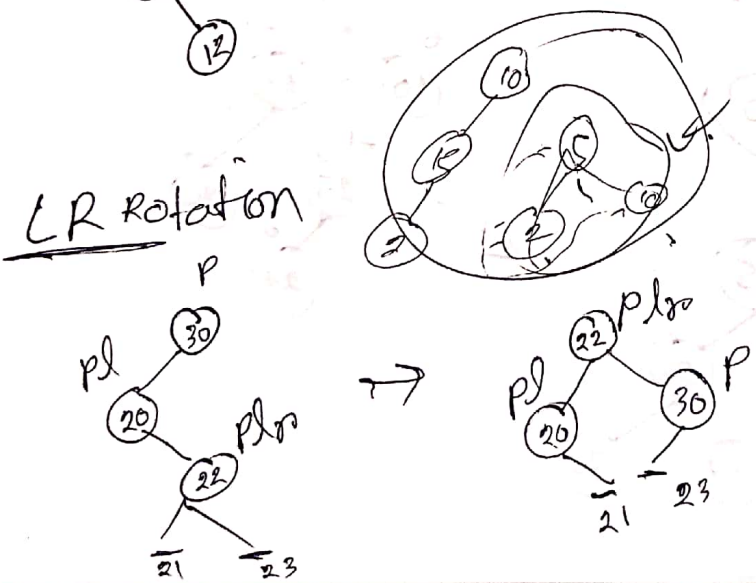
Lecture 4 code pdf

LL Rotation




```
pl = p->lchild ;
plr = pl->rchild;
pl->rchild = p;
p->lchild = plr;
```

LR Rotation



LR Rotation

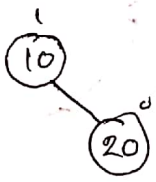
```
pl = p->lchild;
plr = pl->rchild
pl->lchild = plr->lchild;
p->rchild = plr->rchild.
plr->rchild = pl;
plr->lchild = p;
```

Create AVL tree

Class 6

key: 10, 20, 30, 25, 28, 27, 5

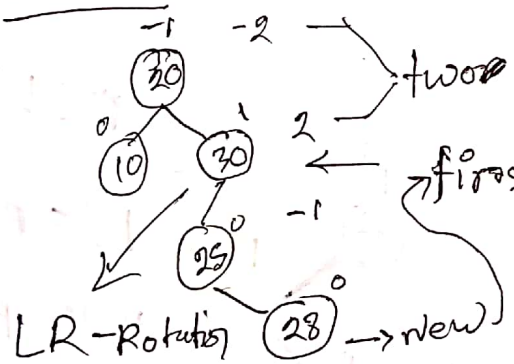
Inser 10, 20

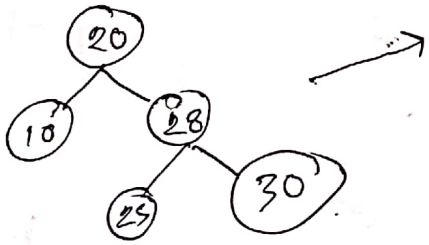


Inser 10, 20, 30

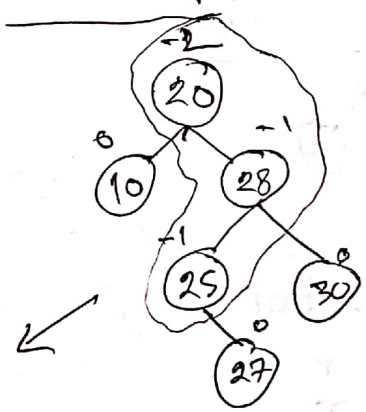
if a tree is imbalance than Rotation perform.
Always tree = AVL tree is Balanced.

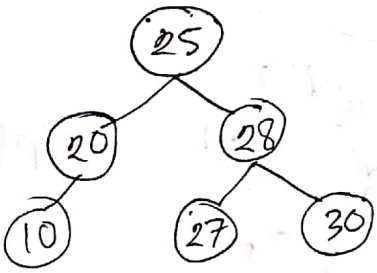
Insert 25, 28





Insert 27





Delection from AVL tree

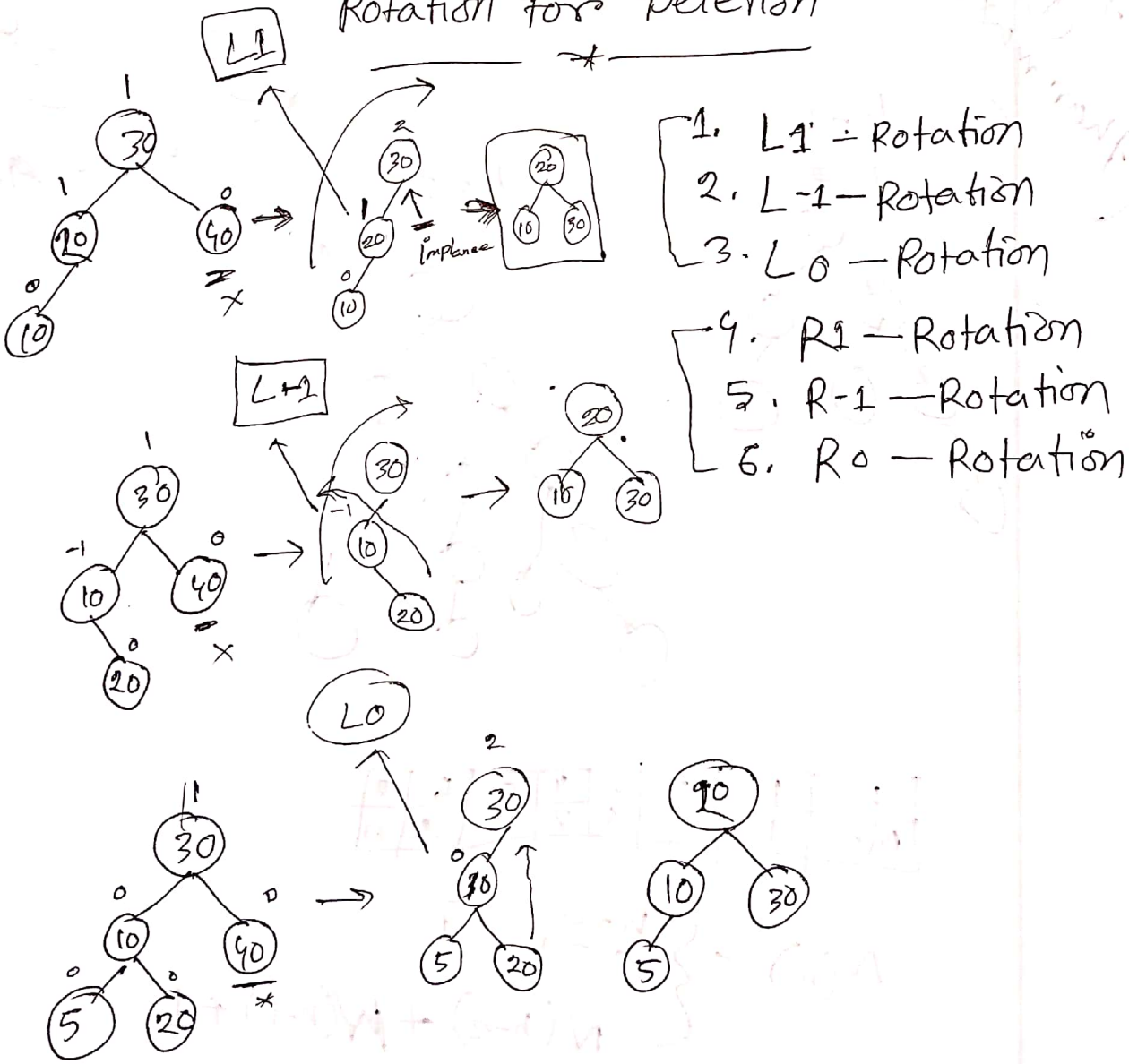
Lecture 7

Like → BST

if you want to delete key & first search the key
→ if it is find delete
→ who will take it's place
inorder predecer
inorder successin

AVL tree

Rotation for deletion



1. L1 - Rotation
2. L-1 — Rotation
3. LO — Rotation
4. R1 — Rotation
5. R-1 — Rotation
6. RO — Rotation

Height vs Nodes of AVL Tree

Lectures

If Height is given find

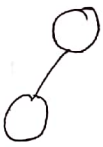
min Nodes =

max nodes $n = 2^{h+1} - 1$ → Binary tree formula

If n start from 1 than max $n = 2^h - 1$

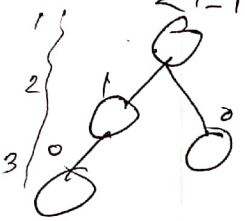
$h = 1$, $n = 1$

$h = 2$, $n = 2$ are required



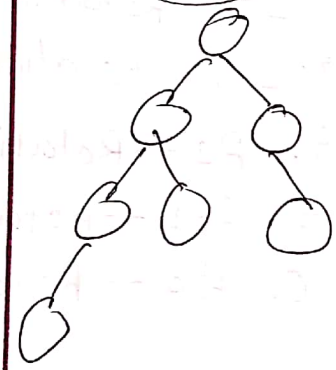
$h = 3$, $n = 3$

$2^3 - 1 = 1$



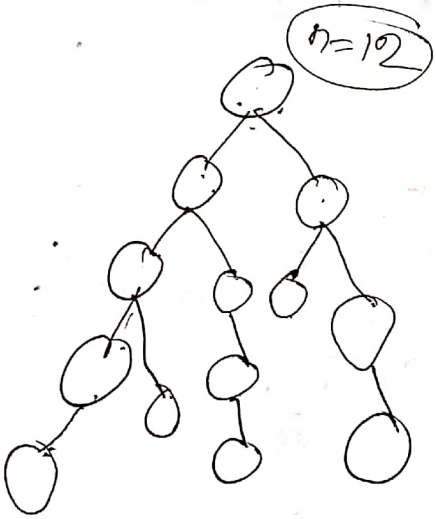
✗ it is not AVL tree

$h = 4$, $n = 7$



$h = 5$

$n = 12$



h	1	2	3	4	5	6	7
N(h)	1	2	4	7	12	20	33

$$N(h) = \begin{cases} h = 0 & 0 \\ h = 1 & 1 \\ N(h-2) + N(h-1) + 1 \end{cases}$$

if 'N' nodes are given find

min Height $h = \lceil \log_2 (n+1) \rceil$

max height h = Look in to the table