17/09

* Splay Tree :-

- BST, self-balanced / self Adjusting after every op.
- Operations :- { Insertion, Deletion, Search } → Splaying [Recent node is moved to the root]

depend { BST & Splaying

| Zig (Left
| Zag (Right

- Most frequently accessed element stay near the top. (move closer to root)

→ Splaying operations - (rotations) — 6.

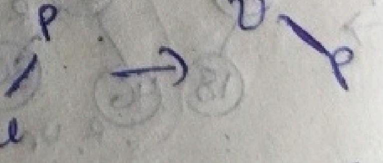
Zig	Zag
Zig - Zig	Zag - Zag
Zig - Zag	Zag - Zig

> Grand parent, Parent, node | Zig (Left position) Zag (Right position)

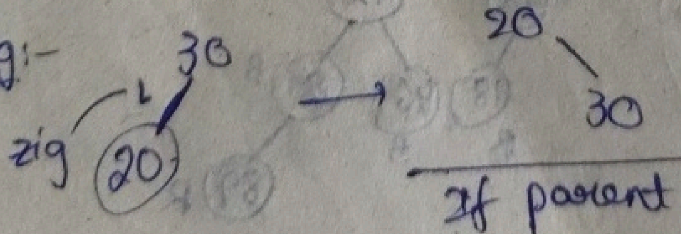
→ splaying :- moving a node to the root using tree rotations.
  ↳ Rotations ensures recently accessed elements are near the root.

Bottom-up
→ Rotations :- | insert & delete then splay

1) Zig :- (Single rotation) (Left position)
- Right rotation (w.r.t P)
- no grand parent (single rotation over u & P)

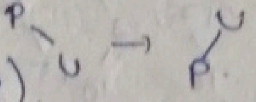


Eg:-

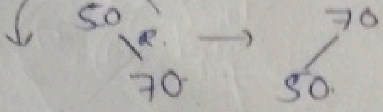


(Zig)
If parent is root & node is left
→ Right rotation

2) ~~Zig~~ Zag (Single rotation) (Right position)
- No grandparent
- Left Rotation (w.r.t p)

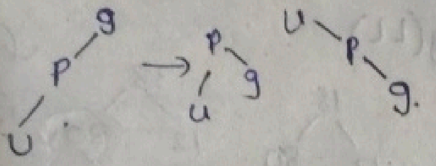


Rotation should start from P/gP node
→ sketch is based on new node

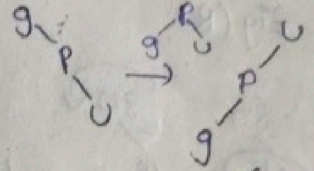


If parent is root & node is right child → left rotation

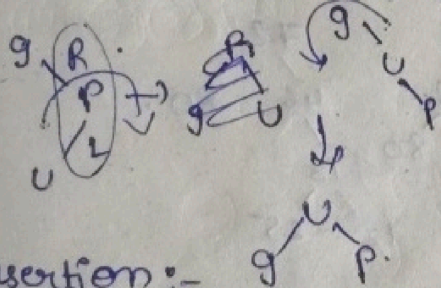
3) Zig-Zig :- (right-right rotat)
- Double rotn
- (left-left) Position (w.r.t g & p)



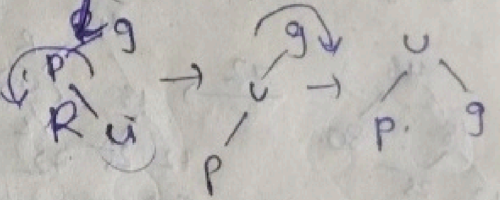
4) Zag-Zag :- (left-left Rotation)
- Double rotn
- (Right-Right) position (w.r.t g & p)



4) Zig-Zag :- (left-Right position)
- Double
- Right-left Rotation (w.r.t p & g)

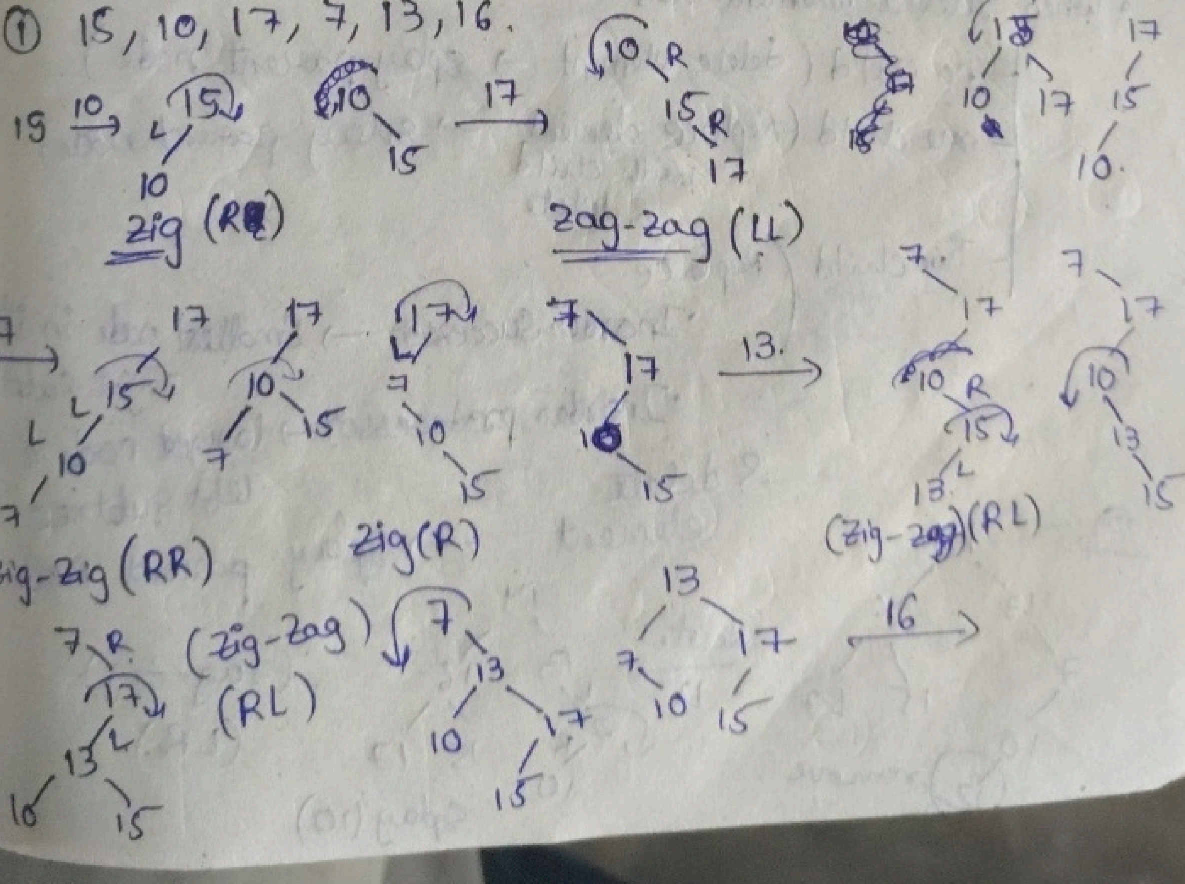


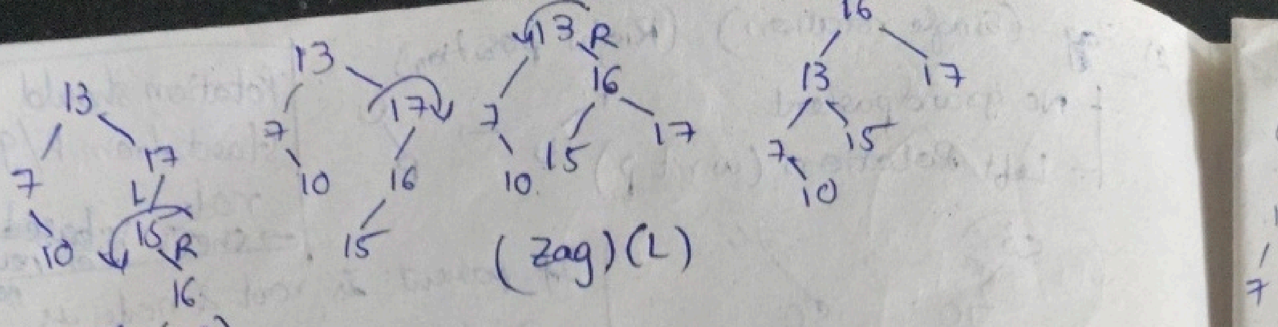
5) Zag-Zig :- (Right-left position) (w.r.t P&g)
- Double
- (left-Right rotation)



→ Insertion :-
Eg:- Insert acc. to BST & splay it (insert node as root node using rotations)

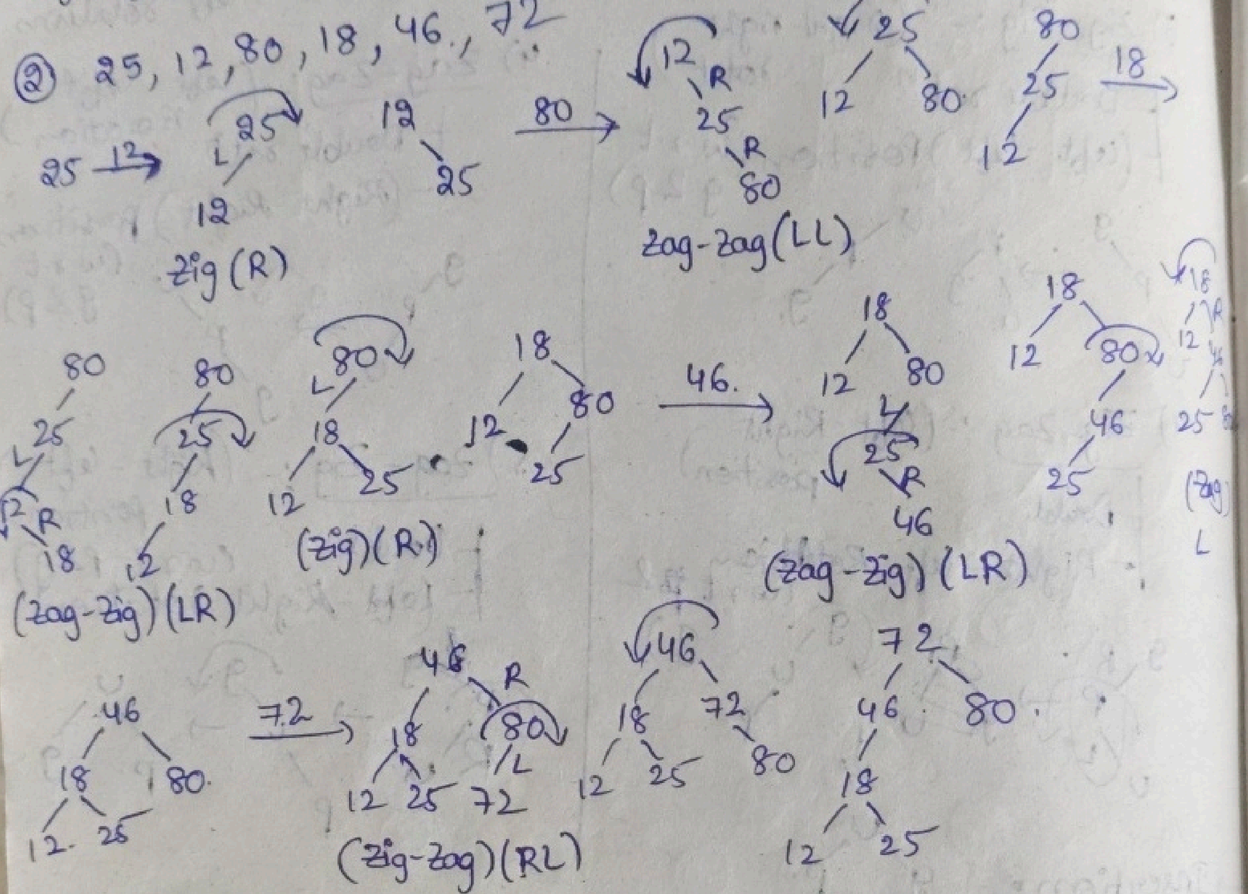
① 15, 10, 17, 7, 13, 16.





Zag-Zig (LR)

② 25, 12, 80, 18, 46, 72

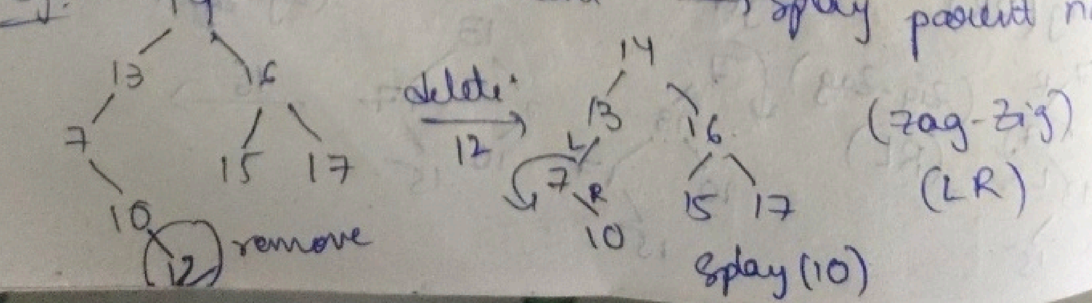


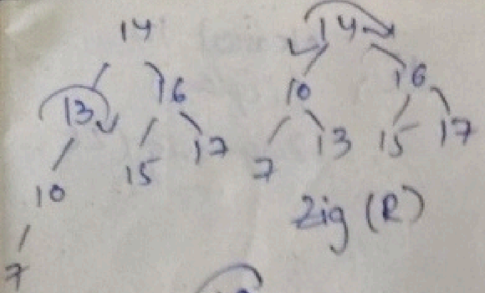
→ Deletion :- (In BST, we only able to delete leaf node)

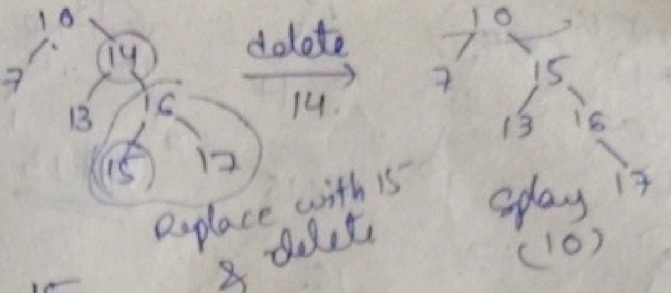
- when deleted element has
  - no child (delete element → splay parent node)
  - one child (Replace element with child & delete → splay parent node)
  - Two child (Replace
    - Inorder Successor → Smallest node in its right subtree
    - Inorder predecessor → largest node in the left subtree

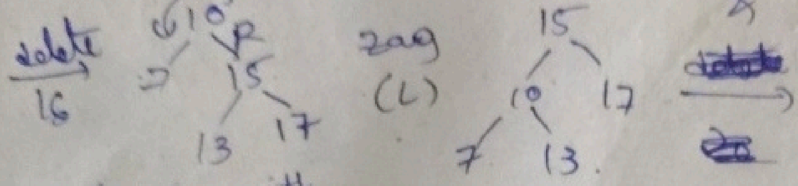
    & delete element → splay parent node)

Eg :-









Replace 16 with one child & delete & splay 15

* Top-down:-
  - Tree empty (insert node as node)
  - otherwise
    - if key exists (do nothing)
    - if not exist (stop at closest node & splay it) → indirectly parent to new node

Slaying before & then insert & delete

→ Create a new node (the element need to be inserted).

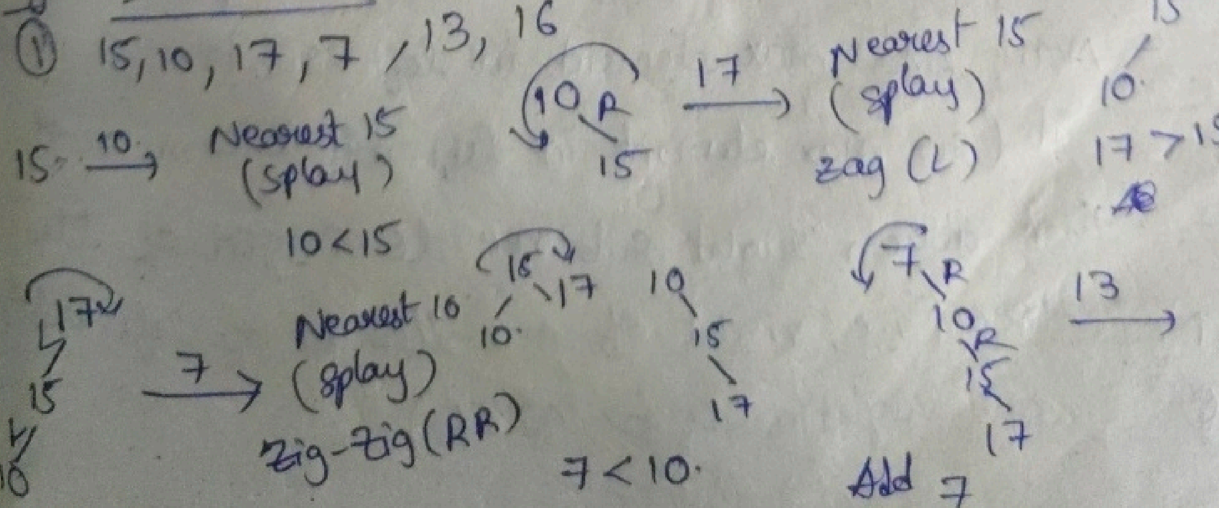
① If element < Root
  - element - left → Root - left tree
  - element - right → Root
  - Root - left → NULL

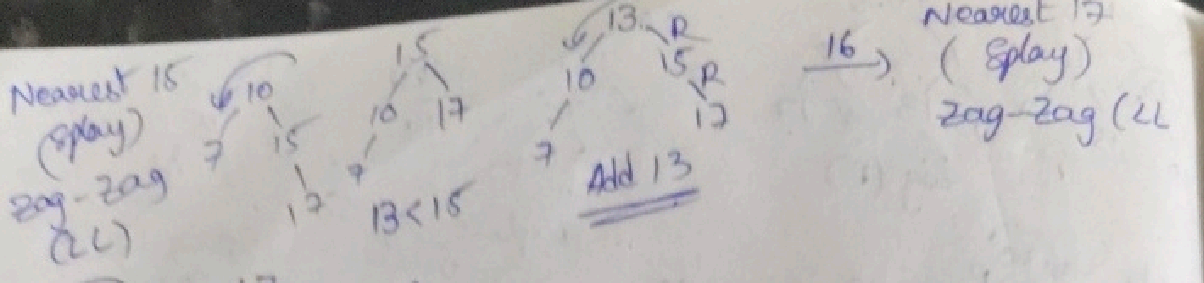
② If element > Root
  - element - right → Root - ~~left~~ Right tree
  - element - left → Root
  - Root - Right → NULL

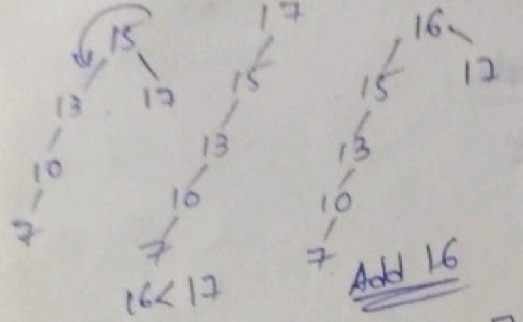
stop at nearest & splay it using Rotations & use Combination

Eg:- Insertion:-

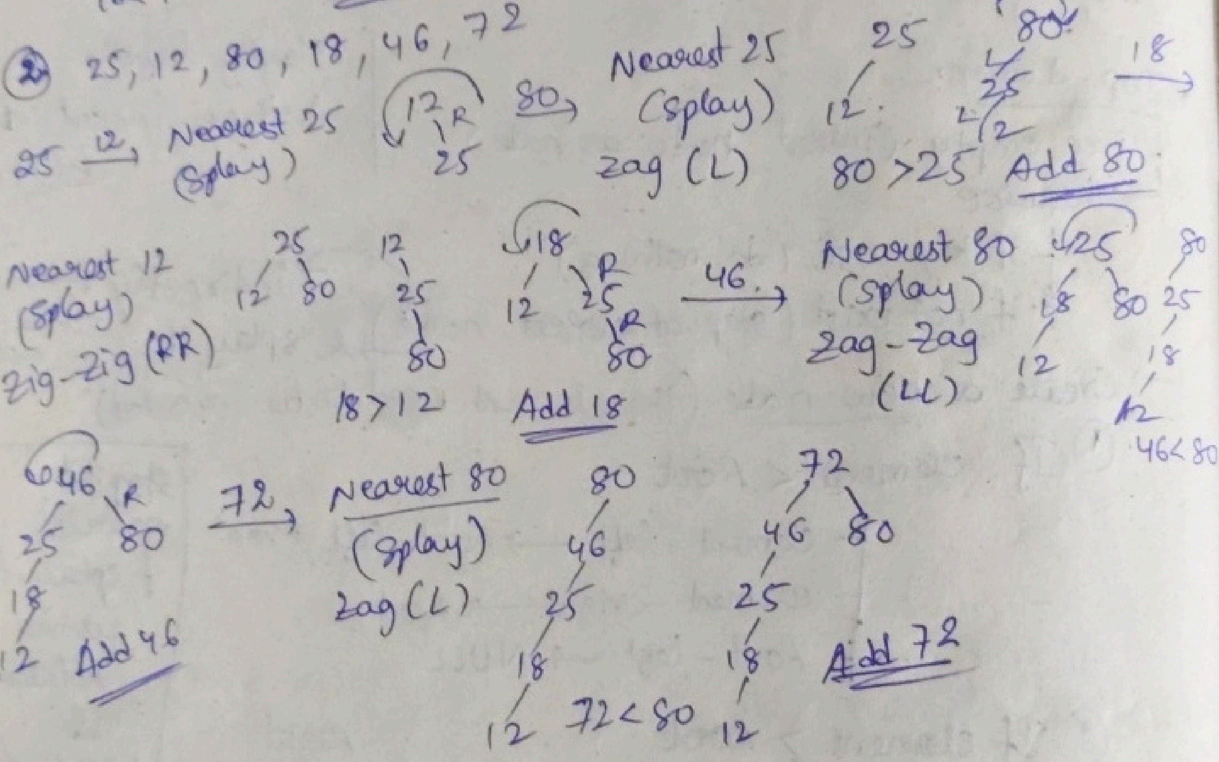
① 15, 10, 17, 7, 13, 16




Nearest 15 (splay) Zag-Zag (LL)
13<15
Add 13
16
Nearest 17 (Splay) Zag-Zag (LL



② 25, 12, 80, 18, 46, 72

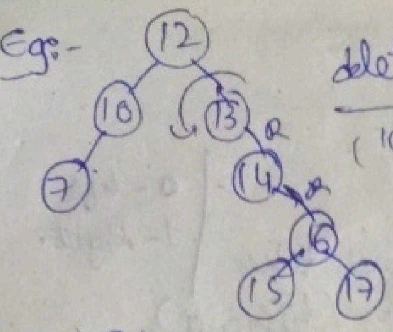

25
12
Nearest 25 (Splay)
80
Nearest 25 (splay) Zag (L)
80>25
Add 80
18
Nearest 12 (splay) Zig-Zig (RR)
18>12
Add 18
46.
Nearest 80 (splay) Zag-Zag (LL)
46<80
Add 46
72
Nearest 80 (splay) Zag (L)
72<80
Add 72

→ **Deletion** :-

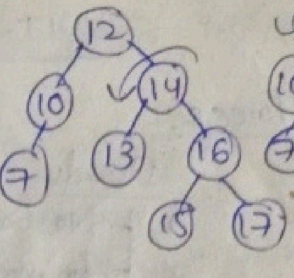
- Splay the element need to be deleted (then it become root)
- then delete the root (the req. element) then we get 2 sub trees (left-subtree & Right subtree)
- Now find max element in left-subtree & splay it (so max element in left will become root)
- then attach right subtree to right of the tree (i.e, right of max-element of left subtree as right will be NULL).

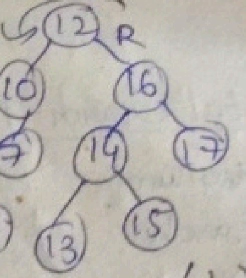
JOIN operation

Eg:-

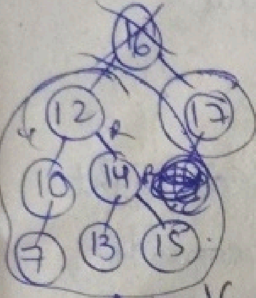


$\xrightarrow[(16)]{\text{delete}}$ splay 16 Zag-Zag (LL)





(Zag)(L)

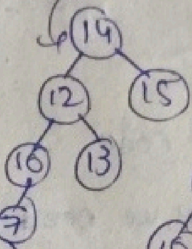


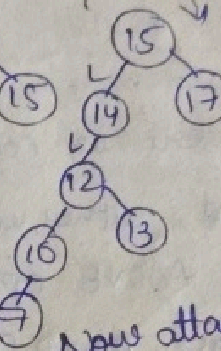
delete 16

Max of Left-subtree (15)

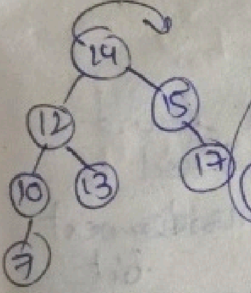
splay 15

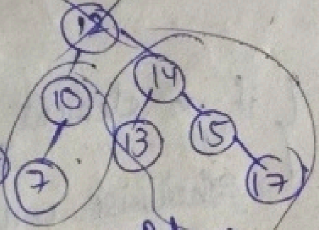
Zag-Zag (LL)





$\xrightarrow[12.]{\text{delete}}$ splay 12. Zig-Zig. (RR)



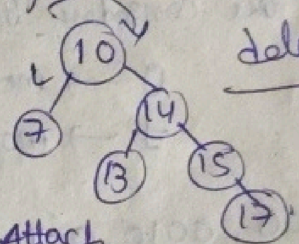


delete 12

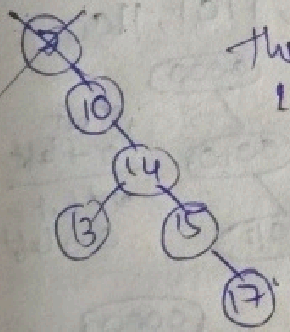
Max of Left-ST 10

Splay 10

Now attach Right subtree



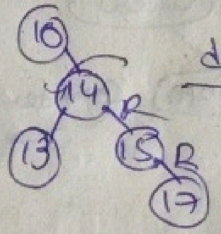
$\xrightarrow[7]{\text{delete}}$ splay 7 zig (R)



There is no LST to find max

So attach Right to NULL

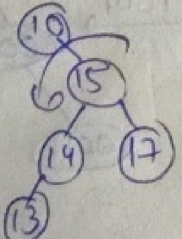
Attach Right subtree

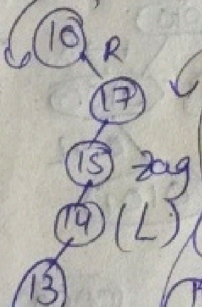


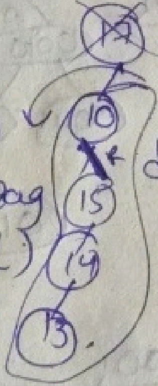
$\xrightarrow[17]{\text{delete}}$

if there is no RST, then find max of LST & make it as root

Splay 17 Zag-Zag (LL)







delete 17

max of LST (15) splay zag (L)