Splay Trees

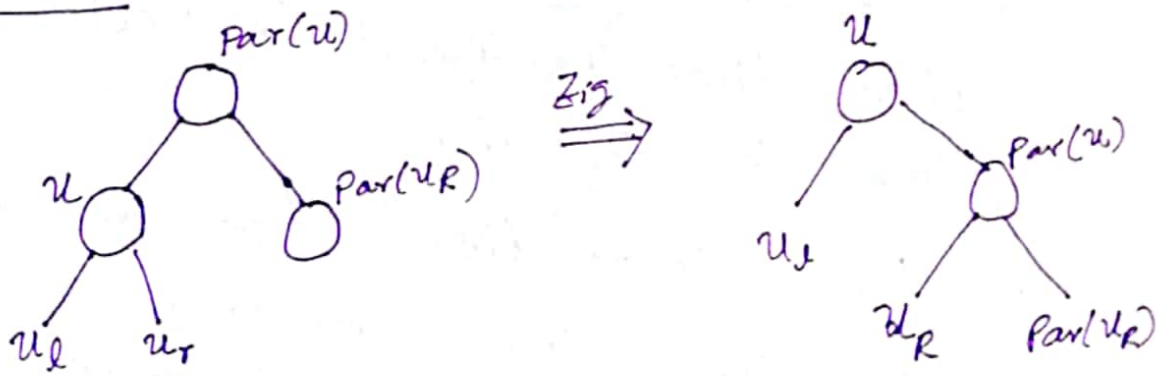
- Splay tree is similar to BST except at the end of each operation insert, delete or search, splaying is done.
- Splaying means pushing the recently accessed node towards root using rotations.
- This is useful when a particular node is accessed frequently.
- In BST, if a node is accessed frequently (n nodes) then time complexity in worst case is O(mn).
- In splay tree, the amortized time complexity (based on sequence of operations) is O(m log n), where m is sequence of operations.

Splay Rotations

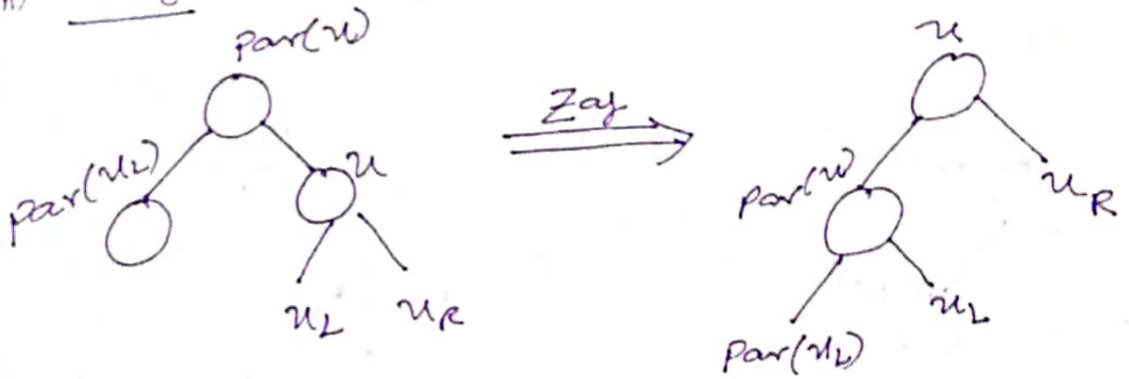
- The accessed node is moved up. (Bottom up)
- Moving can be two levels up
  Called,
  i) Zig - Zig
  ii) Zig - Zag
  iii) Zag - Zig
  iv) Zag - Zag

Can be single level
  (i) Zig
  (ii) Zag.

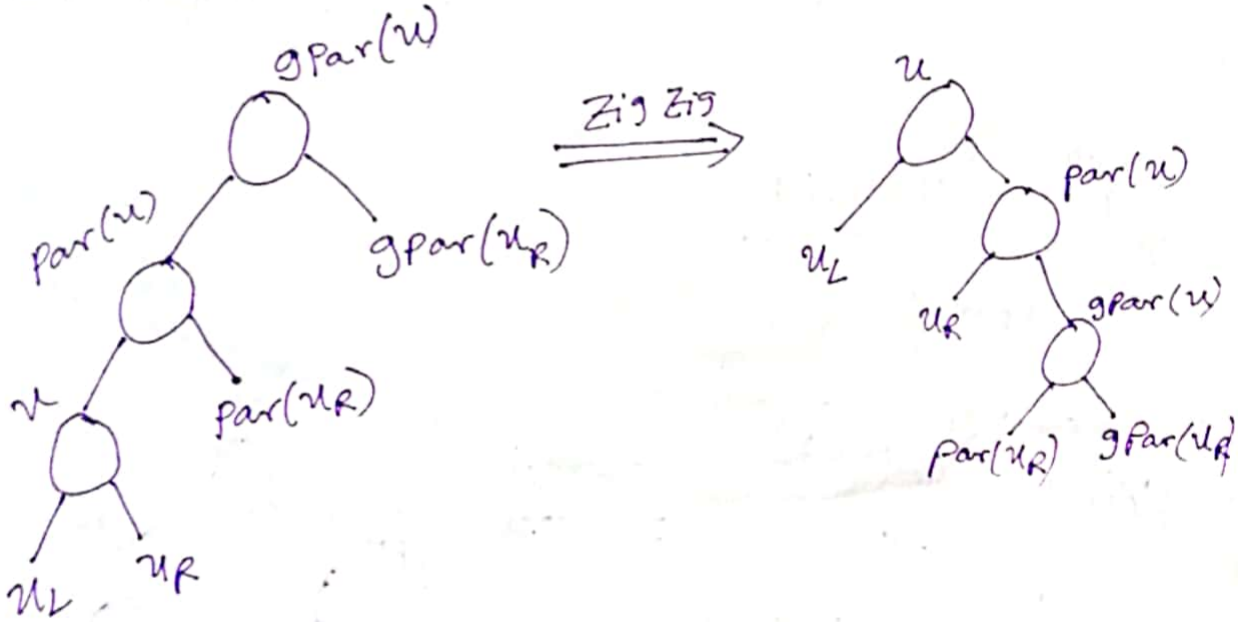
(i) Zig



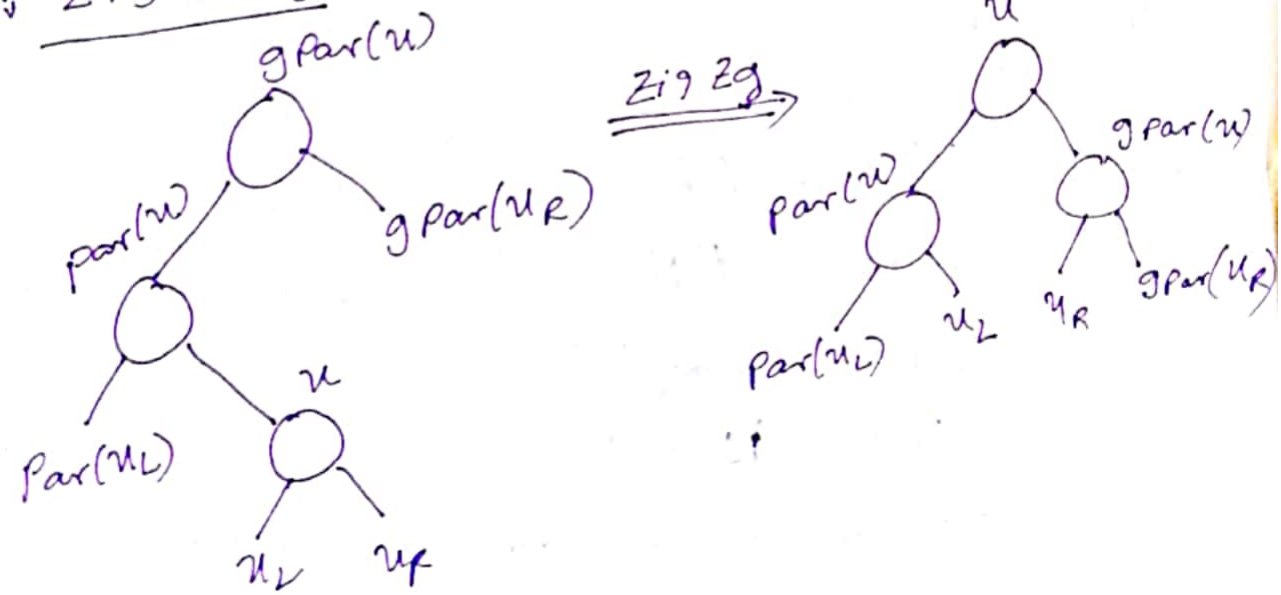
ii) Zag



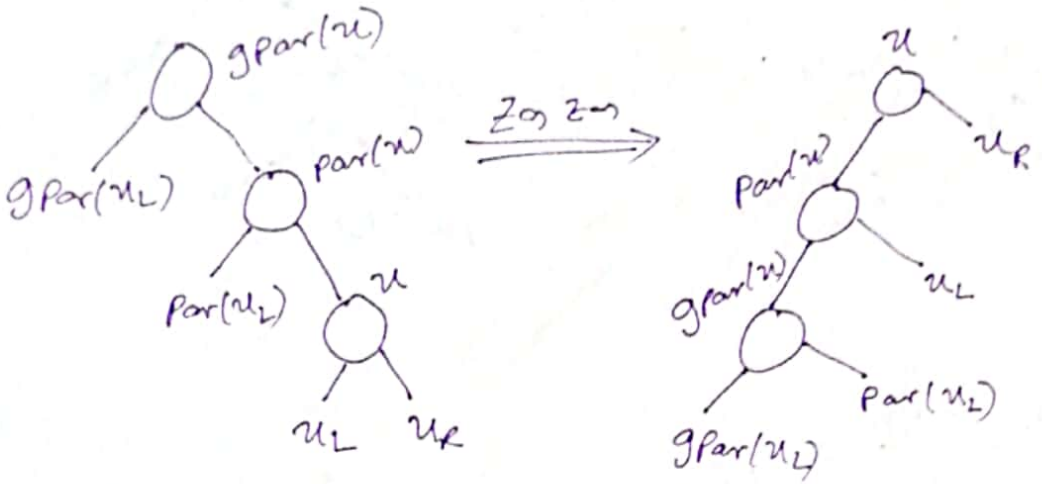
iii) Zig - Zig



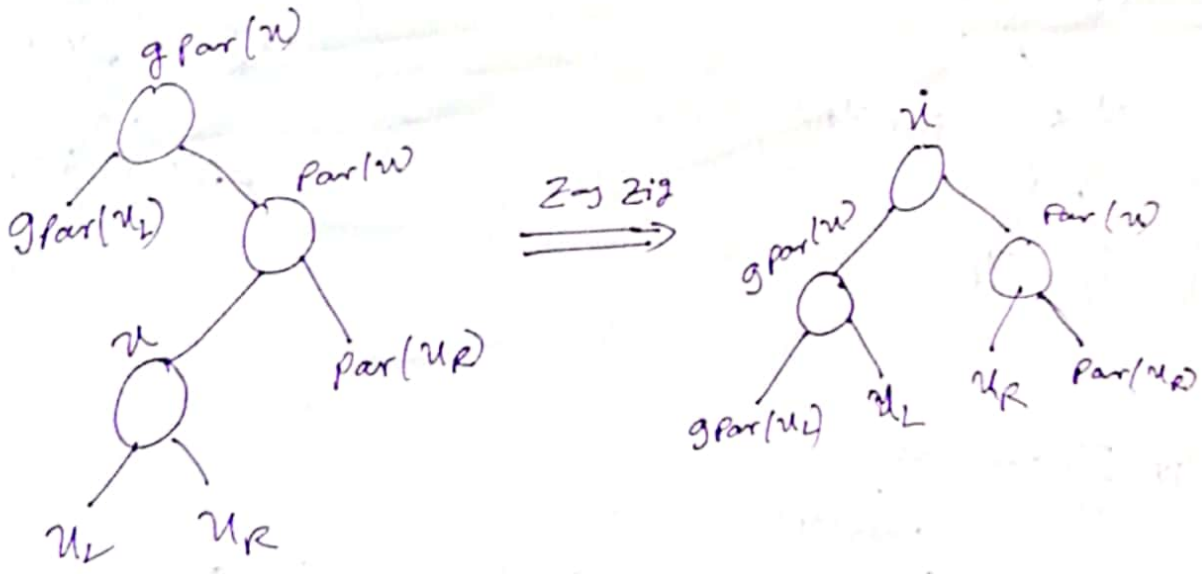
iv) Zig - Zag



v) Zag Zag



vi) Zag Zig



Zig - LL

Zag - RR

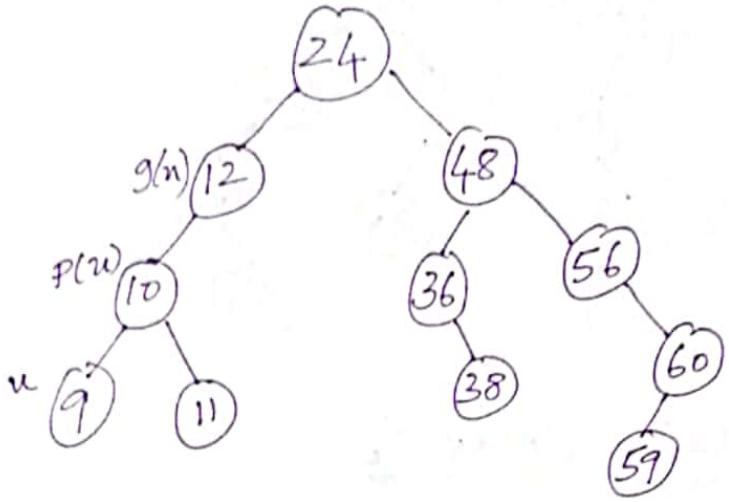
Zig Zag - LR

Zag Zig - RL

Zig Zig & Zag Zag are different.

Example

Consider following BST

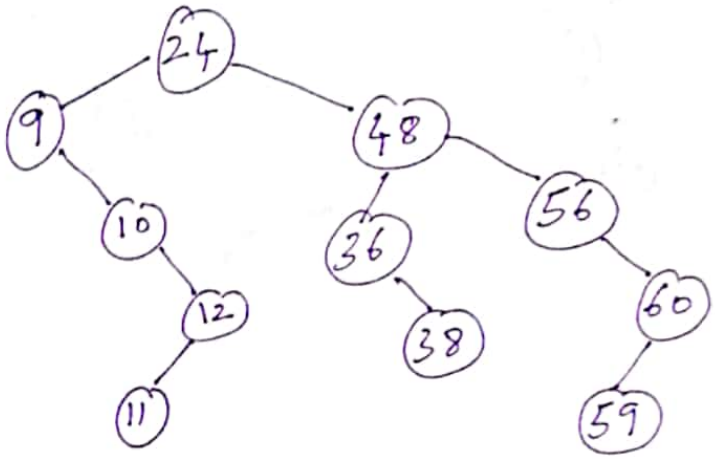


Splay the node 9.

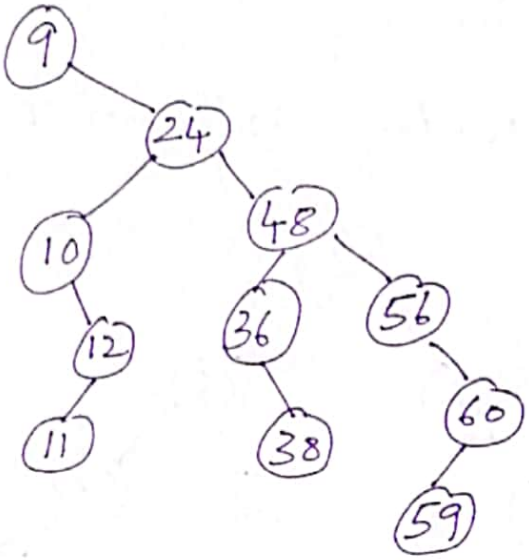
We have to push 9 to root.
Sequence of splay rotations are,

Zig-Zig , Zig.
(ie) two steps followed by Single)

Zig-Zig

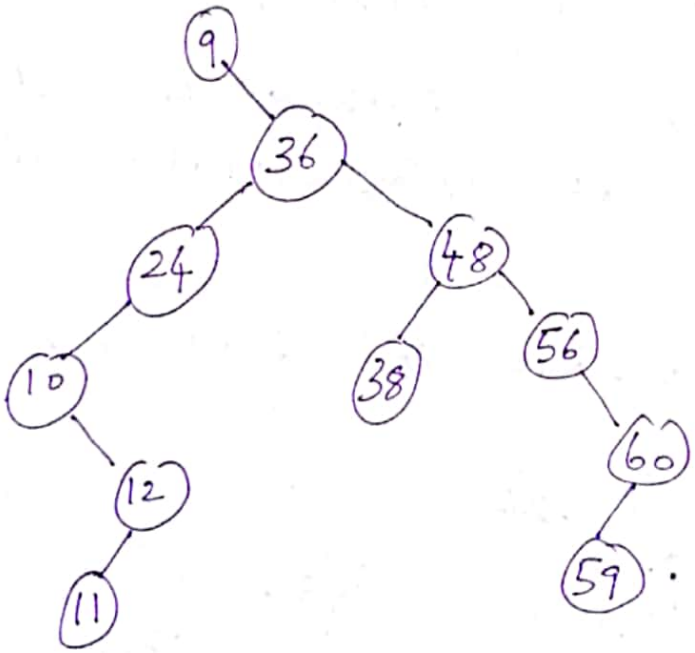


Zig

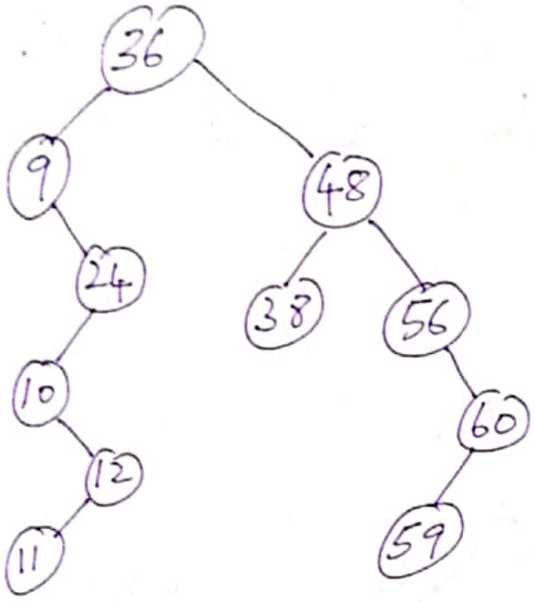


Now again splay 36

(i) Zag Zig



(ii) Zag

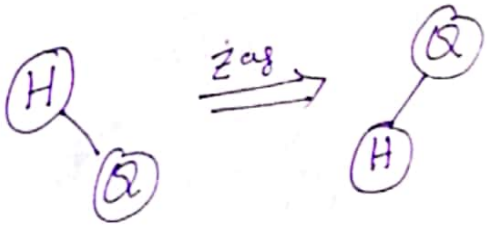


(2) Build a splay tree by inserting following in sequence

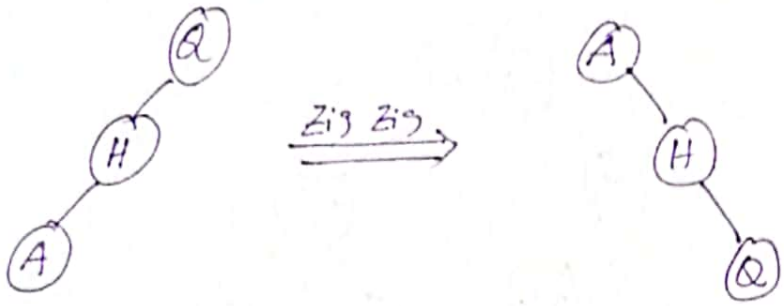
H, Q, A, N, P, O

i) Insert H

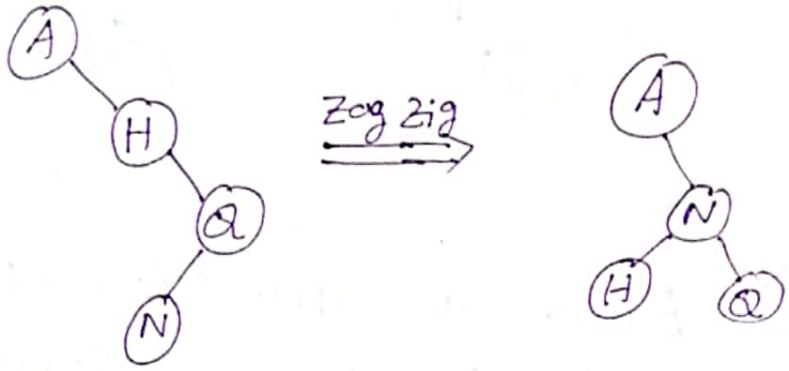
ii) Insert Q

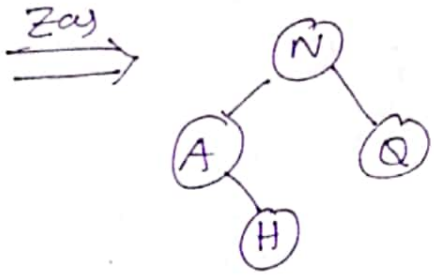


iii) Insert A

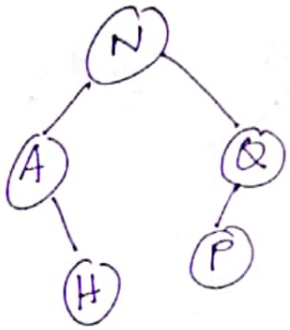


iv) Insert N

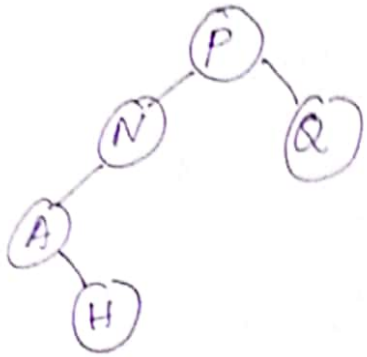




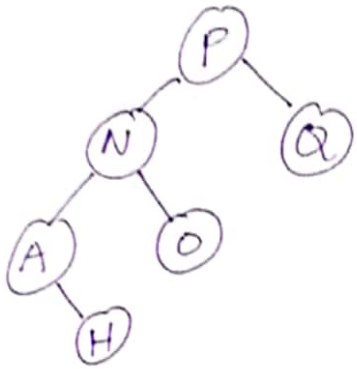
v) Insert P



Zag Zig



vi) Insert O



Zig Zag