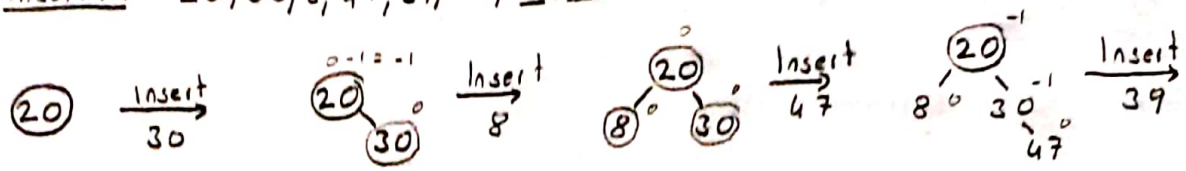
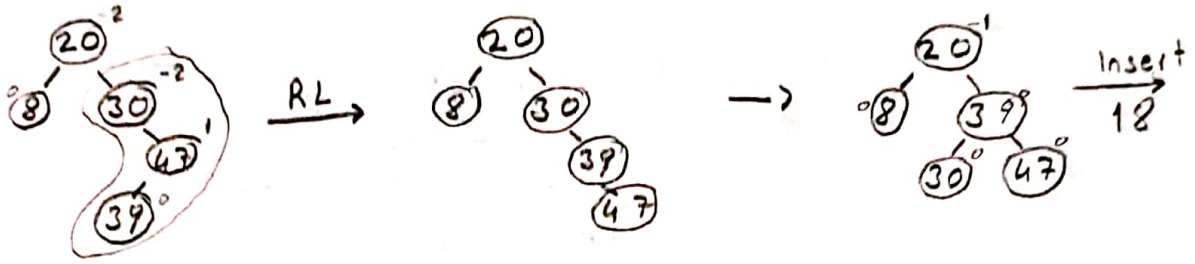
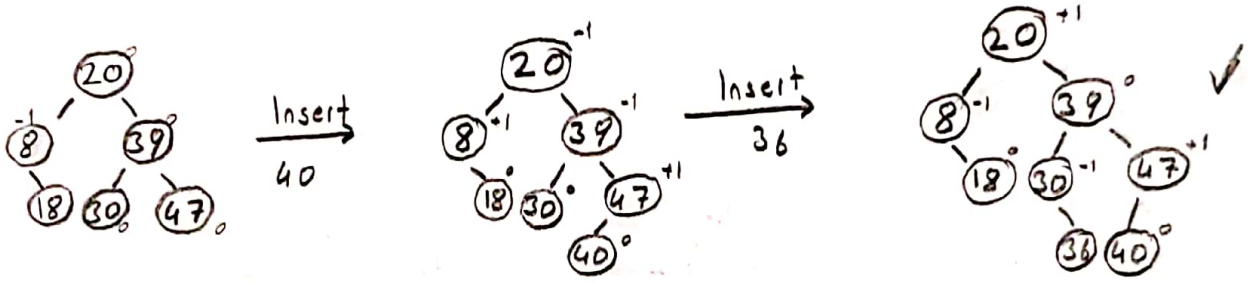
Baran Hasan BOZDUMAN 171044036

→ AVL Tree

→ Insertion 20, 30, 8, 47, 39, 18, 40, 36

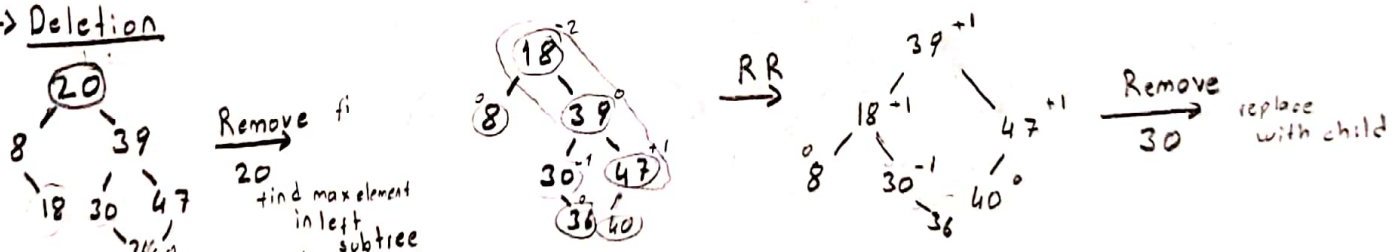


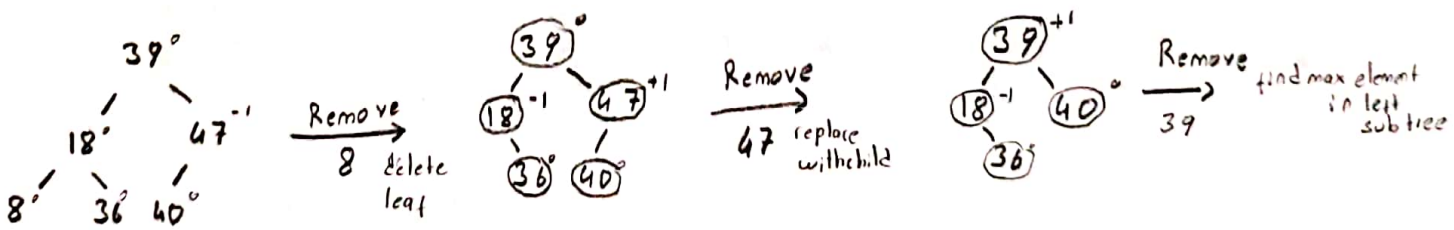


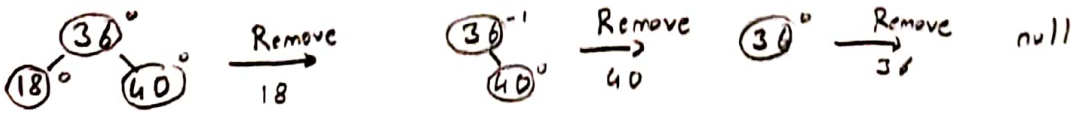


When we're inserting an element to AVL tree we check for each step and decide whether the tree is balanced or not. (if difference between right and left height is {-1, 0, +1} for each node the tree is balanced (|balancefactor| = |height left subtree| - |height right subtree| ≤ 1) otherwise it needs some rotation operations

→ Deletion







When we're removing an element from AVL tree. The operations like BST delete operations but after each deletion operation we check the tree is balanced or not if it is unbalanced we make some rotation operations to obtain a balanced tree

→ Red-Black Tree

Rules

→ Every node is red or black.

→ Root is always black.

→ New insertions are always red.

→ Every path from root-leaf has same number of black nodes.

→ No path can have two consecutive Red nodes.

→ Nulls are black.

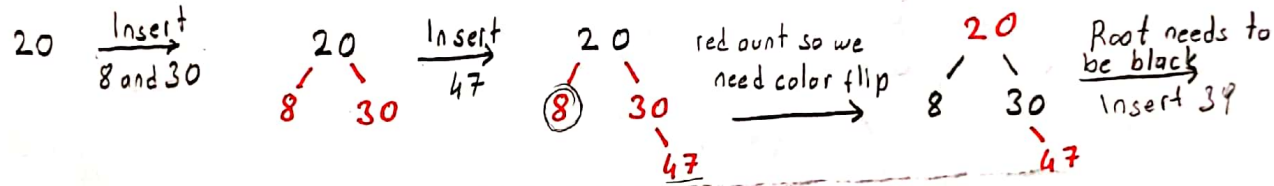
We should check is it providing the roles or not after each operation.

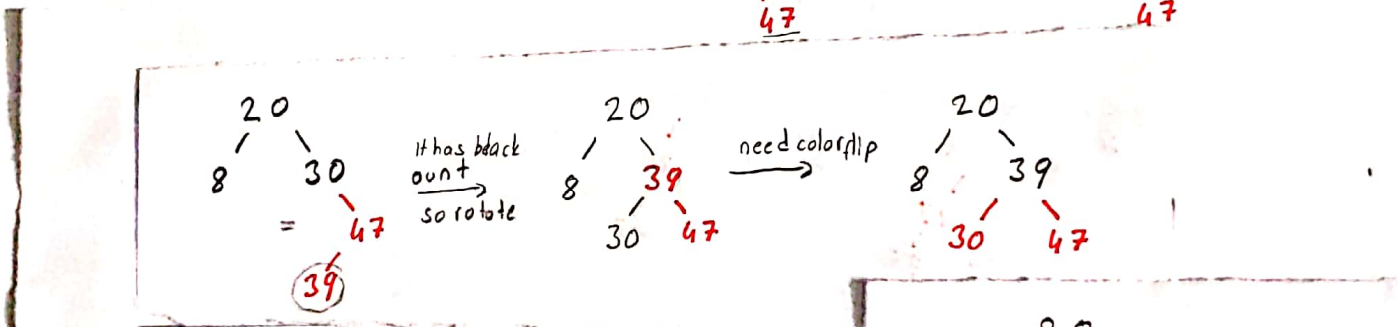
To fix tree

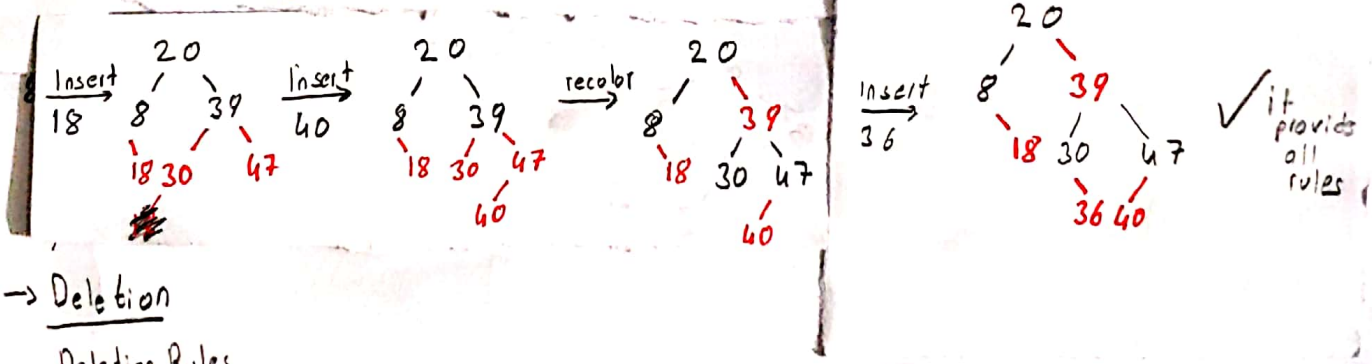
Black Aunt → Rotate

Red Aunt → Color flip

→ Insertion



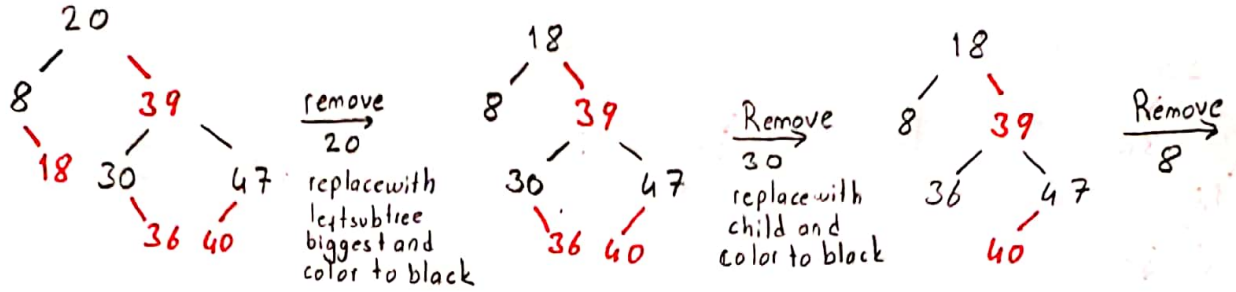


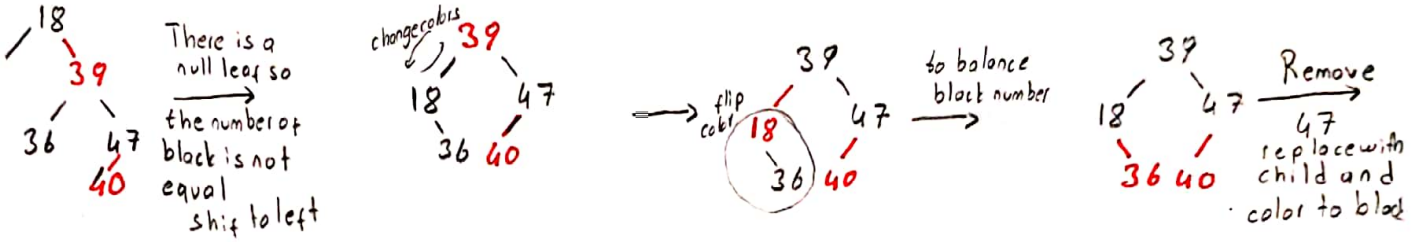


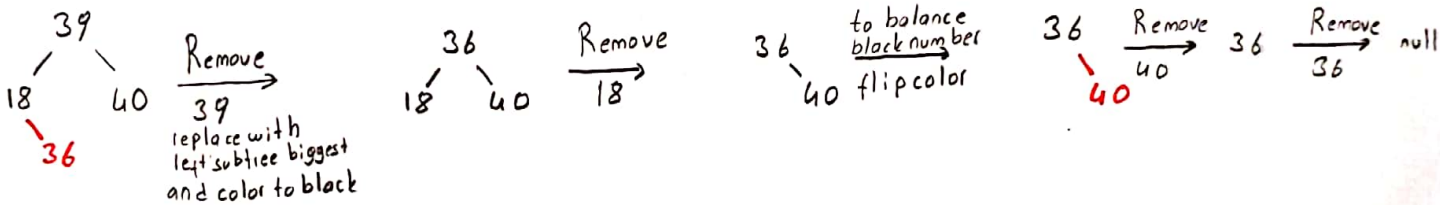
→ Deletion

Deletion Rules

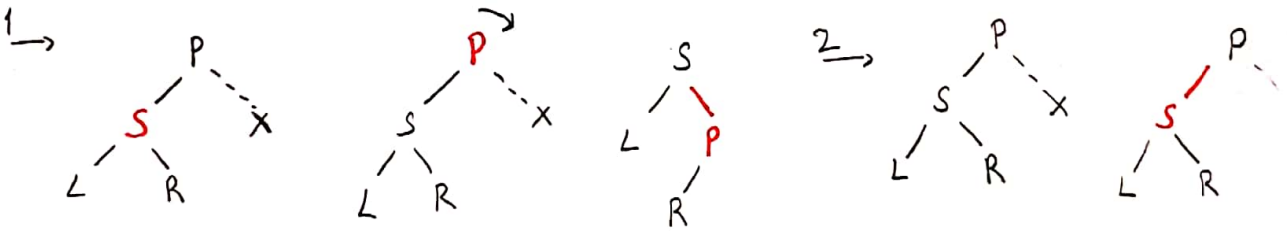
- If the node we deleted is red and its replacement is red or null, we are done
- If the node we deleted is red and its replacement is black, color the replacement red and proceed to the appropriate case.
- If the node we deleted u black and its replacement is red color the replacement black.
- If the node we deleted is black and its replacement is null or black, proceed to the appropriate case

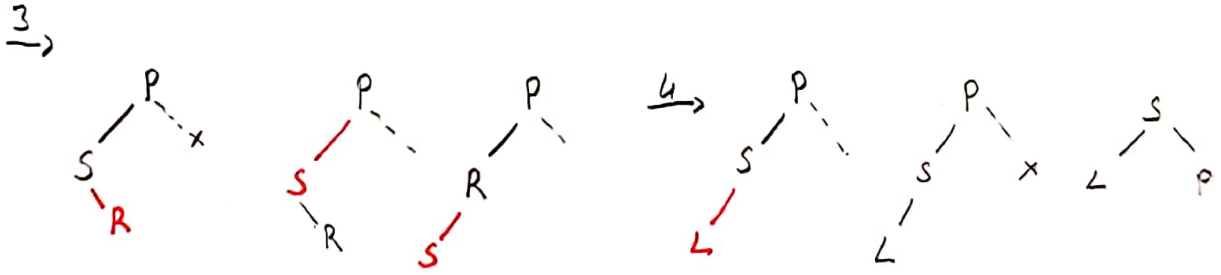






Cases

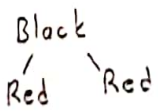




After color flip:

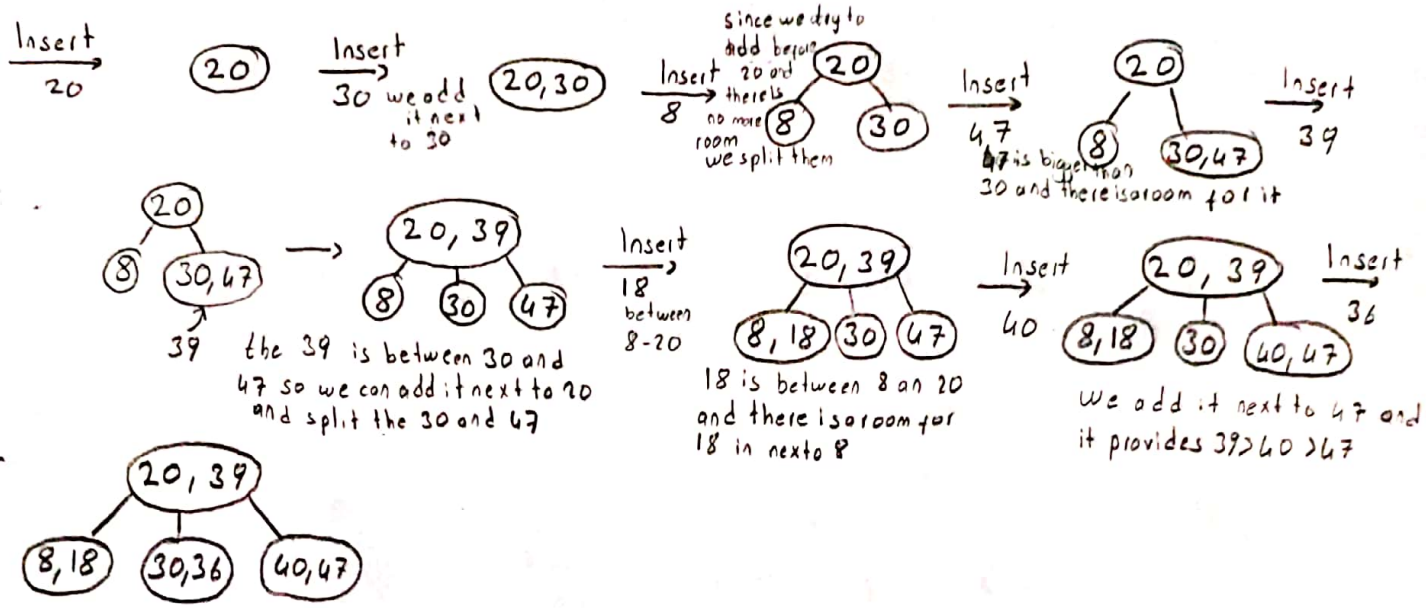
↳ you need to change red to black if its the root

After Rotation



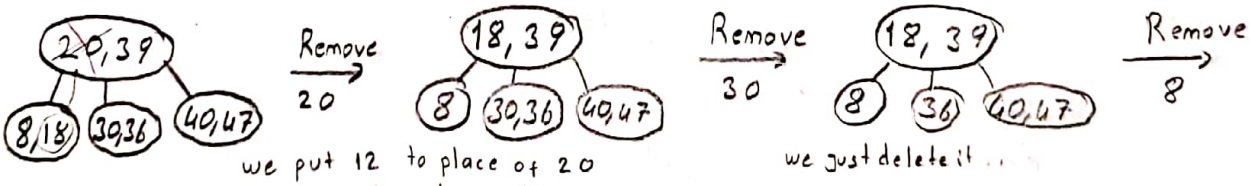
→ 2-3 Tree

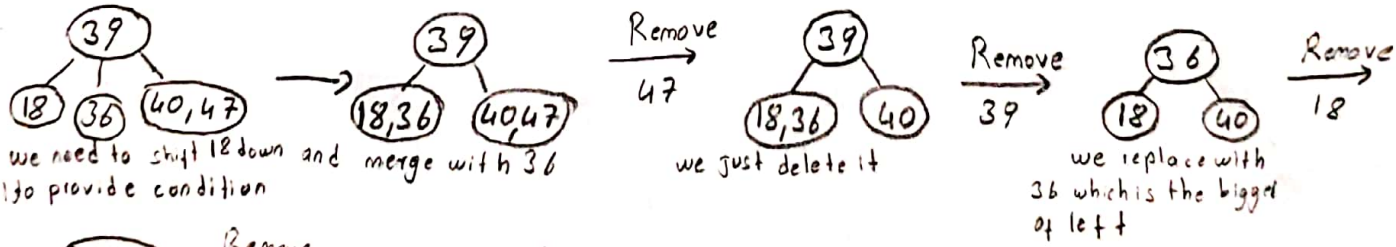
→ Insertion 20-30-8-47-39-18-40-36

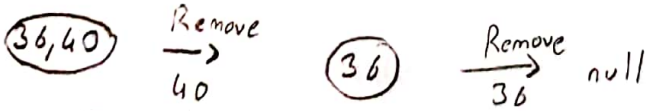


we decide the numbers and where they suppose to be if the given number are between them of a nodes number we add it into a node which is connected to middle.

→ Deletion

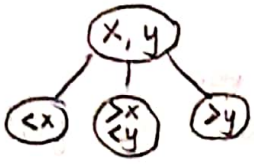






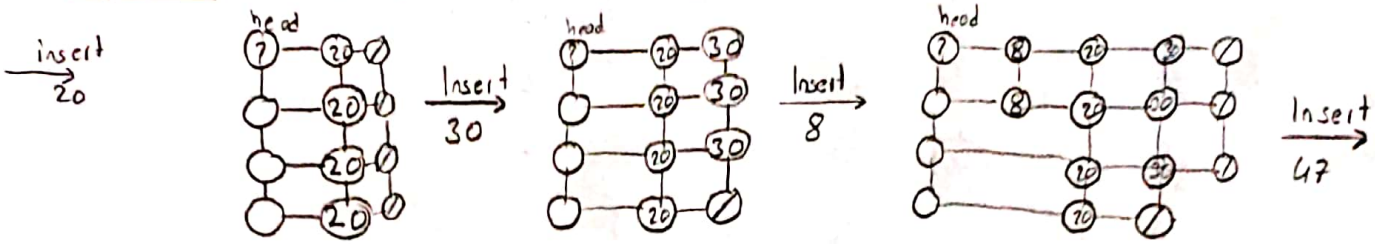
After remove 18 we need to merge them

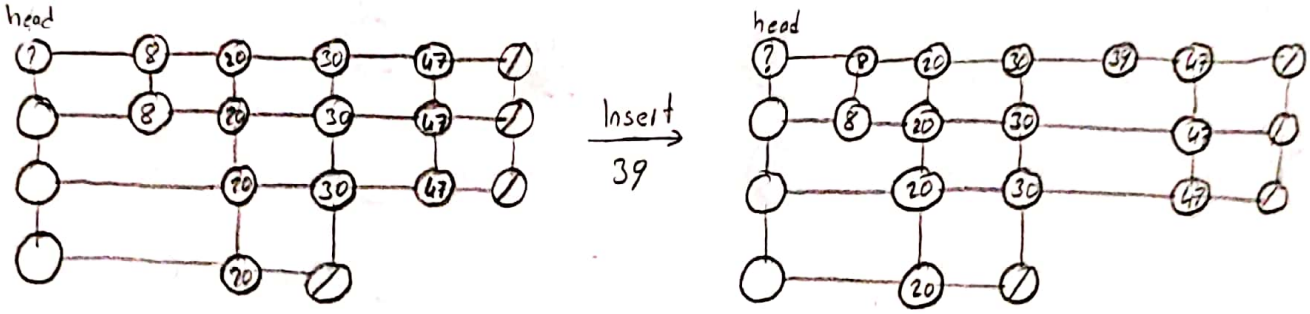
The tree should provide below situation after we make any changes

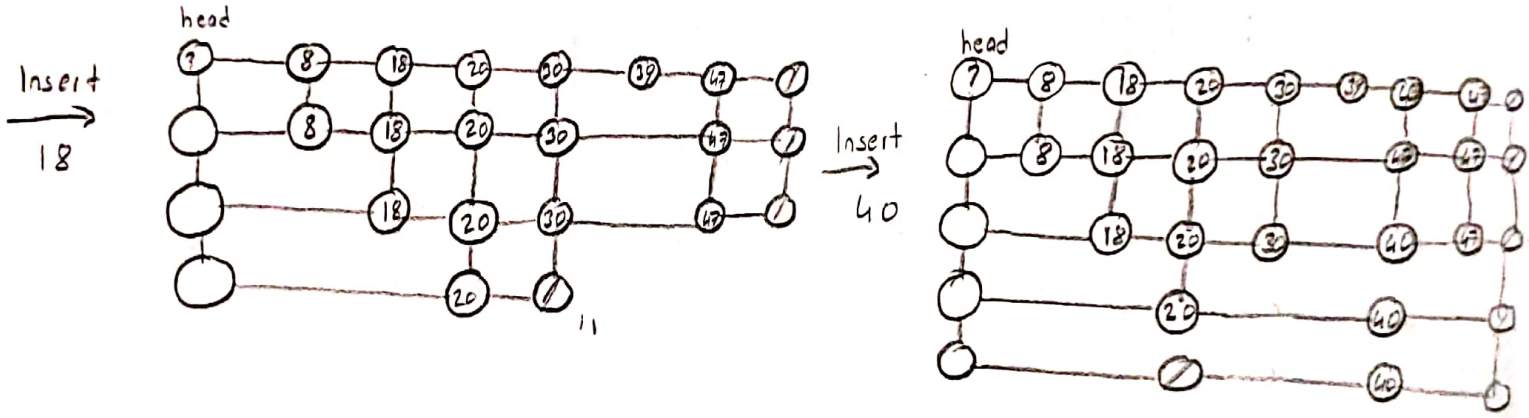


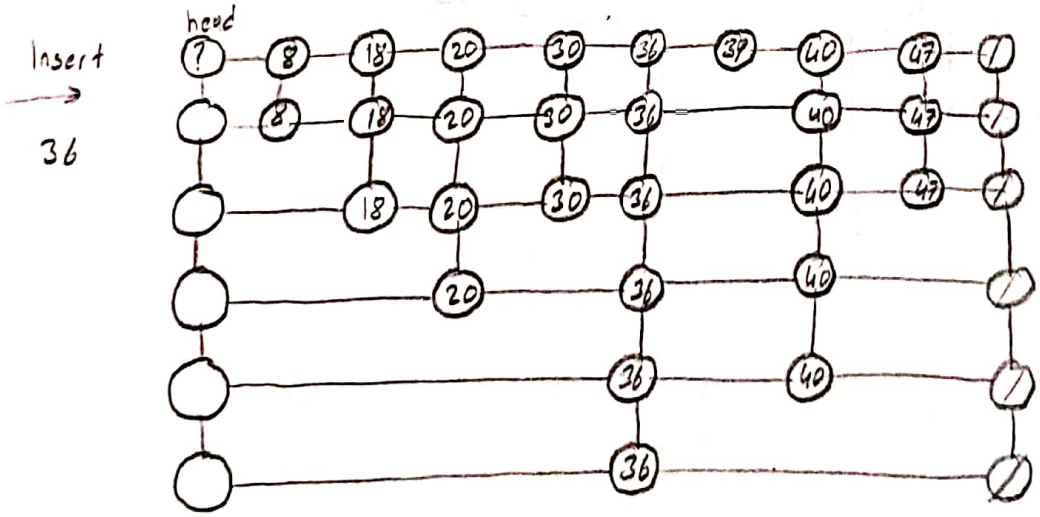
→ Skip List

→ Insertion 20-30-8-47-39-18-40-36





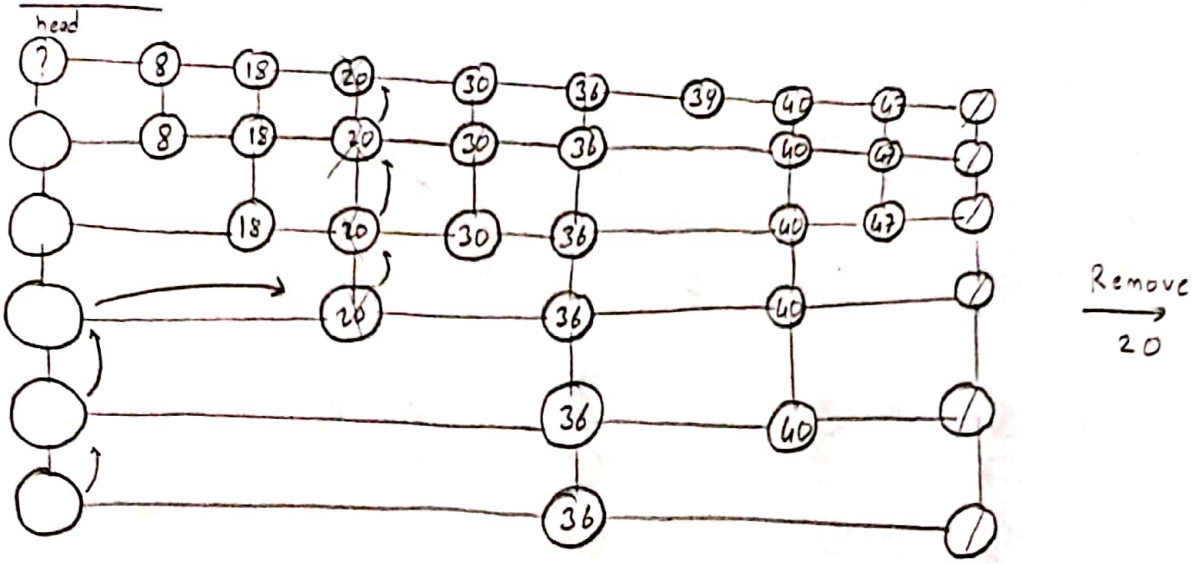


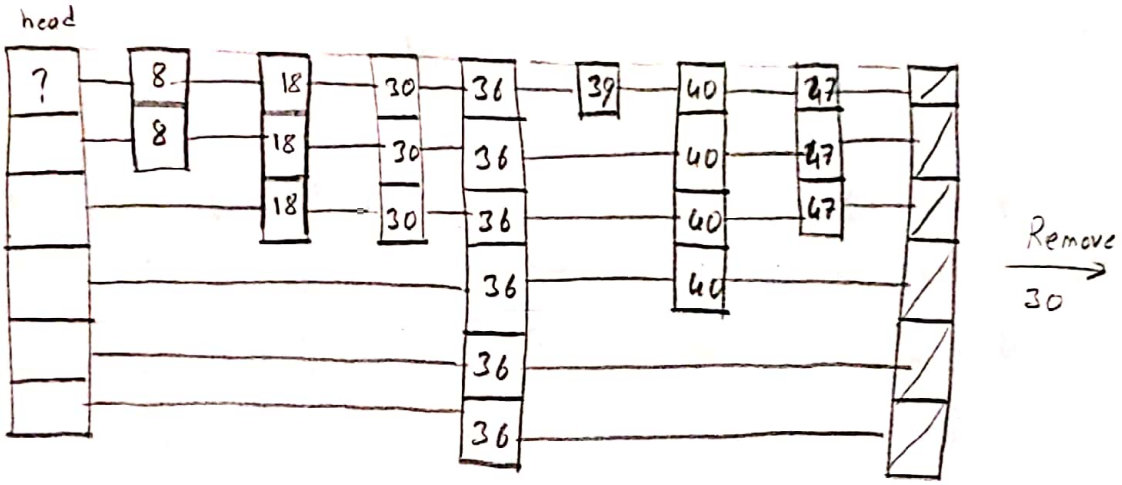


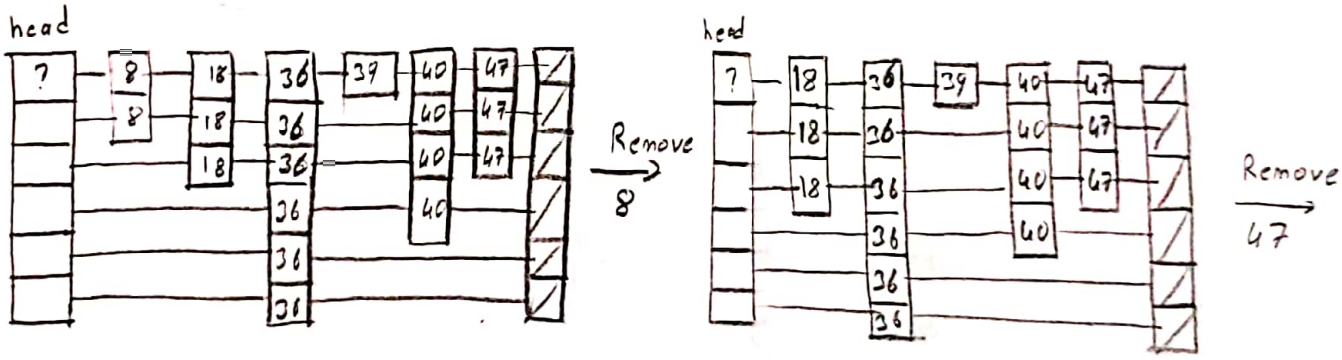
So we can reach any number step less than number of steps as we reach like single or double linked list, we dont need to visit every node we skip some of them

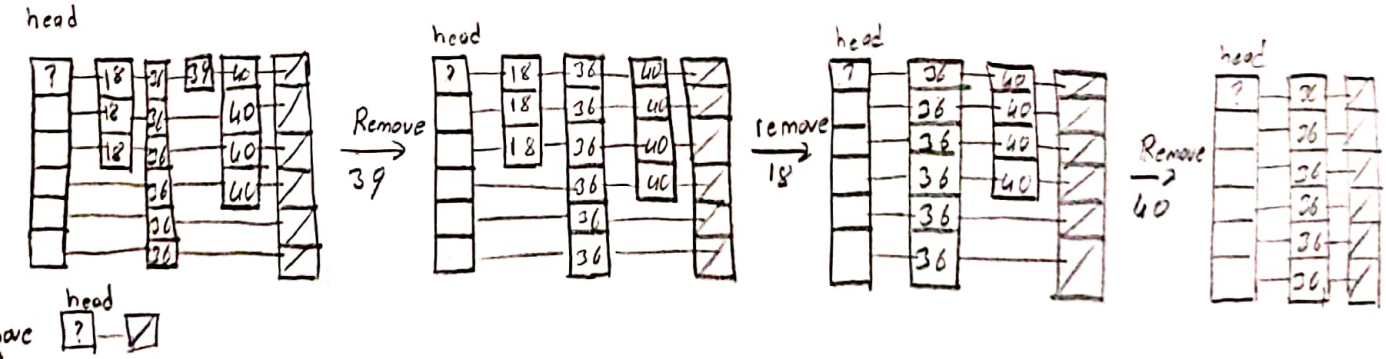
level of anumber is determined by randomly but if we chose it according to number of elements in the list it would be more useful. According to book implementation it chose max level as m and the number of data between 1 to 2^{m-1}. I chose randomly above because I know the number of data. the above 2^{m-1} is valid for data size which is bigger than 15,

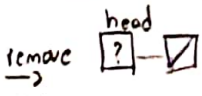
→ Deletion







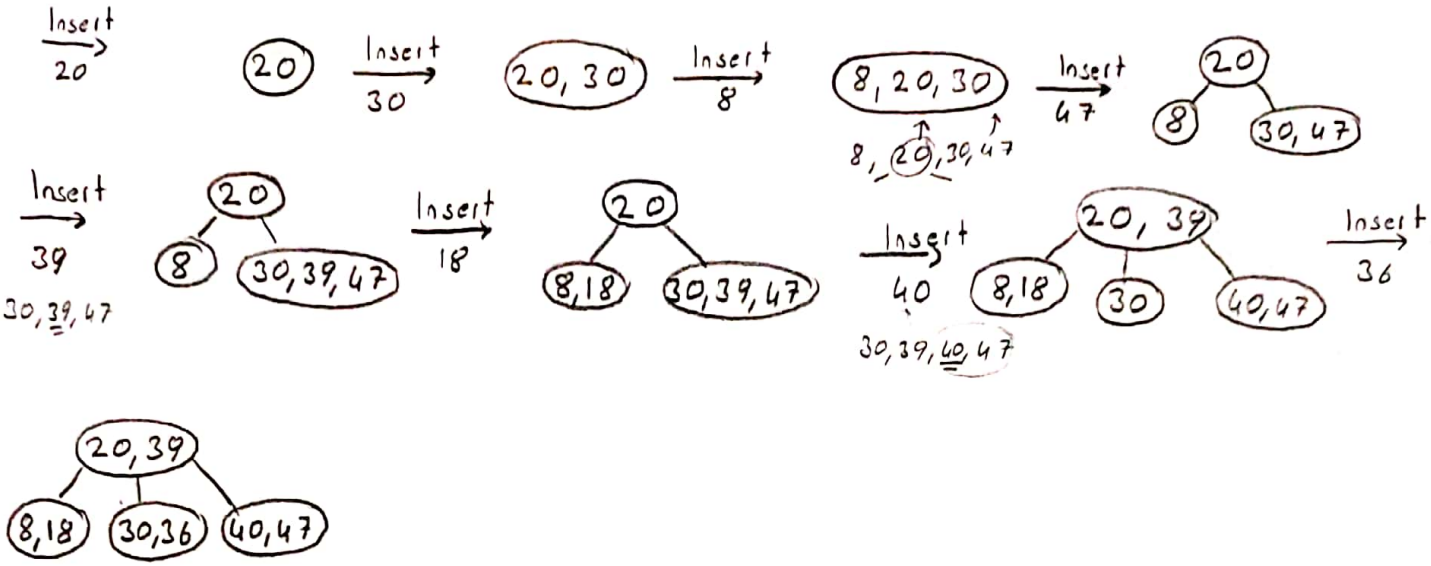




Delete operations like deleting a node from a linked list first we find the datum than goes up until the starting point of datum which will be deleted and we delete one by one and connect the lists again.

→ B-Tree (degree 4)

→ Insertion 20-30-8-47-39-18-40-36



→ Rules for B tree

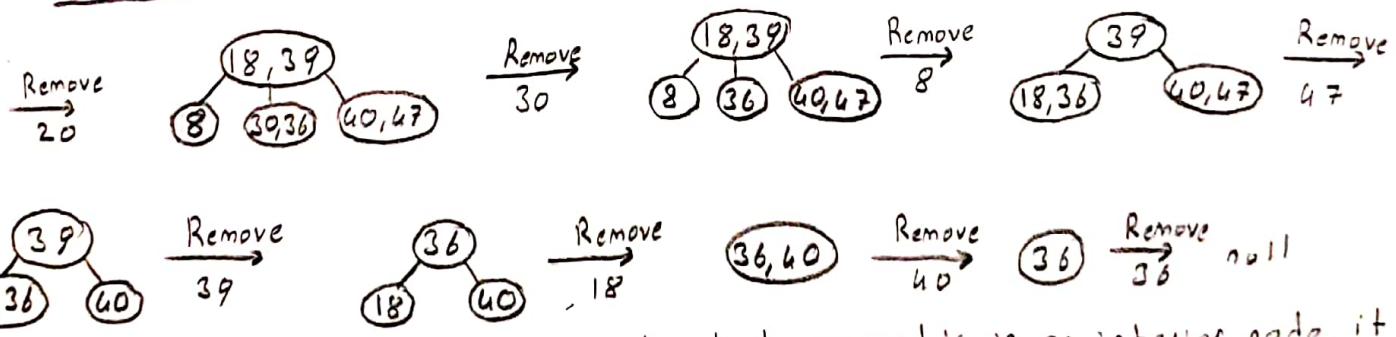
→ For m-degree tree a node must fill at least $\lceil m/2 \rceil$ of node, after that you can add new nodes.

→ The rule is determined above except for the root root can have minimum 2 children

→ All leaf must be at same level

→ Creation process is from bottom to up if there is no room for given datum we shift one of them from bottom to up than we split it new nodes.

→ Deletion



→ Also for deletion the rule (if the item to be removed is in an interior node, it can't be deleted simply because that would damage the B-tree. To retain the B-tree property, the item must be replaced by its inorder predecessor (or its inorder successor) which is in a leaf) is also valid.

References: Lectures
Book
youtube/Abdul Bari
youtube/mycodeschool
youtube/Adam Gowedo
youtube/Jenny's lectures
youtube/RobEdwardsSDSU
online visualization