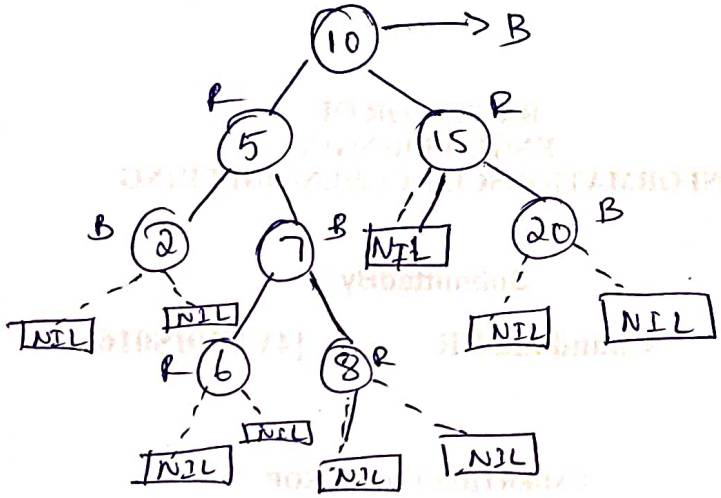
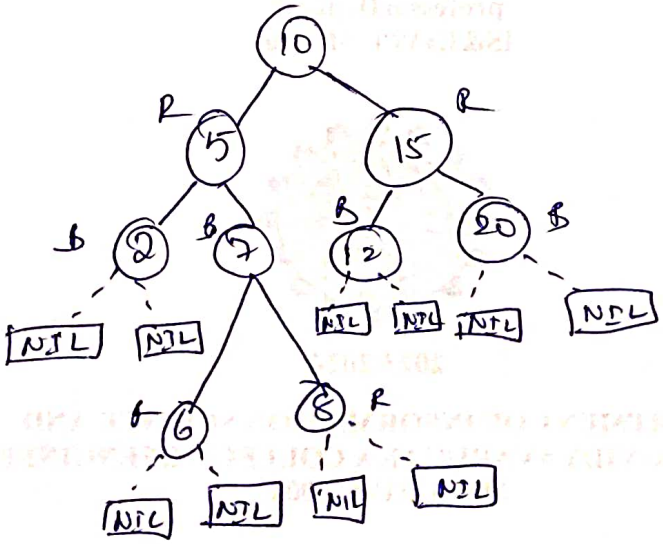
Red Black Trees :-

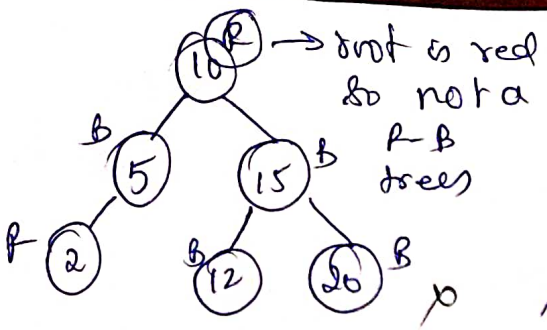
Introduction to Red-Black Trees

→ It is a self-balancing BST
→ Every node is either Black or Red
→ Root is always Black
→ Every leaf which is NIL is black
→ If node is red then its children are Black
→ Every path from a node to any of its descendent NIL node has same no. of Black nodes

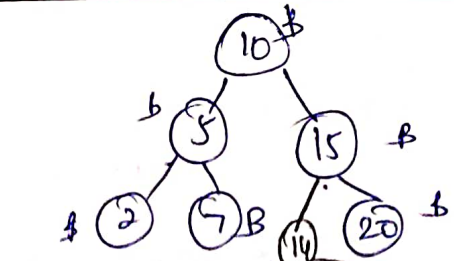




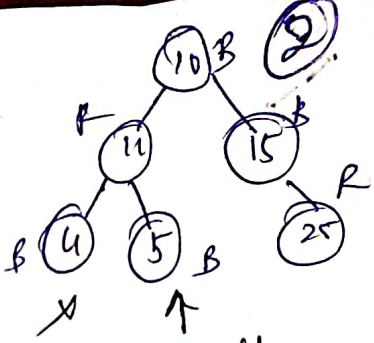
→ AVL-trees are subset of Red black trees, all red black trees cannot be AVL free, because AVL is strightly height balanced trees



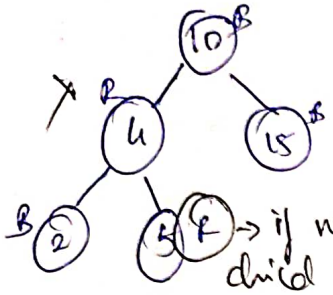
→ root is red so not a R-B trees



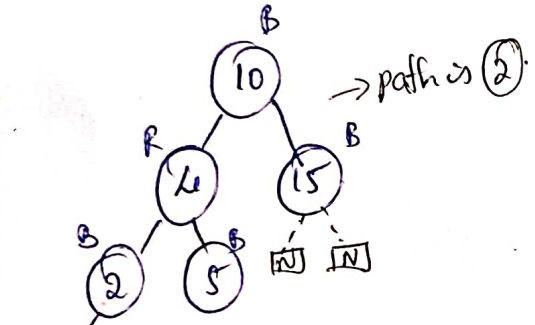
Not a R-B
Last condition not satisfied.
Now its is R-B
↑
Perfect BT



Following all properties of R-B but it is not a BT



→ if node is red child should be Black



→ path is (2).

→ path is (4) $2 \times 2 = 4$ & not more than 4 is acceptable.

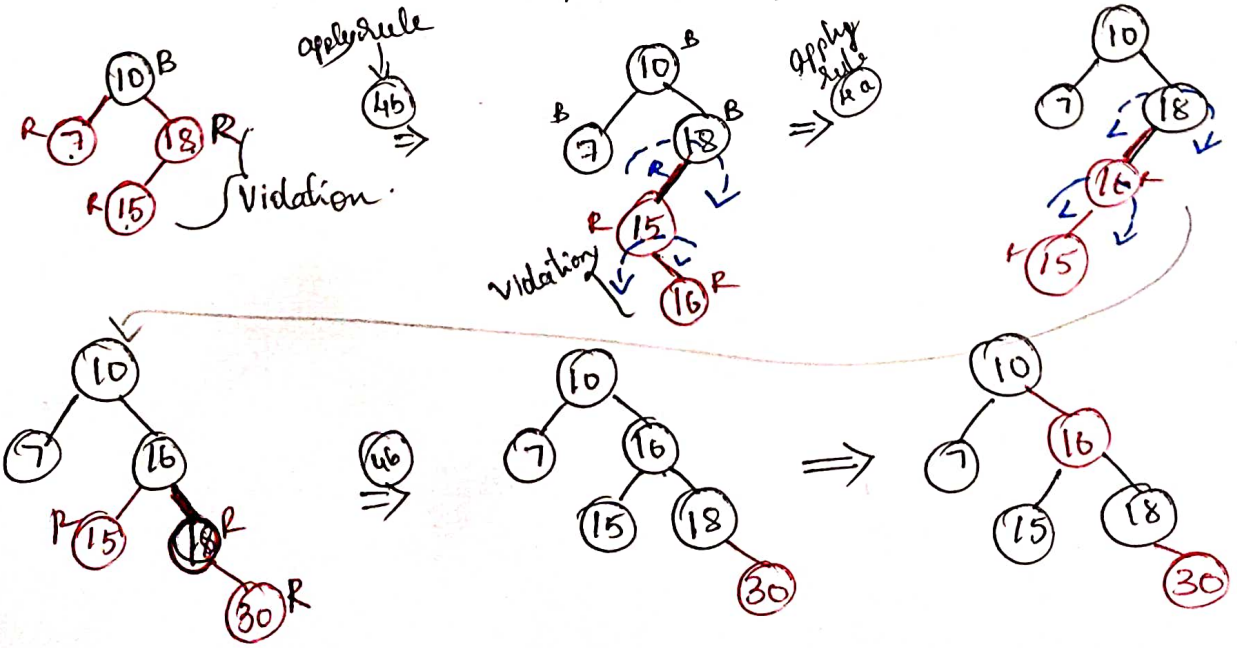
↳ Path from root to extreme leaf node has no more than twice as long as the path from root to nearest leaf node.

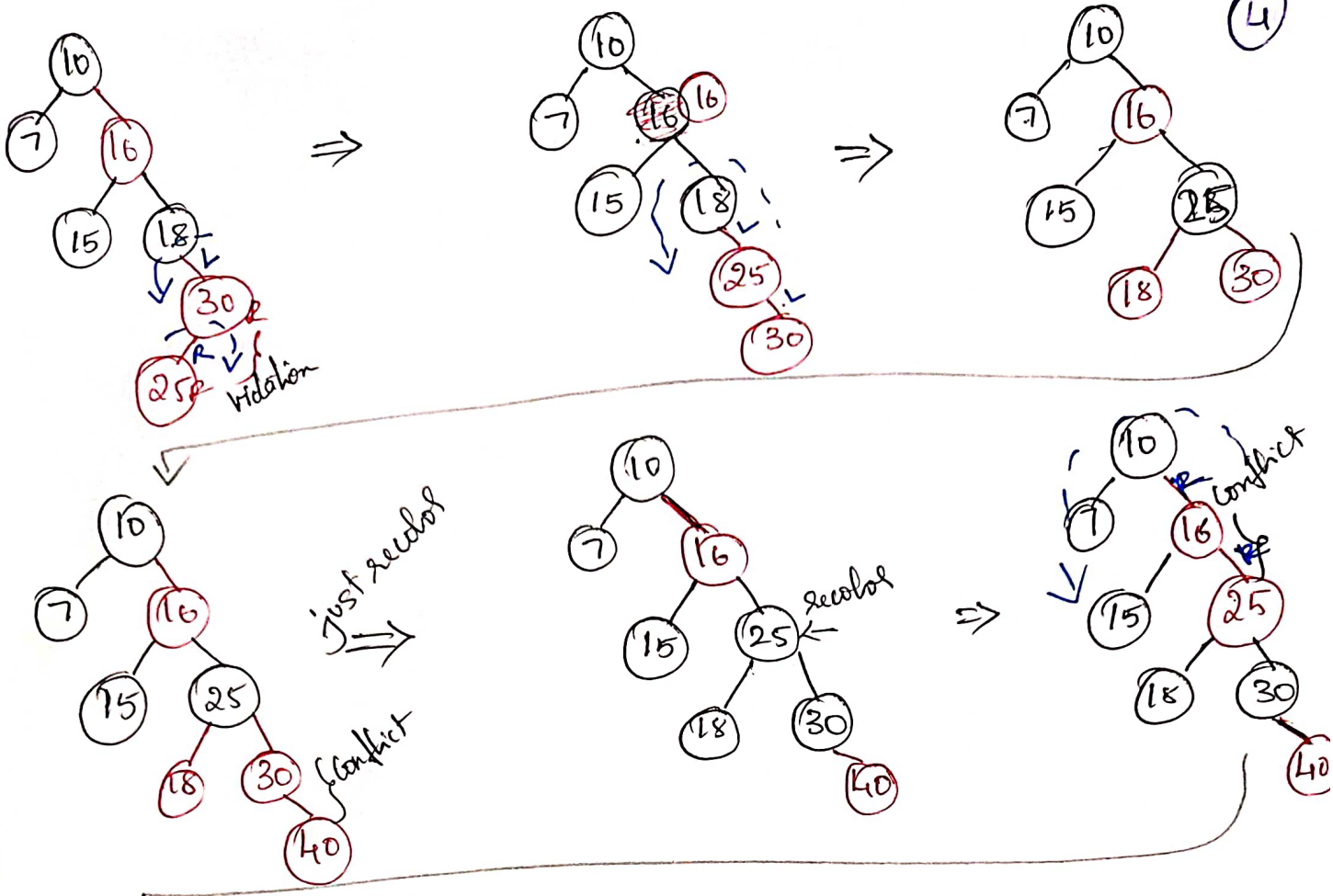
Insertion in R-B Trees

1. If tree is empty, create new node as root node with color black
2. If tree is not empty, create new node as leaf node with color Red
3. If parent of new node is black then exit
4. If parent of newnode is Red, then check the color of parent's sibling of new node.
   a) if color is black or null then do suitable rotation & recolour.
   b) If color is Red then recolor & also check if parent's parent of new node is not root node then recolor it & recheck

→ root = Black
→ no two adjacent red nodes
→ count no. of black nodes in each path.

10, 18, 7, 15, 16, 30, 25, 40, 60, 2, 1, 70





new node 25, its parent is 16 its sibling is 7 which is black so follow 4a