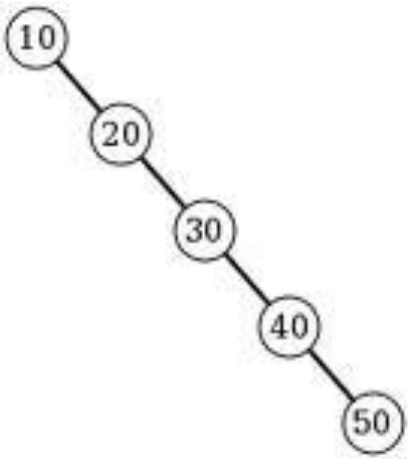
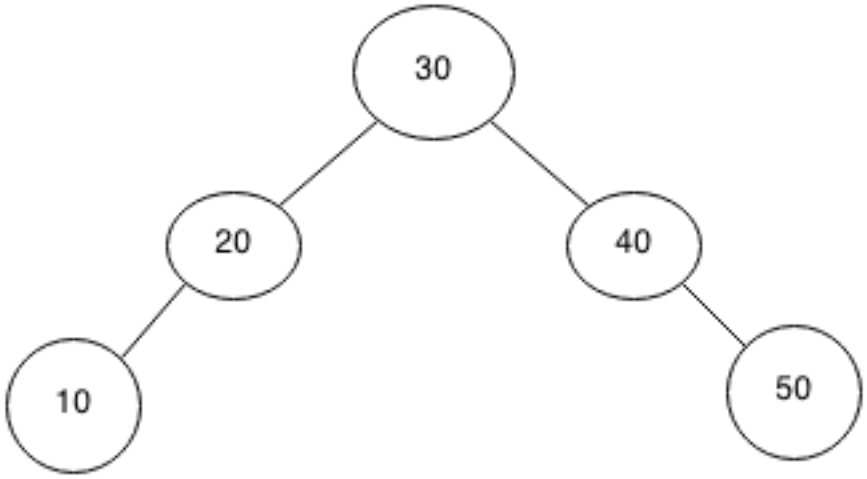
Introduction To Balanced Trees

By Yash Gupta

Balancing

- An unbalanced tree usually the skewed tree violates the properties of binary tree.
- It does not take the advantage in terms of time complexity
- For skewed tree time complexity is equivalent to linear i.e o(n)





Balanced Trees

- AVL TREE
- RED-BLACK TREE

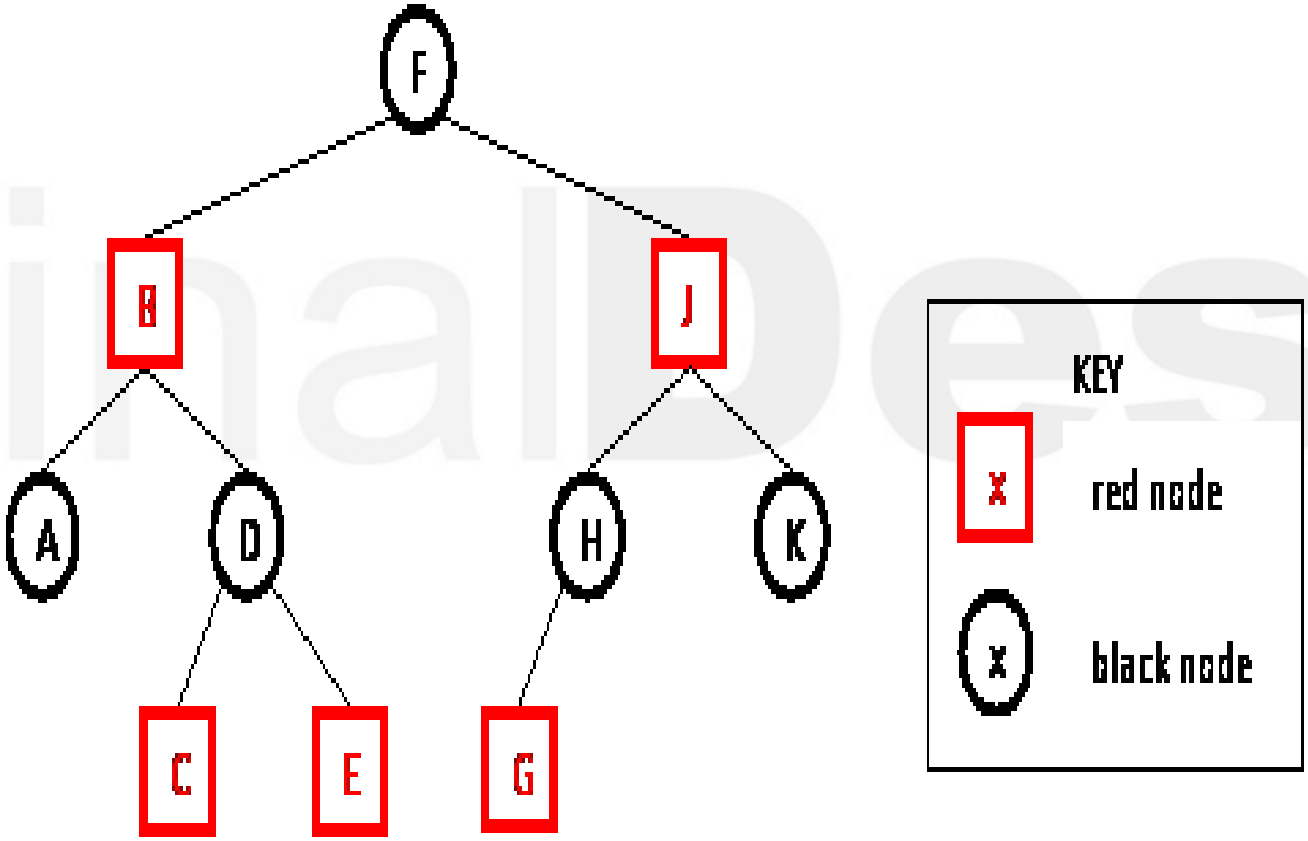
AVL Tree

- An AVL tree is a height balanced tree in which difference of HLeft-HRight <=1
  - Left of Left : A subtree of tree that is left high has also become left high
  - Right of Left : A subtree of tree that is right high has also become right high
  - Left of Right : A subtree of tree that is right high has become left high
  - Right of Left : A subtree of tree that is left high has become right high

RB Tree

- A **red-black tree** is a binary search tree in which each node has a color (red or black) associated with it
- Property:
  - (**root property**) The root of the red-black tree is black
  - (**red property**) The children of a red node are black.
  - (**black property**) For each node with at least one null child, the number of black nodes on the path from the root to the null child is the same.

Example

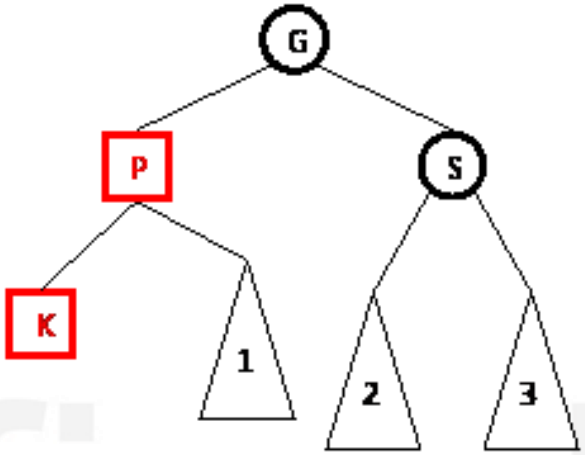


Insert Operation

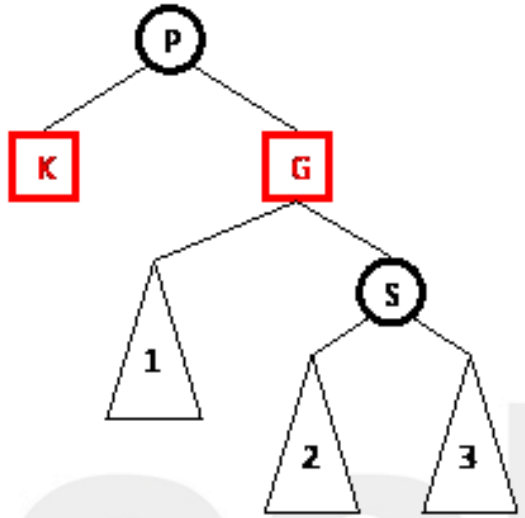
- Let us insert key K into tree T maintaining RB properties and remember initially the color is always RED
- Case 1 :
  - If T is empty replace K by T and recolor to BLACK (ROOT PROPERTY)
- Case 2 : K's parent P Is black
  - If K's parent P Is black , then addition of K did not result into RED property violation

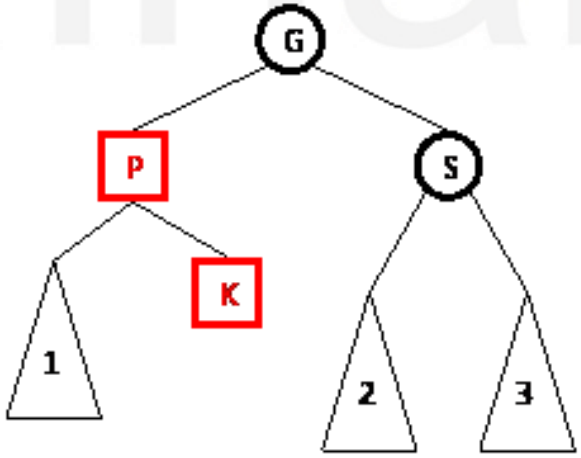
- Case 3 : K's parent P is **red**
  - If K's parent P is red, then P now has a red child, which violates the red property RED-RED situation
- 3a**:** P's sibling S is **black** or null do RESTRUCTURING
  - If P's sibling S is black or null, then we will do a trinode **restructuring** of K (the newly added node), P (K's parent), and G (K's grandparent)
  - Arrange K,P,G in inorder A,B,C. We then make B the parent of A and C, color B black, and color A and C red

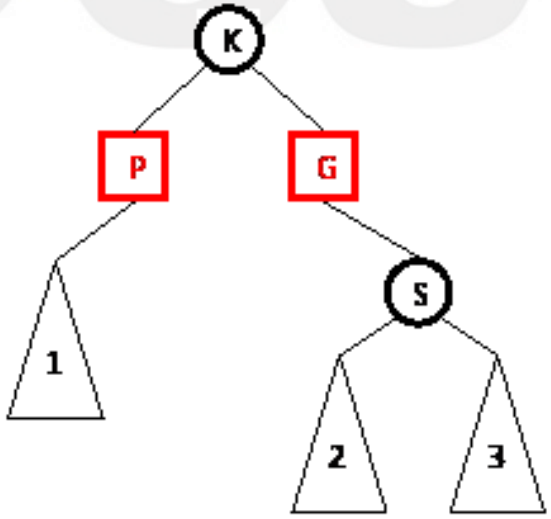
Before restructuring

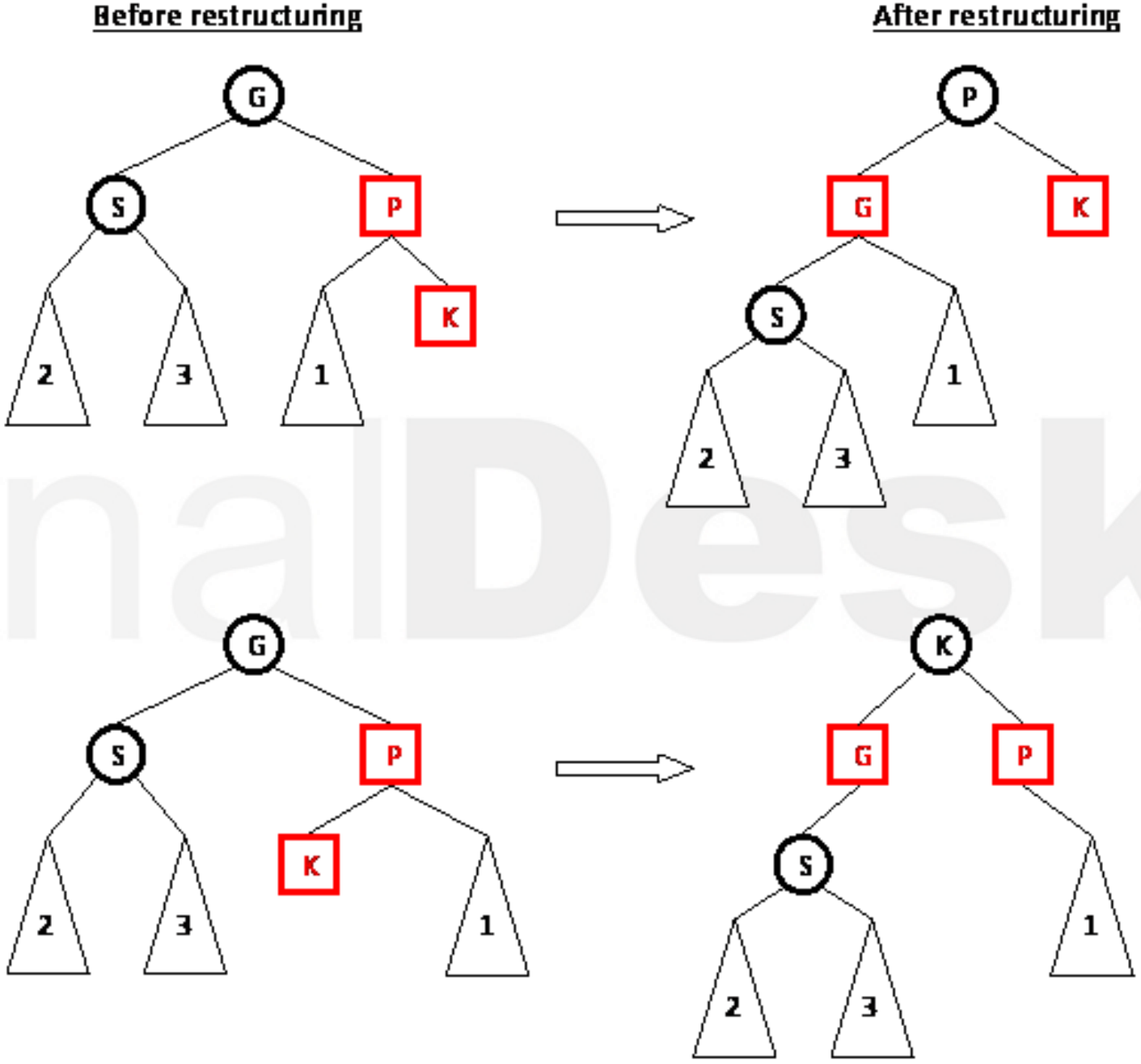


After restructuring

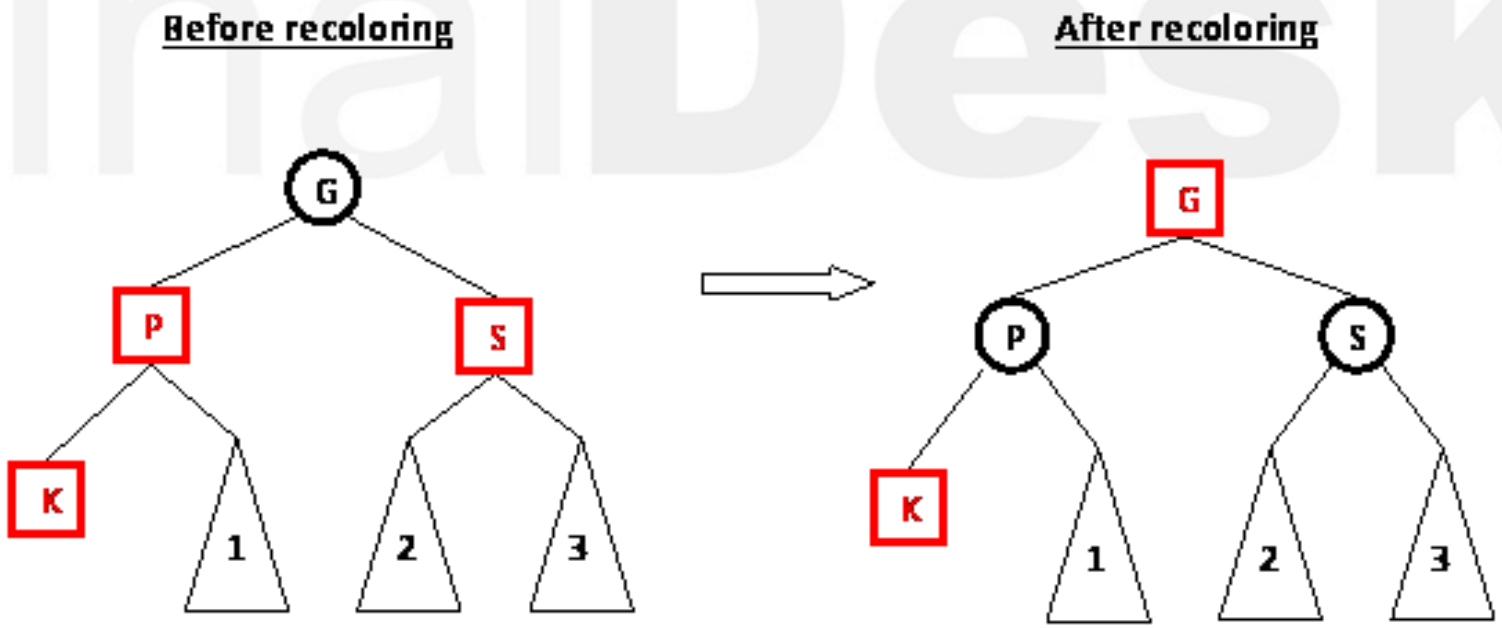







Before restructuring
After restructuring
G
S
P
K
2
3
1
P
G
K
S
1
2
3
G
S
P
K
2
3
1
K
G
P
S
1
2
3

- **3b:** P's sibling S is **red**
  - Do **recoloring** of P, S, and G: the color of P and S is changed to black and the color of G is changed to red (unless G is the root, in which case we leave G black to preserve the root property).



Contact Info

- trainers@finaldesk.com
- rishabh@finaldesk.com
- nilesh@finaldesk.com
- jignesh@finaldesk.com
- yash@finaldesk.com
- anand@finaldesk.com