

Red - Black Tree Datastructure

Red - Black Tree is another variant of Binary Search Tree in which every node is colored either RED or BLACK. We can define a Red Black Tree as follows...

Red Black Tree is a Binary Search Tree in which every node is colored either RED or BLACK.

In Red Black Tree, the color of a node is decided based on the properties of Red-Black Tree. Every Red Black Tree has the following properties.

Properties of Red Black Tree

Property #1: Red - Black Tree must be a Binary Search Tree.

Property #2: The ROOT node must be colored BLACK.

Property #3: The children of Red colored node must be colored BLACK.
(There should not be two consecutive RED nodes).

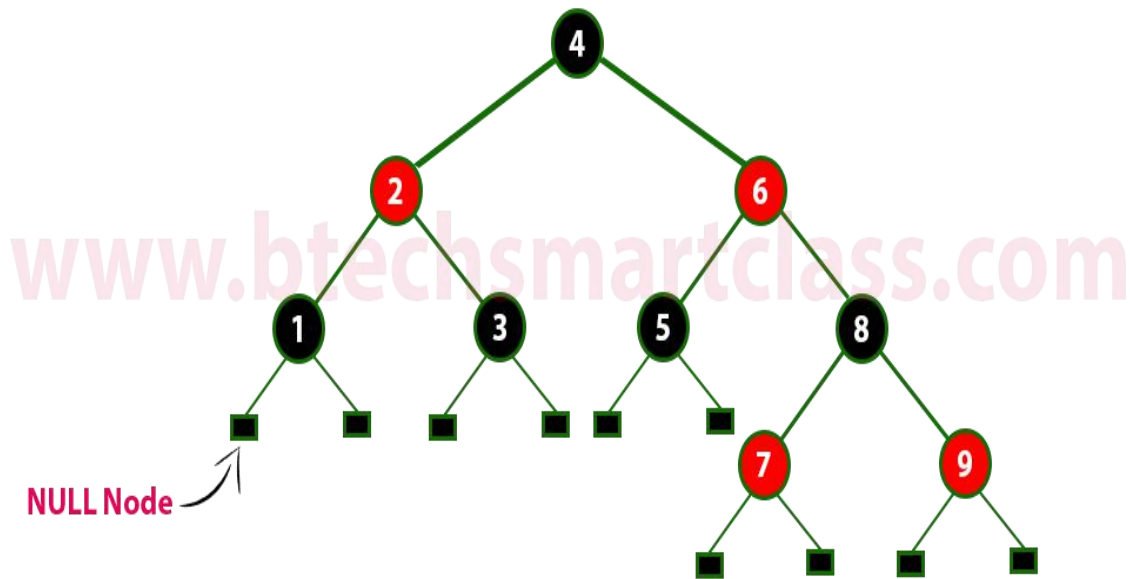
Property #4: In all the paths of the tree, there should be same number of BLACK colored nodes.

Property #5: Every new node must be inserted with RED color.

Property #6: Every leaf (e.i. NULL node) must be colored BLACK.

Example

Following is a Red-Black Tree which is created by inserting numbers from 1 to 9.



The above tree is a Red-Black tree where every node is satisfying all the properties of Red-Black Tree.

Every Red Black Tree is a binary search tree but every Binary Search Tree need not be Red Black tree.

Insertion into RED BLACK Tree

In a Red-Black Tree, every new node must be inserted with the color RED. The insertion operation in Red Black Tree is similar to insertion operation in Binary Search Tree. But it is inserted with a color property. After every insertion operation, we need to check all the properties of Red-Black Tree. If all the properties are satisfied then we go to next operation otherwise we perform the following operation to make it Red Black Tree.

1. Recolor
2. Rotation
3. Rotation followed by Recolor

The insertion operation in Red Black tree is performed using the following steps...

Step 1 - Check whether tree is Empty.

Step 2 - If tree is Empty then insert the **newNode** as Root node with color **Black** and exit from the operation.

Step 3 - If tree is not Empty then insert the newNode as leaf node with color Red.

Step 4 - If the parent of newNode is Black then exit from the operation.

Step 5 - If the parent of newNode is Red then check the color of parentnode's sibling of newNode.

Step 6 - If it is colored Black or NULL then make suitable Rotation and Recolor it.

Step 7 - If it is colored Red then perform Recolor. Repeat the same until tree becomes Red Black Tree.

Example

Create a RED BLACK Tree by inserting following sequence of number 8, 18, 5, 15, 17, 25, 40 & 80.

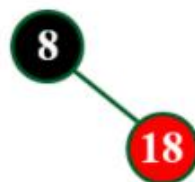
insert (8)

Tree is Empty. So insert newNode as Root node with black color.



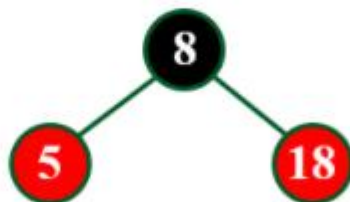
insert (18)

Tree is not Empty. So insert newNode with red color.



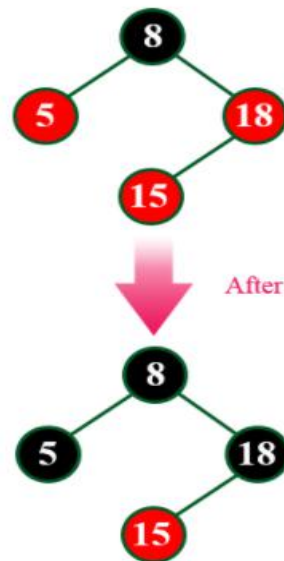
insert (5)

Tree is not Empty. So insert newNode with red color.



insert (15)

Tree is not Empty. So insert newNode with red color.

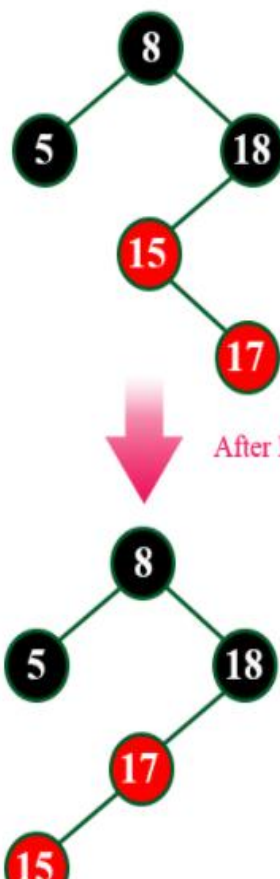


Here there are two consecutive Red nodes (18 & 15). The newnode's parent sibling color is Red and parent's parent is root node. So we use RECOLOR to make it Red Black Tree.

After Recolor operation, the tree is satisfying all Red Black Tree properties.

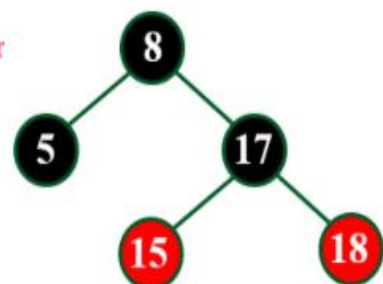
insert (17)

Tree is not Empty. So insert newNode with red color.



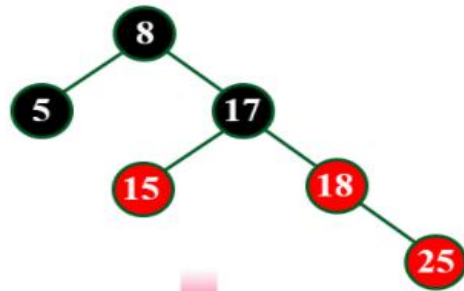
Here there are two consecutive Red nodes (15 & 17). The newnode's parent sibling is NULL. So we need rotation. Here, we need LR Rotation & Recolor.

After Right Rotation & Recolor



insert (25)

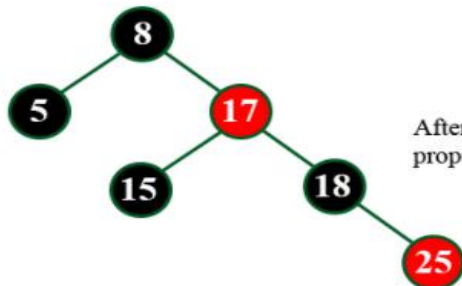
Tree is not Empty. So insert newNode with red color.



Here there are two consecutive Red nodes (18 & 25).
The newnode's parent sibling color is Red
and parent's parent is not root node.
So we use RECOLOR and Recheck.



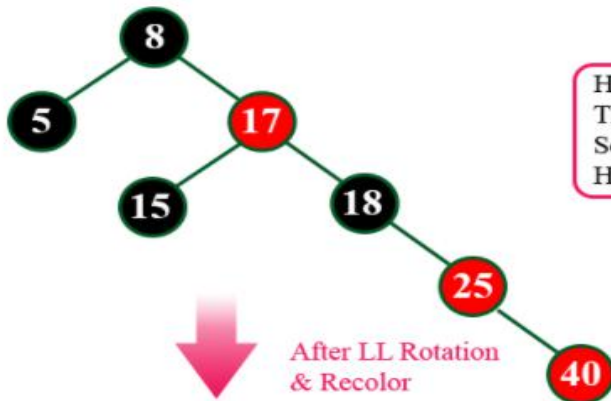
After Recolor



After Recolor operation, the tree is satisfying all Red Black Tree properties.

insert (40)

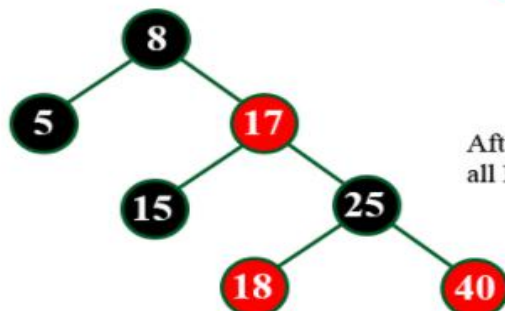
Tree is not Empty. So insert newNode with red color.



Here there are two consecutive Red nodes (25 & 40).
The newnode's parent sibling is NULL
So we need a Rotation & Recolor.
Here, we use LL Rotation and Recheck.



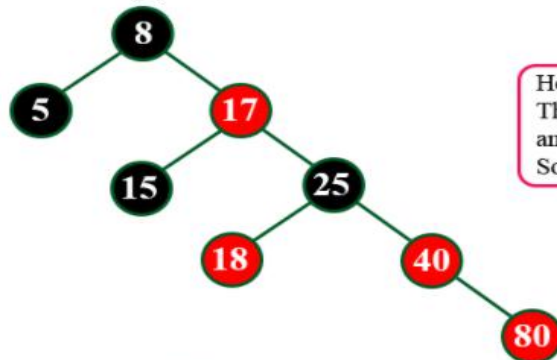
After LL Rotation
& Recolor



After LL Rotation & Recolor operation, the tree is satisfying all Red Black Tree properties.

insert (80)

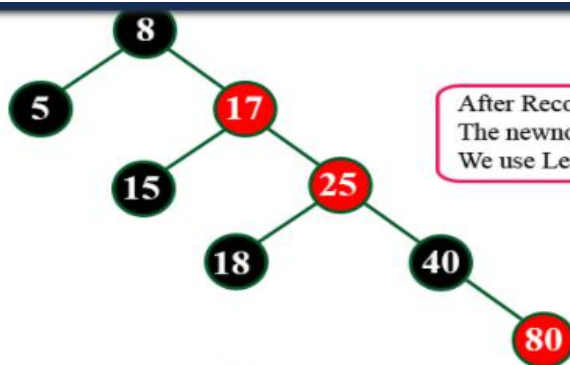
Tree is not Empty. So insert newNode with red color.



Here there are two consecutive Red nodes (40 & 80). The newnode's parent sibling color is Red and parent's parent is not root node. So we use RECOLOR and Recheck.



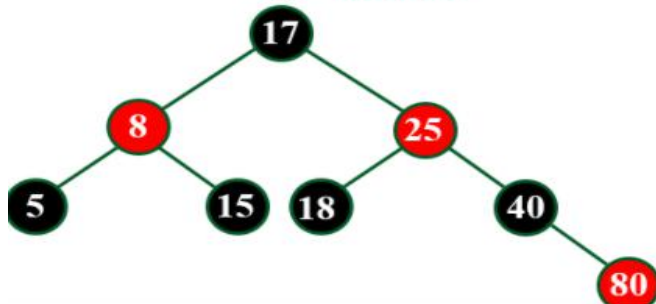
After Recolor



After Recolor again there are two consecutive Red nodes (17 & 25). The newnode's parent sibling color is Black. So we need Rotation. We use Left Rotation & Recolor.



After Left Rotation & Recolor



Finally above tree is satisfying all the properties of Red Black Tree and it is a perfect Red Black tree.