Homework 8 : Algorithms and Data Structures

Digdarshan Kunwar

April 2019

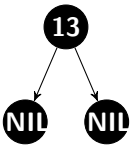
Problem 8.1

Understanding Red Black Trees

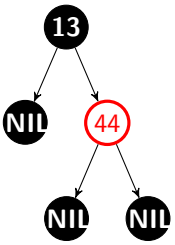
(a) Draw (or describe by using preorder traversal) the red-black trees that result after successively inserting the values step by step in the following order [13,44,37,7,22,16] into an empty red-black tree. You are required to draw (or describe by using preorder traversal) the tree after each insertion, as well as any additional recoloring and balancing.

Initially we have: [13,44,37,7,22,16]

Adding 13

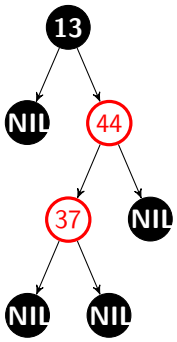


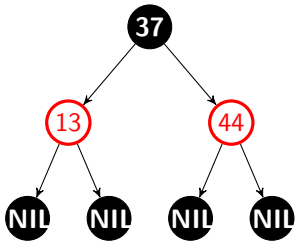
Adding 44



Adding 37

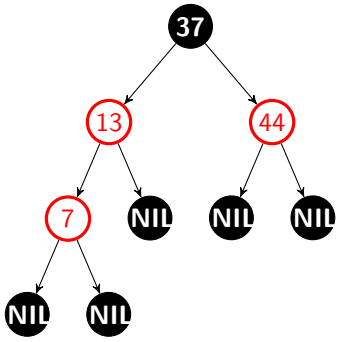
Initially : **After Fix:**



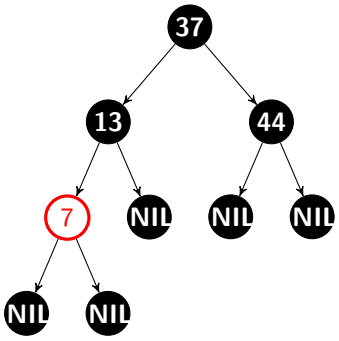


Adding 7

Initially :

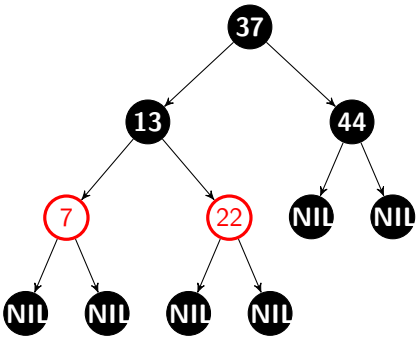


After Fix:

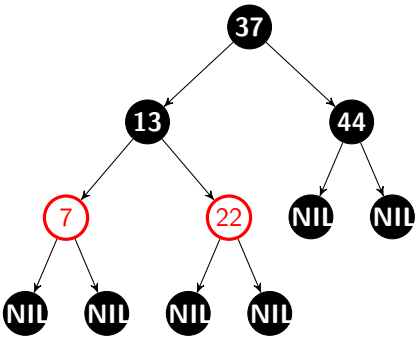


Adding 22

Initially :



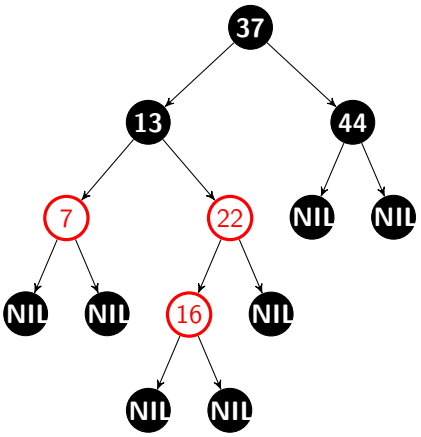
After Fix:

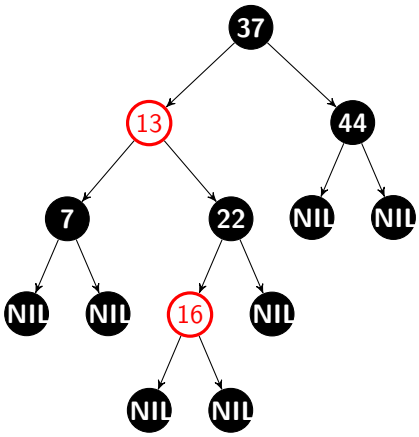


Adding 16

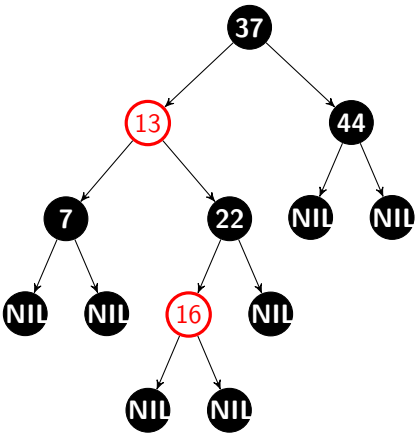
Initially :

After Fix:



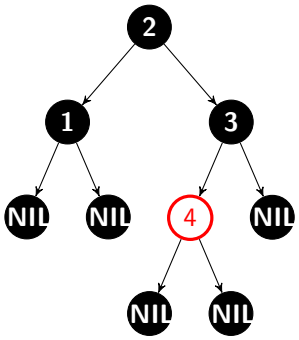


Finally :

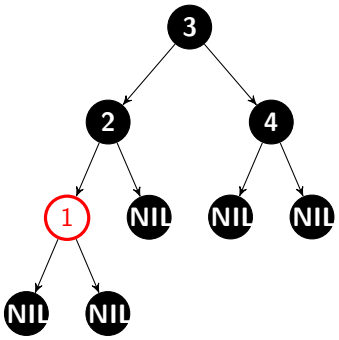


(b) These are the only two valid ways of having a RBT with the elements 1,2,3,4.

Case 1 :



Case 2



(c) **Consider a red-black tree formed by inserting n nodes with the algorithm described in the lecture slides. Prove that if n > 1, the tree contains at least one red node.**

Here when we have n greater than 1. We consider these three possible cases so:
Here z is the newly added element and the p is the parent of that element.

When n is equal to 2, red node ,black node are 1. $\Longrightarrow$ Base Case
Then every time RB-INSERT is called, one element will be inserted. The red node that is being inserted then will call RB-INSERT-FIX-UP. If there are other cases , the number of red nodes will not decrease. For general case, the newly inserted node must be a red node. So when n>2, the number of red nodes is at least one.
Case a:
z and z.p.p are both RED, and after the rotation if z.p could not be the root, thus z.p is RED after the fix up.
Case b:
z and z.p.p are both RED and if the loop terminates, then if z could not be the root, thus z is RED after the fix up.
Case c:
z is RED and z could not be the root, thus z is RED after the fix up.

Therefore, there is always at least one red node in the tree.

Problem 8.2

Implementing Red Black Trees

!! NOTE !!
Implementation of RBT is in RBT folder.

The implementation is in redblacktree.py
$: make redblacktree
or
$: make all

References :
Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (n.d.). Introduction to algorithms.