### CS-202 ASSIGNMENT-04

S. SAMARTH REDBY BIGIOG GIROUR- 25

(1) Given n'integers in Sorted order, we can use the following algorithm to construct a RB Tree in O(n) sime:

Alapoithon:

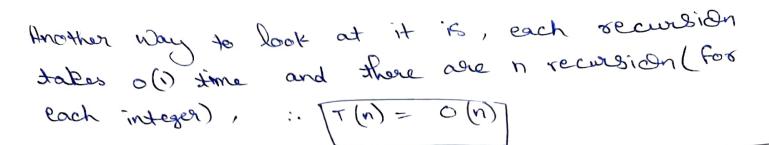
- Make the middle integer of the Sorted agray the root of the Red black tree. Middlemost number [71] the number.
  - Thon, make the middle integer of left part of organ (number to the left of [n/2]th) as the left child of the root node, and the niddlemost integer of right Part of array as the right child of root node.
- Continue this process for the left side and right side arrays recursively till all the integers have been Placed.
  - Coloning all nodes black except in ones of the bottom upon (colone them had).

    would result in a ted black tree Satisfying all the conditions. All null nodes with black.

The algorithm takes sime T(n) Such that

 $\left\{ \tau(n) = 2\tau(n_{12}) + c \right\} \longrightarrow \tau(n) = O(n)$ This can be easily

verified by Substitution.



- 2)-Given a red black tree with m elements, we can find an array with non-decreasing order of elements of 86 tree using inorder traversal. This can be done in O(m) time.
  - : Given two PB trees with m and n number of elements, we can find the Sorted arrays (non-decreasing) of these two trees in o(m) and o(m) time respectively.
  - Now create a new array using these two arrays using merge sort. This can be done in O(n+m) time.
  - we have an array of (17+m) non-decreasing (sorted) elements. Using the algorithm used in quastion 1, the RB Tree can be created in O(m+n) time.
  - :. Total Time taken = 0 (m+n) + 0 (m+n) + 0 (n) + 0 (m)

    Post 1: m = Tr

Total Time =  $O(n+\sqrt{n})$  +  $O(n+\sqrt{n})$  + O(n) + O(n)= O(n) [Since  $n > \sqrt{n}$ ]

Similarly, Post 2:  $m = n_{12}$ Total Time =  $O(3n_{12}) + O(3n_{12}) + O(n) + O(n_{12}) = O(n)$ 

- (3) We know that the morder traversal of BST.

  generates a Sexted (non-decreasing) array of
  the elements Present in the BST.
  - So, using Inorder traversal, we get two arrays representing the elements of the trees.
  - we compare both the arrays, if they are same, then both BSTs have same Set of olements
  - briven a BST, with n elements, the inorder traversal takes ((n) time, And, compassing the two sorted arrays also takes ((n) time.
    - : Time taken to compare two BSTs = O(n) + O(n)= O(n).

Inorder Traverse ( root):

Frorder Traverse (root -> left)
Point value of node
Inorder Traverse (root -> right)

Mr. Lazyrus make a tweak in red-Black trees, Such that a node is not completely deleted to make deletion operation easier. Instead her adds a boolean variable which denotes whether the node is active or not (i.e deleted or not)

## Effectiveness:

- As mantioned above, the node is not completely delated, so it still has the Property of colour. Therefore, it maintains the colour Property of the RB tree after delation and no extra recoverangements are required after delation.
- Insertion of a new node can be done the Same way as before, considering the colours of the nodes.
- As all elements are distinct, we cannot insent a node with the Same value as that of a deleted (inactive) node. So we Search the Same way as before and Search among active nodes to Search a node.

#### Inaffectiveness:

There is wastage of space as the nodes we delate are not really delated and only the booken is changed to inactive. It still uses the Same space after deletion, and we cannot reuse this space.

If there are too many mactive (deleted) roodes, then insertion and Searching would be a lot more time consuming than normal RB tree.

# 5) For Insertion:

- Potations: 0 (i) [at most 2 rotations]

  Potation happens when the uncle is of black

  Colour. There are 4 types of rotations,

  [LL, LP, PP, PL]. At most 2 rotations happen.
- >> Recolouring: O (log(n))

Recolousing happens when the Parent and uncle are of Real colour. We recolor Parent, uncle and opendoperant and then recur on agrand-larent. This can repeat hts times, where h is height of tree, hence recouloring: - o (logn)

## For Delation:

- and we do not have to recur on any Parent like in the case of recolouring.
  - Fe-coloring O (logn)

    Recoloring happens when Sibling and both it's children are black. This can repeat h times, where h is height of the tree, and hence there are O(logn) recolorings.

- 6) To find the Successor of a node in a Red Black Tree, we follow the following Procedure:-
  - Successor lies in the right Subtree . So, 39 the the right Subtree and return the node with minimum key ralue in the right Subtree.
  - is one of the ancestors. So, we travel up the Parent Pointer until we see a node which is left child of its Poscant. The Poscant of Such a node is successor.