CSC265 Fall 2020 Homework Assignment 2

due Tuesday, September 29, 2020

A *leaf-oriented binary search tree* is an implementation of a dictionary in which the elements in the dictionary are stored at the leaves of a binary search tree. The keys of non-leaf nodes are only used for searching. They may or may not be the keys of elements in the dictionary. Every non-leaf node has exactly two children.

A LOBS is a leaf-oriented binary search tree with the following properties:

1. Every node is coloured red or black.
2. The root is black.
3. All leaves are black.
4. Every leaf has a key, a value, a colour, and a pointer to its parent.
5. Every non-leaf node has a key, a colour, and pointers to its left child, right child, parent, and the rightmost leaf in its left subtree.
6. The number of black nodes on all root-to-leaf paths is the same.
7. Every non-leaf node has a black right child.
8. Every non-leaf node has a left child, which is either black or red.
9. Any root-to-leaf path contains at most two consecutive red nodes.

(a) Explain how to insert an element into a leaf-oriented binary search tree.

(b) If L is a LOBS and you insert an element into L using your algorithm from (a), what properties can be violated?

(c) Give an $O(\log n)$ time algorithm to perform INSERT(L, x), where L is a LOBS, n is the number of elements in the dictionary, and x is a pointer to a new node containing a key and a value, whose colour is black, and whose parent pointer is NIL.
Use diagrams and English explanations, as in class, rather than pseudo-code.
In a clear, organized way, describe all the cases that can arise, describe what transformations can be performed so that the resulting tree is a LOBS, and explain why the tree resulting from your transformations satisfies all of the required properties of a LOBS.

(d) Explain why your algorithm runs in $O(\log n)$ time.