Assignment 2
Implementing Red Black Tree

1 Introduction

1.1 Red Black Tree

A red black tree is a kind of self-balancing binary search tree in computer science. Each node of the binary tree has an extra bit, and that bit is often interpreted as the color (red or black) of the node. These color bits are used to ensure the tree remains approximately balanced during insertions and deletions. Balance is preserved by painting each node of the tree with one of two colors in a way that satisfies certain properties, which collectively constrain how unbalanced the tree can become in the worst case. When the tree is modified, the new tree is subsequently rearranged and repainted to restore the coloring properties. The properties are designed in such a way that this rearranging and recoloring can be performed efficiently.

2 Requirements

You are required to implement the following interfaces:

2.1 Red Black Tree

1. getRoot: return the root of the given Red black tree.
2. isEmpty: return whether the given tree isEmpty or not.
3. clear: Clear all nodes in the given tree.
4. search: return the node associated with the given key or null if no value is found.
5. contains: return true if the tree contains the given value and false otherwise.
6. insert: Insert the given value in the tree while maintaining the red black tree properties. If the value is already present in the tree, reject its value.
7. delete: Delete the node associated with the given value. Return true in case of success and false otherwise.

3 Analysis

You are required to do the following:

3.1

1. you have to compare between AVL and redBlack trees in terms total time for insertion and deletion.

2. you have to run two algorithms using random generated string with constant size and gradually increasing number of nodes. For example you have to test two algorithm for n=10,100,1000 and so on then make same test for deletion.

3. Your report should contain your work, charts and analysis.

4 Notes

- Continue to work as the first lab team.

Good Luck