UCR - CS 14

Assignment 4

Deliverables: Create a single pdf file that contains your answers and your C++ code. Then create a zip file that contains this pdf file along with all your code source files. Submit this zip file in iLearn.

Deadline: 11/12/2019 11:59 pm.

Exercise 1

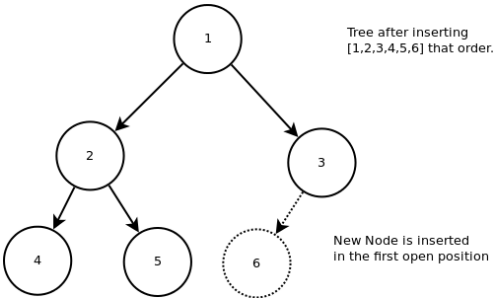
Use provided C++ skeleton files to insert your code.

A.

Implement a binary tree class MyTree using pointers (define struct BinaryNode to store internal nodes), where each node has a pointer to its children and to its parent, and each node holds two variable, a string myString and an integer myInt.

Write functions in MyTree class to do the following:

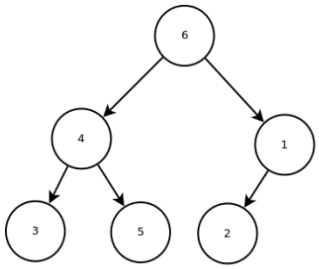
a. Insert (int,string): Insert new node into first available position (to keep the tree almost complete). E.g.,



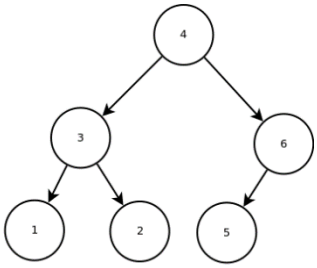
   To get full points your Insert functions should be faster than O(n), where n is # nodes in tree. Hint: you may use an additional private variable in MyTree class.

b. Preorder (): Output all strings in pre-order.
c. FindMax (): returns a pointer to the node with maximum myInt.
d. MakeBST (): converts the binary tree into a binary search tree (BST) with respect to myInt. That is, move around node values (myString and myInt) to satisfy the BST property. Do not change the structure of the tree (i.e. the pointers) but only swap myString and myInt values. (Hint: if you need to sort an array, you can use STL's sort() method)

e.g.,



Original Tree after inserting [6,4,1,3,5,2]



BST: Notice we keep the ORIGINAL structure

B.

What is the big-Oh complexity of your functions above?

Also, what is the space complexity of your functions? Are they all in-place? If not, how much extra space do they need?

C.

Test and measure the performance of your functions.

Create sequences of 100, 1000, 10000, 100000 random (int, string) pairs and insert them into MyTree (using Insert(.,.) function). Measure the times to (a) build the tree, (b) execute Preorder(), (c) execute FindMax(), (d) execute MakeBST().

Report the times for each tree size in a table.