CS 2413 001 Summer 2000 Homework #5

Quiz to be held in class 1:20pm Friday 21 July 2000

You will need extra sheets of notebook paper to write your answers on.

1. Given a preorder traversal and an inorder traversal of a binary tree, a unique binary tree that satisfies the traversals can be constructed. Given the following preorder and inorder traversals, construct the binary tree.
   - Preorder: `a, e, f, h, g, b, c, d`
   - Inorder: `h, f, e, g, a, c, b, d`
2. Let `T` be a binary search tree with the same structure as the tree in the previous question.
   (a) Insert `j` into `T`. Then insert i. Show both results.
   (b) Remove `g` from `T`. (Note that this is the **original** T, not the result after insertion.) Then remove f. Show both results.
3. Two binary trees are said to be *equivalent* if and only if they have the same structure and the same values at corresponding nodes.
   (a) Write a method that overloads `operator==` to determine whether two trees are equivalent, using an enumerator and without recursion. What is the time complexity?
   (b) Write a method that overloads `operator==` to determine whether two trees are equivalent, not using an enumerator and with recursion. What is the time complexity?
4. Given a binary search tree, an inorder traversal gives a sorted list. Assume that this sorted set of elements is stored in an array named `A` of size `n`. A *globally balanced binary search tree* has `A[mid]` in the root of the tree, the left subtree is the globally balanced binary search tree of the elements in the subarray `A[0..mid-1]`, and the right subtree is the globally balanced binary search tree of the elements in the subarray `A[mid+1..n-1]`. Write a method to construct a globally balanced binary search tree from a sorted array `A`. What is the time complexity?
5. Let `T1` and `T2` be two binary search trees such that every element of `T1` is less than every element of `T2`. Let `x` be a value such that `T1` < `x` < `T2`. Write a method `join(T1,x,T2)` that produces a binary search tree containing the elements of `T1`, `x` and `T2`. Note that no node of the result of `join` should point to any node of `T1` or `T2`; i.e., the method must copy every node of each tree. What is the time complexity of `join`?

References

S. Rhadakrishnan, L. Wise & C. N. Sekharan, *Object-Oriented Data Structures Featuring C++*, 1999.