Homework Assignment #6

Note

This assignment is due 2:10PM Tuesday, April 21, 2015. Please write or type your answers on A4 (or similar size) paper. Drop your homework by the due time in Yih-Kuen Tsay's mail box on the first floor of Management College Building II. Late submission will be penalized by 20% for each working day overdue. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

There are five problems in this assignment, each accounting for 20 points. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. Perform insertions of the numbers 7, 5, 2, 1, 4, 3, 6 (in this order) into an empty AVL tree. Show each AVL tree after a number has been inserted. If re-balancing operations are performed, please also show the tree before re-balancing and indicate what type of rotation is used in the re-balancing.

2. The *Partition* procedure for the `Quicksort` algorithm discussed in class is as follows, where *Middle* is a global variable.

   Partition $(X, Left, Right)$;
   begin
   $pivot := X[Left]$;
   $L := Left$; $R := Right$;
   while $L < R$ **do**
   while $X[L] \leq pivot$ and $L \leq Right$ **do** $L := L + 1$;
   while $X[R] > pivot$ and $R \geq Left$ **do** $R := R - 1$;
   if $L < R$ **then** $swap(X[L], X[R])$;
   $Middle := R$;
   $swap(X[Left], X[Middle])$
   end

   (a) Apply the *Partition* procedure to the following array (assuming that the first element is chosen as the pivot).

6	16	9	10	13	12	2	1	11	7	15	5	4	8	3	14

   Show the result after each exchange (swap) operation.

   (b) Apply the `Quicksort` algorithm to the above array. Show the result after each partition operation.

3. (6.10) Find an adequate loop invariant for the main while loop in the *Partition* procedure of the `Quicksort` algorithm, which is sufficient to show that after the execution of the last two assignment statements the array is properly partitioned by $X[Middle]$. Please express the loop invariant as precisely as possible, using mathematical notation.

4. (6.32) Prove that the sum of the heights of all nodes in a complete binary tree with n nodes is at most $n - 1$. (A complete binary tree with n nodes is one that can be compactly represented by an array A of size n, where the root is stored in $A[1]$ and the left and the right children of $A[i]$, $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$, are stored respectively in $A[2i]$ and $A[2i + 1]$. Notice that, in Manber's book a complete binary tree is referred to as a balanced binary tree and a full binary tree as a complete binary tree. Manber's definitions seem to be less frequently used. Do not let the different names confuse you. "Balanced binary tree" in the problem description is the same as "complete binary tree")

5. Below is a variation of the n-coins problem.

   > You are given a set of n coins $\{c_1, c_2, \ldots, c_n\}$, among which at least $n - 1$ are identical "true" coins and at most one coin is "false". A false coin is either *lighter* or *heavier* than a true coin. Also, you are given a balance scale, which you may use to compare the total weight of any m coins with that of any other m coins. The problem is to find the "false" coin, or show that there is no such coin, by making some sequence of comparisons using the balance scale.

   (a) For the case of $n = 12$, design a scheme to find the false coin (if there is one) with only three comparisons using the balance. Please use c_1, c_2, ..., c_{12} to identify the coins in your scheme.

   (b) Prove that, when $n = 12$, it is not possible to find the false coin with just two comparisons, implying that using just three comparisons is optimal. (Hint: think about decision trees and how many possible outcomes there can be.)