Homework Assignment #2

Note

This assignment is due 2:10PM Tuesday, October 1, 2019. 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 2. 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. You must use *induction* for all proofs. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. Consider again the inductive definition in HW#1 for the set of all binary trees that store non-negative integer key values:

   (a) The empty tree, denoted $\perp$, is a binary tree.

   (b) If t_l and t_r are binary trees, then $node(k, t_l, t_r)$, where $k \in Z$ and $k \geq 0$, is also a binary tree.

   Refine the definition, as you did for Problem 5(c) of HW#1, to include only binary search trees such that an inorder traversal of a binary search tree produces a list of all stored key values in *increasing* order. Then, define inductively a function that outputs the rank of a given key value (the position of the key value in the aforementioned sorted list) if it is stored in the tree, or 0 if the key is not in the tree.

2. (2.24) We can define **anti-Gray codes** in the following way. Instead of minimizing the difference between two consecutive strings, we can try to maximize it. Is it possible to design an encoding for any even number of objects such that each pair of two consecutive strings differ by k bits (where k is the number of bits in each string)? How about $k-1$ bits (or $k-2$, $k-3$, etc.)? If it is possible, find an efficient construction.

3. Consider the following recurrence relation:

$$\begin{cases} T(0) = 0 \\ T(1) = 1 \\ T(h) = T(h-1) + T(h-2) + 1, \quad h \geq 2 \end{cases}$$

   Prove by induction the relation $T(h) = F(h+2) - 1$, where $F(n)$ is the n-th Fibonacci number ($F(1) = 1$, $F(2) = 1$, and $F(n) = F(n-1) + F(n-2)$, for $n \geq 3$).

4. (2.30) A **full binary tree** is defined inductively as follows. A full binary tree of height 0 consists of 1 node which is the root. A full binary tree of height $h+1$ consists of two full binary trees of height h whose roots are connected to a new root. Let T be a full binary tree of height h. The **height** of a node in T is h minus the node's distance from the root (e.g., the root has height h, whereas a leaf has height 0). Prove that the sum of the heights of all the nodes in T is $2^{h+1} - h - 2$.

5. Consider the following algorithm for computing the square of the input number, which is assumed to be a positive integer.

```
Algorithm mySquare(n);
begin
    // assume that n > 0
    x := n;
    y := 0;
    while x > 0 do
        y := y + 2 × x − 1;
        x := x − 1
    od;
    mySquare := y
end
```

State a suitable loop invariant for the main loop and prove its correctness.