Homework Assignment #1

Due Time/Date

2:20PM Tuesday, September 12, 2023. Late submission will be penalized by 20% for each working day overdue. Those who enroll late may be allowed an extension upon request.

How to Submit

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. 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. The Harmonic series $H(k)$ is defined by $H(k) = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k-1} + \frac{1}{k}$. Prove that $H(2^n) \geq 1 + \frac{n}{2}$, for all $n \geq 0$ (which implies that $H(k)$ diverges).

2. Consider *proper* binary trees, where every internal (non-leaf) node has two children. For any such tree T, let l_T denote the number of its leaves and m_T the number of its internal nodes. Prove *by induction* that $l_T = m_T + 1$.

3. (2.7) Given a set of $n+1$ numbers out of the first $2n$ (starting from 1) natural numbers 1, 2, 3, ..., $2n$, prove that there are two numbers in the set, one of which divides the other.

4. (2.37) Consider the recurrence relation for Fibonacci numbers $F(n) = F(n-1)+F(n-2)$. Without solving this recurrence, compare $F(n)$ to $G(n)$ defined by the recurrence $G(n) = G(n-1)+G(n-2)+1$. It seems obvious that $G(n) > F(n)$ (because of the extra 1). Yet the following is a seemingly valid proof (by induction) that $G(n) = F(n) - 1$. We assume, by induction, that $G(k) = F(k)-1$ for all k such that $1 \leq k \leq n$, and we consider $G(n+1)$:

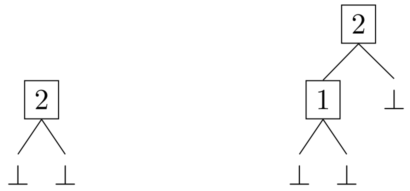
$$G(n+1) = G(n) + G(n-1) + 1 = F(n) - 1 + F(n-1) - 1 + 1 = F(n+1) - 1$$

   What is wrong with this proof?

5. The set of all binary trees that store non-negative integer key values may be defined inductively as follows.

   - The empty tree, denoted $\perp$, is a binary tree, storing no key value.
   - If t_l and t_r are binary trees, then $node(k, t_l, t_r)$, where $k \in \mathbb{Z}$ and $k \geq 0$, is a also binary tree with the root storing key value k.

   So, for instance, $node(2, \perp, \perp)$ is a single-node binary tree storing key value 2 and $node(2, node(1, \perp, \perp), \perp)$ is a binary tree with two nodes — the root and its left child, storing key values 2 and 1, repsectively. Pictorially, they may be depicted as below.



$node(2, \perp, \perp)$ $\qquad$ $node(2, node(1, \perp, \perp), \perp)$

(a) (10 points) Define inductively a function *Max* that determines the largest of all key values of a binary tree. Let $Max(\perp) = 0$, though the empty tree does not store any key value. (Note: use the usual mathematical notations; do not write a computer program.)

(b) (10 points) Suppose, to differentiate the empty tree from a non-empty tree whose largest key value happens to be 0, we require that $Max(\perp) = -1$. Give another definition for *Max* that meets this requirement; again, induction should be used somewhere in the definition.