Fundamental Algorithms, Section 003
Homework 2, Fall 22. Due date: Wednesday, September 21, 4:55pm on Gradescope

1. Consider a Tower of Hanoi with Disordered Start problem with 5 rings. Recall that Ring i has radius i. We write R_i to refer to Ring i. Suppose that initially R_5 and R_1 are on Pole A, R_4 is on Pole B, and R_3 and R_2 are on Pole C. What is the first move if the goal is to move all the rings to Pole B (using the algorithm given in the lecture notes). Justify your answer briefly.

2. Consider the Max Tower of Hanoi problem. The input is a 3 pole Tower of Hanoi problem with n rings on pole A initially. The task is to give a recursive algorithm that causes the rings to go through each possible configuration exactly once, ending with all n rings on pole C.
Hint. Let MaxToH(n, A, B, C) be the recursive procedure that starts with all n rings on pole A and ends with all n rings on pole C, and causes the rings to go through each possible configuration exactly once. First give the procedure for the case $n = 1$. Then using the procedure for MaxToH$(n-1, X, Y, Z)$ for suitable values of X, Y, Z as a subroutine give the algorithm for MaxToH(n, A, B, C). It helps to pay attention to the inductive property that specifies the configurations that MaxToH$(n-1, X, Y, Z)$ goes though.

3. Let T be a tree. Compute the total edge length of all root to leaf paths.
Hint. What information do you need to compute at each node v?

4. Suppose you are given a tree T in which each node is colored blue or red. A path is all-blue if all its vertices are blue. For each node v, determine the edge-length of the longest all-blue path contained in the subtree rooted at v, storing the result in $v.allb$; note that the longest all-blue path does not have to include vertex v. v's color is stored in the field $v.clr$.
Hint. You will need to use an additional field which you need to specify. What is the answer if $v.clr$ = Red? What answers are possible if $v.clr$ = Blue?

Challenge problem. Do not submit.
Let T be a binary tree in which each node v stores a value in field $v.val$. Recall the notion of predecessor given in class.

Give a one-pass recursive algorithm that rotates the values in T. Specifically, for each node other than the leftmost, the value $v.val$ is to be copied to Pred$(v).val$, where Pred(v) denotes the predecessor of v. Finally, the value in the leftmost node is to be copied to the value field of the rightmost node. Note that there is no need to compute the predecessor links.

Again, please begin your answer by specifying what is being stored in each field you create.
Hint. Consider the structure shown for adding predecessor links shown in class (and the lecture notes). What information needs to be passed in and out of v's subtree? What updates should be made in the recursive call for node v when v does and does not have a left subtree and when v does and does not have a right subtree?

Finally, how (or when) can the update to the rightmost node be handled?