ICS621 Homework 4: AVL Trees *or* Josephus Permutation

Choose one of the following.

Problem 13-3 from CLRS. An **AVL tree** is a binary search tree that is height balanced: for each node x, the heights of the left and right subtrees of x differ by at most 1. To implement an AVL tree, we maintain an extra attribute in each node: $x.h$ is the height of node x. As for any other binary search tree T, we assume that $T.root$ points to the root node.

a) Prove that an AVL tree with n nodes has height $O(\lg n)$. (*Hint:* Prove that an AVL tree of height h has at least F_h nodes, where F_h is the hth Fibonacci number.)

b) To insert into an AVL tree, we first place a node into the appropriate place in binary search tree order. Afterward, the tree might no longer be height balanced. Specifically, the heights of the left and right children of some node might differ by 2. Describe a procedure BALANCE(x), which takes a subtree rooted at x whose left and right children are height balanced and have heights that differ by at most 2, i.e., $|x.right.h - x.left.h| \leq 2$, and alters the subtree rooted at x to be height balanced. (*Hint:* Use rotations.)

c) Using part (b), describe a recursive procedure AVL-INSERT(x, z) that takes a node x within an AVL tree and a newly created node z (whose key has already been filled in), and adds z to the subtree rooted at x, maintaining the property that x is the root of an AVL tree. As in TREE-INSERT from Section 12.3, assume that $z.key$ has already been filled in and that $z.left = \text{NIL}$ and $z.right = \text{NIL}$; also assume that $z.h = 0$. Thus, to insert the node z into the AVL tree T, we call AVL-INSERT($T.root, z$).

d) Show that AVL-INSERT, run on an n-node AVL tree, takes $O(\lg n)$ time and performs $O(1)$ rotations.

Problem 14-2 from CLRS. We define the **Josephus problem** as follows. Suppose that n people form a circle and that we are given a positive integer $m \leq n$. Beginning with a designated first person, we proceed around the circle, removing every mth person. After each person is removed, counting continues around the circle that remains. This process continues until we have removed all n people. The order in which the people are removed from the circle defines the (n, m)-**Josephus permutation** of the integers $1, 2, \ldots, n$. For example, the $(7, 3)$-Josephus permutation is $\langle 3, 6, 2, 7, 5, 1, 4 \rangle$.

a) Suppose that m is a constant. Describe an $O(n)$-time algorithm that, given integer n, outputs the (n, m)-Josephus permutation.

b) Suppose that m is not a constant. Describe an $O(n \lg n)$-time algorithm that, given integers n and m, outpus the (n, m)-Josephus permutation.