CSC C78F 1999
Assignment 2

due: Friday, October 29, 1999 at 4:10pm in Nick's office.

1. Binary search trees with equal keys.

   Equal keys can pose a problem for the implementation of binary search trees.

   (a) What is the asymptotic performance of the TREE-INSERT algorithm on page 251 of CLR when used to insert n items with identical keys into an initially empty binary search tree?

   We propose to improve the algorithm by testing before line 5 whether or not $key[z] = key[x]$ and by testing before line 11 whether or not $key[z] = key[y]$. If equality holds we implement one of the following strategies. For each strategy, find the asymptotic performance of inserting n items with identical keys into an initially empty binary search tree. (We just give the strategies for line 5, for line 11 replace x by y)

   (b) Keep a boolean flag $b[x]$ at node x, and set x to either $left[x]$ or $right[x]$ based on the value of $b[x]$, which alternates between FALSE and TRUE each time the node is visited during TREE-INSERT.

   (c) Keep a list of nodes with equal keys at x, and insert z into the list.

   (d) Randomly set x to either $left[x]$ or $right[x]$. (Give the worst-case performance and informally derive the average case performance).

2. Josephus permutations

   The Josephus problem is defined as follows. Suppose that n people are arranged in 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 continues until all n people have been removed. The order in which the people are removed from the circle defines the (n, m)**-Josephus permutation** of the integers $1, \ldots, n$. For example, the (7,2)-Josephus permutation is $\langle 2, 4, 6, 1, 5, 3, 7 \rangle$.

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

3. Consider a hash table of size $m = 100$ and the hash function $h(k) = \lfloor m(kA \bmod 1) \rfloor$ for $A = (\sqrt{5} - 1)/2$. Compute the locations to which the keys 61, 62, 63, 64, and 65 are mapped.

4. Let m be a prime and let U be the set of sequences of length $r+1$ with elements from $\{0, 1, 2, \ldots, m-1\}$. For every sequence $a = \langle a_0, a_1, \ldots, a_r \rangle \in U$ we can define a hash function h_a as follows. For every $x = \langle x_0, x_1, \ldots, x_r \rangle \in U$ let

$$h_a(x) = \sum_{i=0}^{r} a_i x_i \bmod m.$$

Show that the set of all hash functions h_a such that $a \in U$ and all $a_i > 0$ is **not** universal.

5. Give an algorithm for the following problem. The input is a sequence of n numbers $x_1, x_2, \ldots, x_n$, another sequence of n numbers $y_1, y_2, \ldots, y_n$, and a number z. Your algorithm should determine whether or not $z \in \{x_i + y_j \mid 1 \leq i, j \leq n\}$. You should use universal hashing families, and your algorithm should run in expected time $O(n)$. Be very clear about which theorems from class and/or text you are using, and how.