G22.1170: Fundamental Algorithms I
Problem Set 4
(Due Wednesday, April, 26 2006)

The problems in this problem set are about order statistics and data structures, and Graph Algorithms. Please consult Chapters 10, 23 & 24 from the book (CLR).

Problems from Cormen, Leiserson and Rivest:

10-2 (a,b & c) *Weighted Median* (pp. 193)

23.4-5 *Different Topological Sort* (pp. 488)

10-2 Weighted median

For n distinct elements $x_1, x_2, \ldots, x_n$ with positive weights $w_1, w_2, \ldots, w_n$ such that $\sum_{i=1}^{n} w_i = 1$, the **weighted median** is the element x_k satisfying

$$\sum_{x_i < x_k} \leq \frac{1}{2}$$

and

$$\sum_{x_i > x_k} \leq \frac{1}{2}.$$

a. Argue that the median of $x_1, x_2, \ldots, x_n$ is the weighted median of the x_i with weights $w_i = 1/n$ for $i = 1, 2, \ldots, n$.

b. Show how to compute the weighted median of n elements in $O(n \lg n)$ worst-case time using sorting.

c. Show how to compute the weighted median in $\Theta(n)$ worst-case time using a linear-time median algorithm such as Select from Section 10.3.

The **post-office location problem** is defined as follows. We are given n points $p_1, p_2, \ldots, p_n$ with associated weights $w_1, w_2, \ldots, w_n$. We wish to find a point p (not necessarily one of the input points) that minimizes the sum $\sum_{i=1}^{n} w_i d(p, p_i)$, where $d(a, b)$ is the distance between points a and b.

d. Argue that the weighted median is a best solution for the 1-dimensional post-office location problem, in which points are simply real numbers and the distance between points a and b is $d(a, b) = |a - b|$.

e. Find the best solution for the 2-dimensional post-office location problem, in which the points are (x, y) coordinate pairs and the distance between points $a = (x_1, y_1)$ and $b = (x_2, y_2)$ is the Manhattan distance: $d(a, b) = |x_1 - x_2| + |y_1 - y_2|$.

23.4-5 Different Topological Sort

Another way to perform topological sorting on a directed acyclic graph $G = (V, E)$ is to repeatedly find a vertex of in-degree 0, output it, and remove it and all of its outgoing edges from the graph. Explain how to implement this idea so that it runs in time $O(V + E)$. What happens to this algorithm if G has cycles?

Problem 4.1 The input is a sequence of n elements x_1, x_2, …, x_n that we can read *sequentially*. We want to use a memory that can only store $O(k)$ elements at a time. Give a high level description of an algorithm that finds the kth *smallest element* in $O(n)$ time.

Problem 4.2 Let L be a sequence of n elements. If x and y are pointers into list L then $\textsc{Insert}(x)$ inserts a new element immediately to the right of x, $\textsc{Delete}(x)$ deletes the element to which x points and $\textsc{Order}(x, y)$ returns true if x is before y in the list. Show how to implement all three operations with worst case time $O(\log n)$.