Problem 1. (a) Assume you have an alphabet of letters from "o" to "u". Illustrate the operation of radix sort on the following list of English words: *tote, soup, soot, pout, toot, sups, tour, opts, rout, tors.*

(b) Use the words "sup" and "tor" in one English sentence that shows that you understand the meaning of both words.

Problem 2. The Selection (*Blum-Floyd-Tarjan-Rivest-Pratt*) algorithm to find the kth smallest value in a list, described in the class (and in the book), uses columns of size 5. Assume you implement the same selection algorithm using columns of size 7, rather than 5.

(a) Exactly how far from either end of the array is the median of medians guaranteed to be. Just give the high order term. (Recall that with columns of size 5 we got $\frac{3n}{10}$.)

(b) Write down the recurrence for a Selection algorithm based on columns with 7 elements each, using (the full) bubble sort to find the median of each column. (You can ignore floors and ceilings, as we did in class.) You do not have to give the algorithm, but state where each of the terms in your recurrence comes from. (For example, you might say that the n - 1 term comes from partition.)

(c) Solve the recurrence, and give the high order term exactly.

(d) How does this value compare with what we got in class for columns of size 5?

Problem 3. Let $G = (V, E)$ be a directed graph.

(a) Assuming that G is represented by a 2-dimensional adjacency matrix $A[1..n, 1..n]$, give a $\Theta(n^2)$-time algorithm to compute the adjacency list representation of G, with $A[i, j]$ representing an edge between i and j vertices. (Represent the addition of an element(vertex), v, to an adjacency list, l, using pseudo code, $l \leftarrow l \cup \{v\}$.)

(b) Assuming that G is represented by an adjacency list Adj$[1..n]$, give a $\Theta(n^2)$-time algorithm to compute the 2-dimensional adjacency matrix representation of G.