CPSC 320 Problem Set 3 9 October 2009

W. Evans Due: 16 October 2009 at the start of class.

1. Let $A[1 \ldots n]$ be an array of n elements and $k \leq n$ an integer. We want to find an algorithm that outputs the k smallest elements in A (in any order).

   (a) Describe an algorithm that takes $O(n)$ time to solve this problem.

   (b) Use the decision tree lower bound technique to argue that any algorithm must make at least $k \lg(n/k)$ comparisons in the worst case to output the k smallest elements. (Why is $\binom{n}{k} \geq (n/k)^k$?)

   Is this a good lower bound?

2. Suppose we are given a list of n people who want to fly to the moon (and return). Person i has priority p_i and they weigh w_i pounds. You may assume all of the priorities are different. Our spaceship can carry at most k pounds which is, unfortunately, smaller than the total weight of all the people. If priority $p_i > p_j$ then if we take person j we must take person i.

   Describe an algorithm that runs in $O(n)$ time that finds the largest set of people we can fly to the moon without exceeding the weight limit. In other words, we are looking for the priority p_m of a person m such that

$$\sum_{p_i > p_m} w_i \leq k \qquad \text{and} \qquad \sum_{p_i \geq p_m} w_i > k$$

   We fly everyone with priority greater than p_m. Notice that if everyone weighed 1 pound, this would be the k-select problem.

   Hint: Use recursion. How can you use the linear time k-select algorithm to insure that the subproblems are small?