CPSC 320 Problem Set 4 16 October 2009

W. Evans Due: 23 October 2009 at the start of class.

1. You are given n elements and an integer k such that $1 \leq k \leq n$. The problem is to find *any one* of the k smallest elements. For example, if $k = 3$, the output may be the first-, second-, or third-smallest element.

   (a) Give a fast algorithm to solve this problem. How many comparisons does your algorithm perform?

   Hint: Don't look for something complicated. One insight gives a short, simple algorithm.

   (b) Give a lower bound, as a function of n and k, on the number of comparisons needed to solve this problem. Try to find a lower bound that matches the number of comparisons made by your algorithm exactly.

2. Consider the problem of determining if a bit string of length n contains two consecutive 0's. The basic operation is to examine a position in the string to see if it is a 0 or a 1. For each $n = 2, 3, 4, 5$ either give an adversary strategy to force any algorithm to examine every bit (in other words, describe how to provide an input to any algorithm "on-demand" that forces the algorithm to examine every bit), or give an algorithm that solves the problem by examining fewer than n bits.

   Extra credit (hard): For what integers n must any algorithm that solves this problem examine every bit of a (worst case) bit string of length n? Why?

3. A set of bit strings is called a *prefix code* if none of the strings is a prefix of another string. For example, the four strings 00, 01, 10, and 11 form a prefix code, as do 001, 1110, 101001, 0001; but the strings 001, 1110, 0011, 0001 do not (because 001 is a prefix of 0011). Prove that any prefix code of n bit strings must contain some string with at least $\lg n$ bits.

   Hint: Relate prefix codes to binary trees, and number of bits to depth.

4. Suppose A and B are two arrays, each with n elements sorted in ascending order. You may assume that all elements are distinct.

   (a) Devise an $O(\log n)$ algorithm to find the nth smallest of the $2n$ elements.

   (b) Give a asymptotically matching lower bound for this problem.