Summer 2008 **CMSC 351: Homework 7** Clyde Kruskal

Due at the start of class, Friday, June 27.

Problem 1. Consider the bubble-sort algorithm, for an input (a b c) of the three integer values 1, 2, and 3 in any order, where all input orders are equally likely (uniform distribution).

(a) Let M be the number of exchanges. What are E(M), Var(M), and $\sigma(M)$?

(b) Let x be the number of comparisons. What are E(M), Var(M), and $\sigma(M)$?

(c) What percentage of the sample space is within a standard deviation of average.

Problem 2. Again consider the bubble-sort algorithm, for an input (a b c) of the three integer values 1, 2, and 3 in any order. But now suppose the input orders are *not* equally likely; instead, there is a 1/5 chance of $a = 1$ being the first item in the input (the 1/5 distributed equally among those inputs) and 4/5 of a not being 1 (again the 4/5 shared equally among all these inputs).

(a) Let M be the number of exchanges. What are E(M), Var(M), and $\sigma(M)$?

(b) Let x be the number of data comparisons made. What are E(M), Var(M), and $\sigma(M)$?

(c) What percentage of the sample space is within a standard deviation of average.

Problem 3. For this problem you may use a calculator for a few calculations.

Consider $\sum_{k=1}^{100} k^{3/2}$.

(a) Use a non-integral method to show that the sum is between 15,000 and 70,000.

(b) Approximate the sum using integrals. Make sure to get an upper and lower bound.

Problem 4. Assume you have a list of n elements where the first n/k elements are the smallest (but not sorted), the next group of n/k elements are the next smallest (but not sorted), ..., and the last n/k elements are the largest (but not sorted). You may assume k divides n.

(a) Give an algorithm that sorts this list with as few comparisons as possible (as a function of n and k). Just get the high order term right. How many comparisons does your algorithm use?

(b) Show that your algorithm is optimal using a decision tree argument on the *entire* list. (I.e., do not argue that you must solve k independent sorting problems.)