Problem Set 1—Due Friday, 26 June 1999

Problem 1. For each of the following functions $f(n)$ and the following times t, determine the largest size n of a problem that can be solved in time t assuming that the algorithm to solve the problem takes $f(n)$ microseconds. Arrange your answers in a table with rows $f(n) = \{\lg n,\ \sqrt{n},\ n,\ n\lg n,\ n^2,\ n^3,\ 2^n$ and $n!\}$ and columns of t of 1 second, 1 minute, 1 hour, 1 day, 1 month, 1 year, and 1 century.

Problem 2. Let A be an array of n integers such that $A[1] < A[2] < \cdots < A[n]$. Specify an $O(\lg n)$ divide-and-conquer algorithm which either finds an $i \in \{1, \ldots, n\}$ such that $A[i] = i$ or else determines that there is no such i. Justify the correctness and running time of your algorithm.

Problem 3. You are given an array $A = (A[1], \ldots, A[n])$ of n real numbers. You want to determine the maximum and minimum value in A. You do this by comparing elements of A. Assume $n = 2^d$ for some $d \geq 1$.

(a) The "obvious" algorithm makes $2n - 2$ comparisons. Explain.

(b) Carefully specify a more efficient divide-and-conquer algorithm.

(c) Let $T(n) =$ the number of comparisons your algorithm makes when the input array has n elements. Write a recurrence relation for $T(n)$. (Recall that $n \geq 2$ is a power of 2.)

(d) Show that your recurrence relation has as its solution $T(n) = 1.5n - 2$.

Hint: Use induction.

Comment: If your $T(n)$ is larger than this, you need to improve your algorithm. If your $T(n)$ is smaller than this, your algorithm is incorrect.)