CS140: Exam 1 Name ______________________________

10/1/99

This exam consists of 4 problems. The space provided should be adequate for your solutions. Each problem is worth 25 points. Your answers will be judged for clarity, conciseness, and correctness. Bounds and algorithms should be, respectively, as tight and as fast as possible. You may consult any published texts and your class notes to complete this exam. You may not discuss the exam (at all) with anyone except me. The exam is due at the start of class on Monday, October 4, 1999.

1. Describe a monotonically increasing function $f : \mathcal{N} \longrightarrow \mathcal{N}$ that satisfies the following property.

   (a) $f(n)$ is as slow-growing as possible and $f(n+1) \neq O(f(n))$

   (b) $f(n)$ is as fast-growing as possible and $f(n+1) = O(f(n))$

   (c) $f(n)$ is as slow-growing as possible and $f(2n) \neq O(f(n))$

   (d) $f(n)$ is as fast-growing as possible and $f(2n) = O(f(n))$

   (e) $f(n)$ is as slow-growing as possible and $f(n^2) \neq O(f(n))$

   (f) $f(n)$ is as fast-growing as possible and $f(n^2) = O(f(n))$

(g) $f(n)$ is as slow-growing as possible and $f(2^{n+1}) \neq O(f(2^n))$

(h) $f(n)$ is as fast-growing as possible and $f(2^{n+1}) = O(2^n)$

2. $M$ is an $n \times n$ matrix with 0/1 entries. We want to determine if there is an $i$ such that each entry in the $i$-th row is a 0 and each entry in the $i$-th column, except the diagonal entry $M(i, i)$, is a 1. Prove the best upper bound you can on the number of bits of $M$ you need to read to solve this problem.

3. $M$ is an $n \times m$ matrix with distinct integer entries. Each row and column of $M$ is sorted in ascending order. We want to determine if the integer $x$ is an element of the matrix and, if so, its indices. To solve this problems you are given a comparison oracle that takes queries of the form "$x?M(i,j)$" and answers "$x < M(i,j)$," "$x = M(i,j)$," or "$x > M(i,j)$." Prove the best upper bound you can on the number of comparisons needed to solve this problem.

4. As we showed in class, any comparison-based sorting algorithm has a running time that is $\Omega(n \log n)$. This lower bound derives from the fact that any of $n!$ possible permutations may sort the input. If we restrict the input so that some of these permutations are necessarily ruled out, the lower bound may no longer hold. Consider the following examples where the input $S$ is "partially sorted." For each case either prove that the $\Omega(n \log n)$ bound holds despite the restriction or give an $o(n \log n)$ algorithm. (You may assume the elements of $S$ are distinct.)

   (a) The input array $S$ is a min-heap.

   (b) For $x_i \in S$, if $x_i$ is the $j$-th smallest integer in $S$ then $i - \log n \leq j \leq i + \log n$.