CS140: Exam 1 Name ______________________________

10/2/00

This exam consists of 4 problems and 7 pages. Make sure your exam is complete. Write your name above.

Your answers will be judged for clarity, conciseness, and correctness. The space provided should be adequate for your solutions. Bounds and algorithms should be, respectively, as tight and as fast as possible. You may use a one page cheat sheet but no text or other notes during this exam. You may not use a calculator during the exam.

1. (20 points) Order the following functions by increasing rate of growth; i.e. the slowest should be first.

$$\log_3 n^3, \quad \lg^3 n, \quad \sqrt{n}, \quad \sqrt{\lg n}, \quad n^{\frac{1}{4}} \lg n$$

2. (20 points) Identify each of the following statements as either true or false. The functions $f(n), g(n)$ and $h(n)$ used in these problems are monotonically, increasing functions on the natural numbers.

   (a) For every n there is some input of size n for which algorithm $\mathcal{A}$ makes at least cn steps. Thus $\mathcal{A}$ is $\Omega(n)$.

   (b) If $f(n) = O(g(n))$ and $f(n) = \Omega(h(n))$ then $h(n) = O(g(n))$.

   (c) If $f(n) = O(g(n))$ and $f(n) = \Omega(h(n))$ then $h(n) = o(g(n))$.

   (d) If $f(n) = 2^n + n \log n$ then $f(2n) = O(f(n))$.

   (e) If $\log(f(n)) = O(\log(g(n)))$ then $f(n) = O(g(n))$.

3. (30 points) A binary vector $B = b_0 b_1 \dots b_{n-1}$ is an *almost alternating* vector if there is exactly one i such that $b_i = b_{i+1}$. The index i is called the flip index. For example 11, 10110, and 1010100 are almost alternating vectors. The respective flip indices are 0, 2, and 5. (Note the flip index cannot be $n-1$.) Consider the following problem:

   Input: An almost alternating binary vector B.

   Output: The flip index of b.

   Assume that an algorithm for this problem can access the input through queries of the form "Is $b_i = b_j$?".

   (a) (10 points) Let S be a subset of $\{0, \dots, n-1$. Suppose you know that S contains the flip index. Which elements of S can be ruled out by the query "Is $b_i = b_j$?" (assume $i < j$) when:

      i. $j - i$ is even and the answer is yes:

      ii. $j - i$ is even and the answer is no:

      iii. $j - i$ is odd and the answer is yes:

      iv. $j - i$ is odd and the answer is no:

   (b) (10 points) Let $T(n)$ denote the worst case number of queries an algorithm must make in order to solve the problem. Give the best upper bound you can for $T(n)$. Prove your answer. Note: Your bound should be exact; i.e. don't use big-O notation. If you find it convenient, you may assume that $n = 2^m + 1$.

(c) (10 points) Give the best lower bound you can for $T(n)$. Prove your answer. Note: Your bound should be exact; i.e. don't use big-O notation. If you find it convenient, you may assume that $n = 2^m + 1$.

4. (30 points) Algorithm $\mathcal{A}$ is recursive. On an input size of 1 or less the algorithm does a constant amount of work. For input size of $n > 1$, the algorithm makes two recursive calls; one on size $\frac{n}{3}$ and the other on size $\frac{2n}{3}$. (Algorithm $\mathcal{A}$ is able to work on fractional input sizes so we don't have to worry about floors and ceilings.) After the recursive calls return, $\mathcal{A}$ does an additional cn steps to compute its output.

   (a) (5 points) Draw the work tree for the case when $n = (\frac{3}{2})^2 = \frac{9}{4}$. Use fractional input sizes and recall that the algorithm halts if the input is 1 or less. (You should notice that some root-to-leaf paths in the work tree are shorter than others.)

(b) (8 points) Now consider the work tree for $n = (\frac{3}{2})^m$, where $m \geq 2$. Let P denote the shortest root-to leaf path in this tree. Let R denote the longest path.

i. What are the input sizes associated with the nodes along P, starting at the root?

ii. Which level contains the leaf of P? (The root is at level 0.)

iii. What are the input sizes associated with the nodes along R, starting at the root?

iv. Which level contains the leaf of R?

(c) (7 points) Let ℓ denote the level containing the leaf of path P. Prove that for any level j, where $0 \leq j \leq \ell$, the total work done by the nodes at level j is cn. (Hint: Use induction on j).

(d) (5 points) Using big-O notation, give the best upper bound you can for the running time of $\mathcal{A}$.

(e) (5 points) Prove your upper bound.