Algorithms
Computer Science 140 & Mathematics 168
Fall 2003
Instructor: B Thom
Homework 1b
Due Tuesday, Sept. 9

1. [**10 points**] **Growth of Functions.** In this problem you will be doing two proofs by induction. Please write your proofs clearly, with a basis, induction hypothesis, and induction step.

   (a) Use induction to show that $n^2 \geq 2n + 1$ for $n \geq 3$.

   (b) Now use induction to show that $2^n \geq n^2$ for $n \geq 5$. (Using the result from part (a) of this problem will be most useful in doing this proof correctly!)

2. [**10 points**] State whether each of the following statements is true or false and then carefully **prove** your answer using the formal definition of Big-Oh notation and properties of logarithms from Homework 1a. Remember that to show that one of these statements is false, you must obtain a formal contradiction; it does not suffice to just say "false".

   (a) $2^{n+1} = O(2^n)$.

   (b) $2^{2n} = O(2^n)$.

   (c) $3n^2 \log_2 n + 16n = O(n^3)$.

   (d) $25 \log_2 8n^{10} = O(\log_{10} n)$.

   (e) $4^{\log_2 n} = O(n^3)$.

3. [**15 Points**] **Ranking Functions!** List the functions below in increasing order by placing one function on each line, with the top line containing the smallest function and the bottom line containing the largest function. If two functions are in the same group (functions $f(n)$ and $g(n)$ are in the same group if $f(n) \in O(g(n))$ and $g(n) \in O(f(n))$ then write those two functions together on the same line. (You should assume, that the size of the problem, n, is an integer in all cases.) In this problem, we are not asking for proofs (that would take a long time!), just the correct ranking.

$\sqrt{n}$	n^2	$(\sqrt{2})^{\log_2 n}$	$(\frac{3}{2})^n$
$4^{\log_2 n}$	$n \log_{10} n$	$n^{1/3} + \log_2 n$	1.0001^n
2^n	42	$\log_2 n^2$	$\log_2 4^n$
e^n	$\log_2 n!$	$(\log_2 n)^2$	n^e

4. [**15 points**] Page 27, Exercise 2.2-2. (*Note:* For pseudocode, you can use the book's conventions or code similar to that given in class. The syntax isn't important, as long as it's clear).

5. **[25 Points] The Stock Market Problem!** You've been hired as Director of Algorithm Design by the brokerage firm of Weil, Proffet, and Howe. Periodically, the firm looks at how a particular stock has done in the last n days and wishes to know what the maximum possible profit would have been if someone had bought and sold a share during those n days. More formally, let A be an array of n numbers where $A[i]$ represents the value of a share in the stock on day i. We wish to find

$$\max_{1 \leq j < k \leq n} A[k] - A[j]$$

The interpretation of this expression is that on day j we could buy a share for $A[j]$ dollars and at a later date, k, we could sell the share for $A[k]$ dollars, making a profit of $A[k] - A[j]$ dollars. It's critical, of course, that $j < k$ since one can only sell a share *after* they have purchased it!

   (a) Describe a simple algorithm for solving this problem (that is, determining the maximum profit) and show that this algorithm has time complexity $\Theta(n^2)$.

   (b) Describe a recursive divide-and-conquer algorithm for this problem.

   (c) Write the recurrence relation for the running time of your divide-and-conquer algorithm and explain how you got it.

   (d) Use the recursion tree method (see the example of Mergesort that we did in class) to find a closed-form formula for the running time of your algorithm and express this in Θ notation. Show each step of your analysis very carefully. How does this algorithm compare to the simple $\Theta(n^2)$ algorithm?

6. **[25 Point OPTIONAL BONUS PROBLEM] The Diogenes Problem! It's fun and challenging!** Problem 4-6 (page 87) parts (b) and (c). (*Note:* The notation $\lfloor x \rfloor$ is pronounced "the floor of x" and represents the largest integer that is less than or equal to x.) Who was Diogenes and what is he doing in this problem?