Algorithms, Winter 2010-11, Homework 1
due Friday 12/10/10, 4pm

Problem 1

Rank the following functions by order of growth; that is, find an arrangement $g_1(n), g_2(n), \ldots, g_{24}(n)$ of functions satisfying $g_i(n) = O(g_{i+1}(n))$ for every $i \in \{1, \ldots 23\}$. Partition your list into equivalence classes such that $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n))$. You do not have to prove your answers.

$$\begin{array}{cccccccc}
n \log n & n^{2/3} & n^{3/2} & n^{1/\log n} & \log n & 1 & (\log n)^{\log n} & 2^{n^2} \\
\log_{10} n & 4^{\log n} & n & 2^n & 2^{n+1} & \log \log n & n^{\log \log n} & n! \\
2^{2n} & 2^{\log n} & \log^2 n & \log(n^2) & \sqrt{2}^{\log n} & \sqrt{\log n} & n2^n & n + n^2/10^{20}
\end{array}$$

Remarks:

- In this class we use $\log n$ to denote the logarithm base 2.
- Use the Stirling's formula to figure out how to rank $n!$. The Stirling's formula is:

$$n! = \sqrt{2\pi n} \left(\frac{n}{e}\right)^n (1 + O(\frac{1}{n}))$$

- Use also this fact: for any constants $b_1, b_2 > 0$:

$$\log^{b_1} n = O(n^{b_2}) \quad \text{and} \quad n^{b_2} \neq O(\log^{b_1} n)$$

  In short, logarithm of n raised to any power grows slower than any power of n.

Problem 2

Given is an array $A[1..n]$ containing integers from the set $\{0, 1, 2, \ldots, n^2 - 1\}$. Give an $O(n)$ algorithm that sorts the array.

Hint: Typically we write numbers in decimal notation. This implies that an integer x needs about $\log_{10} x$ digits (why?). How many digits do we need for numbers from $\{0, 1, 2, \ldots, n^2 - 1\}$ if we write them in base n?

Problem 3

This problem asks you to compare several algorithms for the same problem. The problem has multiple parts:

- Implement MergeSort and a quadratic sorting algorithm of your choice.
- Generate input files for $n = 10$, $n = 100$, $n = 1000$, $n = 10000$ and $n = 100000$ by randomly choosing n numbers from the set $\{0, 1, 2, \ldots, n^2 - 1\}$.
- Run the MergeSort, the quadratic sorting algorithm, and the algorithm for Problem 2 on your inputs. Compare the running times of the three algorithms.