Algorithms, Spring 2012-13, Homework 1
due Wednesday, March 13, 2013, 10am

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.

$n \log n$	$n^{2/3}$	$n^{3/2}$	$n^{1/\log n}$	$\log n$	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}	$2^{\log n}$	$\log^2 n$	$\log(n^2)$	$\sqrt{2}^{\log n}$	$\sqrt{\log n}$	$n2^n$	$n + n^2/10^{20}$

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

Problem 2

Find the tightest possible bound for $T(n)$ that satisfies the following recurrence:

$$T(n) = \begin{cases} 2T(n/4) + O(1) & \text{for } n \geq 4 \\ O(1) & \text{for } n < 4 \end{cases}$$

Use either the "unrolling the recurrence" technique or the "substitution/induction" technique from the textbook/class.

Problem 3

Given is a sequence of n integers integers from the set $\{0, 1, 2, \ldots, n^2 - 1\}$. Design an $O(n)$ algorithm that sorts the sequence.

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?