CSE 101 Homework 2
Due Tuesday, May 13, 2 PM
Divide-and-Conquer
100 points total = 10 %

Is there a triangle? 20 pts Last homework, we gave an algorithm to list all the triangles in an undirected graph in $O(n^2 + nm)$ time, which is $\Theta(n^3)$ when the graph is dense. Show how, given a graph G in matrix representation, it is possible to use Strassen's matrix multiplication algorithm (see MM, Chapter 2.5) to determine if the graph has a triangle in time $o(n^3)$. (You can use the algorithm without description or proof, but you must explain connection to the existence of a triangle in the graph).

Binary Conversion: 20 pts . Last homework, we found an $O(n^2)$ algorithm for converting a decimal integer to binary, where the basic operations involved single digits. Present and analyze a divide-and-conquer algorithm that does better. You will probably need to use the faster integer multiplication algorithm, *prod*2, discussed in pages 75-78 of NN.

Multiplication in threes-20pts. Describe and analyze a divide-and-conquer algorithm for integer multiplication based on partitioning up each input into 3 blocks of digits. Is this better than the divide and conquer algorithm *prod*2 in NN? If so, is it possible to get further improvements along the same lines?

Skylines:(20 points) Present and analyze a divide-and-conquer algorithm for the Skylines problem discussed in class. As a reminder, the problem was as follows. In a video game, a city's buildings are rectangles along an axis, specified as $B_i = (s_i, f_i, h_i)$, for $1 \leq i \leq n$, for a building starting at $x = s_i$ and going to $x = f_i$ at height h_i. When seen from a distance, the graphics is supposed to plot the skyline, which at each vertical point x has height the height of the tallest building B_i with $s_i \leq x \leq f_i$. Give an algorithm to plot the skyline, as a series of points connected by line segments.

Implementation of Skylines:(20 points) Implement the above divide and conquer algorithm for the Skylines problem, and the algorithm using a heap shown in class. Compare the running times (graph of a log-log scale) for different values of n. For each value, generate n random buildings (s, f, h) by picking s randomly between 1 and n^2, pick $f = s + 2^i$, where i is chosen randomly between 0 and $\log n$, and pick h randomly between 1 and n. Plot the average time for several random instances with n all powers of 2 between 2^4 and 2^{12}.