Problem Set 4—Due Friday, October 24

Problem 15.

(a) Let a and b be distinct real numbers. Write a pair of assignment statements (that is, `x = <something>; y = <something>`) equivalent to:

```
if (a<b) {x=a; y=b;}
    else {x=b; y=a;}
```

The right hand side of your assignment statements (what we called "`<something>`" above) may use only the arithmetic operators `+`, `-`, `*`, `/` and functions `sqrt`, `exp`, `ln`. Don't assume anything about the representation of real numbers.

(b) Generalize part (a): show how to sort n distinct real numbers, for any n, in zero comparisons, without assuming anything about the internal representation of the numbers.

(c) Does what you've just done contradict the $\lceil \lg n! \rceil$ lower bound we did for sorting n numbers? Explain. *Comment:* Reflect seriously on this question; a satisfying answer is *not* obvious.

Problem 16. The greedy algorithm for 0/1 KNAPSACK is as follows: starting with the emptyset of items S, repeatedly add to S the item i not yet in S with largest v_i/w_i but subject to constraint that adding i to S will not make S have weight exceeding B. (Notation is as in class: v_i is the value of item i; w_i is the weight of item i; and B is the total weight that can be carried.) Prove that the greedy algorithm does *not* correctly solve 0/1 KNAPSACK.

Problem 17. Consider the following SUBSET SUM problem:

Instance: A set of natural numbers $S = \{s_1, \ldots, s_n\}$ and a number t.

Question: Is there an $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} s_i = t$?

Give the best algorithm you can for SUBSET SUM and analyze its efficiency. Implement your algorithm and show that it works.

Problem 18. Let $x = x_1 \ldots x_n$ and $y = y_1 \ldots y_m$ be strings. We say that $z = z_1 \ldots z_{n+m}$ is a *shuffle* of x and y if z can be formed by interspersing the characters from x and y in a way that maintains the left-to-right ordering of the characters from each string. For example, `thhelerloe` is a shuffle of `hello` and `there`. Devise an algorithm that, given strings x, y and z, determines if z is a shuffle of x and y. Analyze your algorithm's running time. *Hint: the values in the table you construct can be Boolean.*

Problem 19. Prove or disprove: there is a checker for the MEDIAN problem that verifies an answer using fewer comparisons than *any* comparison-based algorithm can find the median.