Problem Set 3—Due Friday, October 18

Instructions: Write up your your solutions as clearly and succinctly as you can. Don't forget to acknowledge anyone with whom you discussed problems.

Problem 3–1 We consider some variations of the determinisitic SELECT algorithm described in class an in your text. For each of the following cases, analyze the running time of the algorithm by writing down a recurrance relation for $T(n)$ —the maximum number of comparisons to select an indicated element among n elements— and solving it (including the leading constant). For simplicity, you may assume the elements are distinct and that any values that arise in your algorithm are divisable by small numbers.

(a) The list is divided into groups of size 3. (Use the fact that we can find the median of 3 elements in 3 comparsions).

(b) The list is divided into groups of size 5. (Use the fact that we can find the median of 5 elements in 6 comparisons.)

(c) The list is divided into groups of size 7. (Use the fact that we can find the median of 7 elements in 10 comparisons.)

(d) The list is divided into groups of size 9. (Use the fact that we can find the median of 9 elements in 14 comparisons.)

(e) Interpret the above findings.

Problem 3–2. Consider the recurrance $T(n) = n-1+\frac{1}{n}\sum_{k=0}^{n-1} T(k)$ which we found in our analysis of RANDOMIZEDSELECT. Prove (by induction) that, for sufficiently large n, $T(n) \le cn$ for some constant c that you determine. Make c as small as you can. Compare your results with those from Problem 3–1 and explain the significance of what you've found.

Problem 3–3. Recall the pancake-flipping problem, Problem 2–5, from the last homework. State and prove the best (worst-case) *lower bound* you can find for the number of flips required to sort the n pancake. *(You are not giving an algorithm here. You are proving that any "spatula-insertion algorithm" takes at least $T(n)$ time, for the $T(n)$ which you specify.)*

Problem 3–4. Consider the following LARGEST BOUNDED-WEIGHT SUBSET problem: given a set T of n (distinct) numbers, $x_1, \ldots, x_n$, and a number B, find a maximal cardinality $S \subseteq T$ for which the sum of the numbers in S is at most B. For example, if $T = \{35, 27, 22, 55, 87, 74\}$ and $B = 100$ then the answer is $\{35, 22, 27\}$, as every other cardinality-3 subset sums to more than 100.

(a) Prove that the set S is not necessarily unique.

(b) Give an $O(n)$-time algorithm which finds a set S as above.