6.046 Spring 2005 Quiz Review—from 6.046 Fall 2004:

1. Recurrences (18 points)
Solve the following recurrences (provide only the $\Theta()$ bounds). You can assume $T(n) = 1$ for $n$ smaller than some constant in all cases. You do not have to provide justifications, just write the solutions.

(a) $T(n) = 3T(n/5) + \lg^2 n$
(b) $T(n) = 2T(n/3) + n \lg n$
(c) $T(n) = T(n/5) + \lg^2 n$
(d) $T(n) = 8T(n/2) + n^3$
(e) $T(n) = 7T(n/2) + n^3$
(f) $T(n) = T(n-2) + \lg n$

2. True or False, and Justify (15 points)
Circle T or F for each of the following statements to indicate whether the statement is true or false, respectively. If the statement is correct, briefly state why. If the statement is wrong, explain why. Your justification is worth more points than your true-or-false designation.

T  F  If $f(n)$ does not belong to the set $o(g(n))$, then $f(n) = \Omega(g(n))$.
T  F  A set of $n$ integers in the range $\{1, 2, \ldots, n\}$ can be sorted by RADIX-SORT in $O(n)$ time by running COUNTING-SORT on each bit of the binary representation.
T  F  An adversary can construct an input of size $n$ to force RANDOMIZED-MEDIAN to run in $\Omega(n^2)$ time.

3. Short Answer (15 points) Give brief, but complete, answers to the following questions.

(a) Consider any priority queue (supporting INSERT and EXTRACT-MAX operations) in the comparison model. Explain why there must exist a sequence of $n$ operations such that at least one operation in the sequence requires $\Omega(n \lg n)$ time to execute.
(b) Suppose that an array $A$ has the property that only adjacent elements might be out of order—i.e., if $i < j$ and $A[i] > A[j]$, then $j = i + 1$. Which of INSERTION-SORT or MERGE-SORT is a better algorithm to sort the elements of $A$? Justify your choice.
(c) Recall from lecture that for $n$-by-$n$ matrices $A$, $B$, and $C$, we can probabilistically test whether $AB = C$ by choosing a random 0-1 vector $x$ and checking whether $ABx = Cx$. In lecture we showed a lower bound of $p \geq 1/2$ on the probability $p$ that this test is successful as follows. For a matrix $M$, let $M_{ij}$ denote the $ij$th element of $M$. We argued that if $(AB)_{ij} \neq C_{ij}$, then at most one of the two values for $x_i$ can make the equation

$$\sum_{k=1}^{n} (AB)_{kj} x_k = \sum_{k=1}^{n} C_{kj} x_k$$

hold.

Suppose that instead we choose a random vector $x$ where each element of $x$ is chosen independently at random from the set $\{0, 1, 2, 3\}$ instead of from $\{0, 1\}$, and perform the same test. What lower bound on $p$ can be deduced in this case?

4. Anagram pattern matching (20 points)
Assume you are given a text array $T[1 \ldots n]$ containing letters from the standard Latin alphabet. In other words, $T[i] \in \{a, b, \ldots, z\}$ for $i = 1 \ldots n$. In addition, you are given a pattern array $P[1 \ldots m]$, $m < n$, which also contains letters from the Latin alphabet. For any sub-array $T[i \ldots i+m-1]$ of $T$, we say that $T[i \ldots i+m-1]$ is an anagram of $P$ if there is a way of permuting symbols in $T[i \ldots i+m-1]$ so that the resulting array is equal to $P$.

(a) Give an algorithm that, given an index $i$ (and $m$), determines whether $T[i \ldots i+m]$ is an anagram of $P$. Try to give an algorithm that is as efficient as possible. However, partial credit will also be given for less efficient solutions.
(b) Design an algorithm, which given $T$ and $P$ as an input, reports all $i$'s such that $T_i^m$ is an anagram of $P$. Ideally, your algorithm should run in $O(n + m)$ time. However, partial credit will also be given for less efficient solutions.

5. Mean 6.042 instructors (12 points)

There are two types of professors who have taught 6.042: nice professors and mean professors. The nice professors assign A's to all of their students, and the mean professors assign A's to exactly 75% of their students and B's to the remaining 25% of the students. For example, for $n = 8$, the arrays [A A A B A B A A] and [A B B A A A A A] represent grades assigned by mean professors, and the array [A A A A A A A A] represents grades assigned by a nice professor. Given an array $G[1..n]$ of grades from 6.042, we wish to decide whether the professor who assigned the grades was nice or mean.

Give an efficient randomized algorithm to decide whether a given array $G$ represents grades assigned by a mean or nice professor. Your algorithm should be correct with probability at least 51%.

6. Extra: Finding the smallest elements of an array (20 points)

This question was in problem set 2, except the last part. In this question, we will explore several algorithms for finding the $k$ smallest elements of an array of $n$ integers, in sorted order. For example, given the array [5 2 1 9 6 7 3 4 8] and $k = 3$, such an algorithm should return [1 2 3].

(a) Algorithm A sorts the numbers using merge sort and outputs the first $k$ elements in the sorted order. Analyze the worst-case running time of Algorithm A in terms of $n$ and $k$.

(b) Algorithm B builds a min-heap from the numbers and calls EXTRACT-MIN $k$ times. Analyze the worst-case running time of Algorithm B in terms of $n$ and $k$.

(c) Algorithm C uses SELECT to find the $k$th-largest element in the array, partitions around that number, and then insertion-sorts the $k$ smallest numbers. Analyze the worst-case running time of Algorithm C in terms of $n$ and $k$.

(d) Briefly describe an algorithm that is asymptotically at least as good as Algorithms A, B, and C, and give its running time. You need not argue correctness.