Midterm Examination (1993)

- This exam ends at 10:55 A.M. It contains 5 problems. You have 85 minutes to earn 100 points.
- This exam is closed book. You may use one $8\frac{1}{2}'' \times 11''$ crib sheet. When the exam is over, staple your crib sheet to the back of the exam.
- Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem. Do not put part of the answer to one problem on the back of the sheet for another problem.
- Do not spend too much time on any problem. Read them all through first and attack them in the order that allows you to make the most progress.
- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Be neat.

Problem	Points	Grade
1	15	
2	25	
3	20	
4	20	
5	20	
Total	100	

Name: ______________________________

Have you taken cs109A? ____________ cs109B? ____________

Problem 1. [15 points]

(a) Prove the following statement or give a counterexample: If $f(n) = O(g(n))$ then $2^{f(n)} = O(2^{g(n)})$.

(b) Prove the following statement or give a counterexample: If $f(n) = O(g(n))$ then $f(n)^2 = O(g(n)^2)$.

Give asymptotically tight (Θ) bounds for $T(n)$ in each of the following recurrences. Justify your solutions.

(c) $T(n) = T(n/2) + \sqrt{n}$.

(d) $T(n) = T(n/4) + T(3n/4) + n$.

Problem 2. [25 points]

Let $a_1, \ldots, a_n$ be distinct integers. Let b_i be the ith smallest integer among $a_1, \ldots, a_n$, let $c_j = b_{2^j}$ $(j = 0, \ldots, \lfloor \log_2 n \rfloor)$ and let

$$C = \sum_{j=0}^{\lfloor \log_2 n \rfloor} c_j \qquad\qquad (= b_1 + b_2 + b_4 + b_8 + b_{16} + \ldots + b_{2^{\lfloor \log_2 n \rfloor}}.)$$

(a) Assuming that n is even, how much time does it take to find $b_{n/2}$?

(b) Assuming that n is a power of 2, describe an algorithm which computes C. Your algorithm should be asymptotically faster than the algorithm which simply computes all c_j's and then adds them up. You may rely on algorithms described in lectures or in the book.

(c) Give a recurrence that describes the worst-case running time of your procedure.

(d) Solve the recurrence and prove a matching upper and lower bound by substitution.

(e) How can you use your algorithm to handle the case when n is not necessarily a power of 2?

Problem 3. [20 points]

(a) We would like to build a dictionary, but we don't know in advance the number of keys n it will contain. We start with a hash table with $m = 2$ slots. Whenever the number of keys n in our dictionary equals the number of slots m in the hash table, we simply create a new hash table with twice as many slots and we reinsert all the keys we had previously inserted. Under the assumption of simple uniform hashing, what is the expected time of inserting n keys in the dictionary? Justify your answer.

(b) Let $\mathcal{H}$ be the universal collection $\{h_a\}$ studied in the text, defined by

$$h_a(x) = a_0 x_0 + a_1 x_1 \pmod{m}$$

where $a =< a_0, a_1 >$ and $x =< x_0, x_1 >$. Let $m = 17$ and let $x =< 5, 10 >$, $y =< 6, 12 >$, and $z =< 3, 6 >$.
What is the probability that $h_a(x)$ differs from $h_a(y)$?

What is the probability that $h_a(x), h_a(y)$ and $h_a(z)$ are all equal?

Problem 4. [20 points]

(a) Which of the following algorithms sort in place? INSERTION-SORT?, MERGE-SORT?, HEAP-SORT?

(b) Give a lower bound on the worst-case running time of any comparison based algorithm which need not correctly sort a given input, but correctly sorts at least $\frac{1}{2^n}$ of all permutations on n inputs.

(c) An inversion in the sequence $< a_1, \ldots, a_n >$ is a pair of elements (a_i, a_j) with $i < j$ but $a_i > a_j$. What is the maximum possible number of inversions and what input provides it?

(d) Describe a divide-and-conquer algorithm for computing the total number of inversions in $< a_1, \ldots, a_n >$ in $O(n \lg n)$ time (Hint: You may want to design an algorithm similar to MERGE-SORT).

Problem 5. [20 points]

(a) Describe a procedure to find the second smallest element in a binary search tree. What is its running time?

(b) Draw the binary search tree obtained by inserting the elements $7, 4, 5, 9, 3, 8, 1, 11$ (in that order) using TREE-INSERT in an empty binary search tree. You don't need to show the intermediate trees.

(c) Using TREE-DELETE, delete the node containing key 9 and draw the resulting binary search tree.

(d) Can the nodes of the tree from part (c) be labeled `r` or `b` such that the tree becomes a legal red-black tree? If yes, give the labeling (you can put the labeling directly on the above tree); otherwise, draw a legal red-black tree containing the same elements (i.e. $7, 4, 5, 3, 8, 1, 11$).