CS109A Final Examination
Monday, March 20, 1995, 12:15–3:15PM

Directions

The exam is *open book*; any written materials may be used. Answer all 15 questions on the exam paper itself. The total number of points is 300.

If you do not wish to do heavy arithmetic, you may leave answers as a product of integers or as a quotient of products of integers. You **may not** leave answers in terms of factorials, permutations (Π), combinations $\binom{n}{m}$, or other formulas. Also, you must **cancel common factors** from numerator and denominator.

You may make additional comments at the bottoms of pages or at the end if you feel the need to clarify or justify your answers.

Do not forget to **sign the pledge** below.

I acknowledge and accept the honor code.

____

Print your name here: ____

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Problem 1: (20 points) Below is a function f that computes a certain sum on a nonempty integer list. It returns the lone element on a list of length 1, and on lists of greater length calls itself recursively on two lists formed by deleting either the first or second element, summing the results.

```
fun f([x:int]) = x
|   f(x::y::zs) = f(x::zs) + f(y::zs);
```

Let $T(n)$ be the running time of $f(L)$, where n is the length of the list L that is the argument of f. We wish to write a recurrence relation for T.

a) Give an appropriate basis rule (formula) below. You may use appropriate big-oh expressions for unknown running times.

____

b) Give the inductive formula for $T(n)$ below, in terms of big-oh expressions and $T(i)$ for one or more values of $i < n$.

---

c) What is the value returned by $f([1, 2, 3])$? ________

Problem 2: (15 points) Let us consider the behavior of f from Problem 1 on a list of n 1's, that is, let $L_n = [1, 1, \ldots, 1]$ (n 1's in the list). We claim that

$S(n)$: $f(L_n)$ has the value 2^{n-1}

You must prove $S(n)$ by induction on n.

a) What is a suitable basis case? ________

b) Prove the basis case:

---

c) Prove the inductive step:

Problem 3: (15 points) Below is a fragment of C code.

```
(1) for(i=0; i<n; i++) {
(2)     if(A[i]>0)
(3)         for(j=0; j<n; j++)
(4)             B[j] += A[i];
(5)     else A[i] = 0;
    }
```

a) In the space below, draw the structure tree for this fragment. Indicate the lines comprising each node of your tree.

b) Using n as the measure of "input size," give simple, tight big-oh bounds on the running time for each node of your tree. These bounds should be indicated near each node of your tree.

Problem 4: (20 points) Consider the recurrence

BASIS: $T(1) = 1$.

INDUCTION: $T(n) = n^2 + 2T(n/2)$ for integers $n > 1$.

Note that since there are no symbolic constants, this is not a "big-oh" problem; we can solve for $T(n)$ exactly. However, $T(n)$ is only defined when n is a power of 2, so we shall assume n is a power of 2 throughout this problem.

a) To begin, what are the initial values of $T(n)$?

$T(1) =$ ______; $T(2) =$ ______; $T(4) =$ ______; $T(8) =$ ______; $T(16) =$ ______.

b) Expand the inductive rule so $T(n)$ is expressed in terms of $T(n/4)$.

---

c) Express $T(n)$ in terms of $T(n/8)$.

---

d) What is the general pattern? That is, express $T(n)$ in terms of $T(n/2^i)$.

---

e) For what value of i can we eliminate $T(n/2^i)$ from the expression? $i =$________

f) Use your answer to (e) to express $T(n)$ as a function of n alone (i.e., without any terms involving the function T).

$T(n) =$________________.

Problem 5: (20 points) In parts (a) and (b), give witnesses n_0 and c that demonstrate the stated big-oh relationship. *Note*: your witnesses must be minimal; i.e., lowering either while keeping the other fixed must lead to nonwitnesses.

a) $n^2 + 100$ is $O(n^2)$. $n_0 =$ ________; $c =$ ________

b) Let $f(n) = n$ for $n < 100$ and $f(n) = n^2$ for $n \geq 100$. Let $g(n) = n^2$ for $n < 50$ and $g(n) = n$ for $n \geq 50$. Then $g(n)$ is $O\big(f(n)\big)$. $n_0 =$ ________; $c =$ ________

c) It is not true that n^2 is $O(100n)$. To prove this fact, we suppose to the contrary that there are witnesses n_0 and c demonstrating "n^2 is $O(100n)$." We need to select a value of n, say n_1, that leads to a contradiction. Give a formula for n_1 in terms of n_0 and c.

Explain why your choice of n_1 violates the statement "for $n \geq n_0$, $n^2 \leq c(100n)$."

Problem 6: (20 points) Write an ML function *divide* that takes a list L and returns a pair of lists (M, N), where M consists of all those elements of L that are $\leq$ 10, and N consists of all those elements of L that are $>$ 10. *Note*: your function must not use any "helper" functions; e.g., it is not acceptable to write two functions, one that finds M and the other N.

Problem 7: (20 points) The following datatype describes expressions with integer operands and binary operators $+$ and $\times$.

```
datatype exp =
    Op of int |
    Sum of exp * exp |
    Prod of exp * exp;
```

An operand with value i is represented by the expression `Op(i)`. If E_1 and E_2 are expressions, then $(E_1) + (E_2)$ is represented by `Sum(`E_1`,` E_2`)` and $(E_1) \times (E_2)$ is represented by `Prod(`E_1`,` E_2`)`.

a) Write as a value of datatype `exp` the infix expression $(2 + 3) \times (4 + 5)$.

b) Write an ML function `eval(e)` that returns the integer value of an expression e.

Problem 8: (20 points) Consider an ML function `test(f,L)` that takes a function f and a list $L = [a_1, a_2, \ldots, a_n]$, where $n \geq 1$, and produces the list of $n-1$ values $[b_1, b_2, \ldots, b_{n-1}]$ such that for $i = 1, 2, \ldots n-1$, b_i is the result of $f(a_i, a_{i+1})$. You do not have to handle the case where L is empty.

a) Write `test` here.

---

b) Examining `test`, what type would ML find for parameter f?

---

c) What type would ML find for `test`?

Problem 9: (20 points) An *edge* in a tree is one of the lines we use to represent the relationship between a node and its parent. There is exactly one edge for every node in the tree, except for the root.

A *strict-two-tree* (STT) is a tree in which every interior node has exactly two children. (Note a STT is an "ordinary" tree, not a binary tree; in particular there is no notion of an "empty" STT.)

Show the following by structural induction:

$S(T)$: if T is an STT with e edges and n leaves, then $e = 2n - 2$.

Important: There will be no credit for a proof that is not a structural induction.

a) Prove the basis here:

---

b) Prove the inductive step here:

Problem 10: (30 points) Count the following. *Note*: Please see comment on appropriate forms of answer in the exam directions.

a) The number of ways to divide five quarters and three dimes into two piles, a "left" pile and a "right" pile. Assume that the coins of each denomination are indistinguishable and that order within a pile does not matter. ____________

Explain your reasoning for partial credit.

b) The number of ways to assign five distinct tasks to three people. Assignments are distinguishable if any task is assigned to a different person. ____________

Explain your reasoning for partial credit.

c) The number of ways to arrange the integers 1, 2, 3, 4, and 5 in a heap. ______

Explain your reasoning for partial credit.

Problem 11: (20 points) We wish to compute the probability that at least one of seven cards selected from an ordinary, 52-card deck is an Ace. *Note*: Please see comment on appropriate forms of answer in the exam directions.

a) Describe the probability space that you will use for your calculation. (There are several options here, but your choice now determines the correct answer in subsequent parts.)

b) How many points are there in your probability space?

Explain your reasoning for partial credit.

c) How many points are in the event "there is at least one Ace"?

Explain your reasoning for partial credit.

d) What is the probability of this event?

Problem 12: (15 points) Set S has 10 members. A and B are subsets of S and have 7 and 8 members, respectively. Give tight upper and lower bounds on the number of members of the following sets.

a) $A \cup B$. Lower ______ Upper ______

b) $A \cap B$. Lower ______ Upper ______

Problem 13: (20 points) Below we see a sequence of steps used to prove the statement "if x is in $S \cap T$, then x is in $S - (S - T)$."

a) For each, provide a reason, which is either:

1. Given.
2. Definition of "difference."
3. Definition of "intersection."

You may use the numbers 1–3 to indicate your choice. *Note*: part (a) is scored by deducting 2 points for each unanswered part and 4 points for each wrong answer.

(1) x is in $S \cap T$.________
(2) x is in S.________
(3) x is in T.________
(4) x is not in $S - T$.________
(5) x is in $S - (S - T)$.________

b) Now, write your own proof of the converse of (a), that is: "if x is in $S - (S - T)$, then x is in $S \cap T$.

Problem 14: (15 points) We wish to maintain a set S of n integers. For each of the following data structures, give worst- and average-case big-oh running times for the operation *max* of finding the largest integer in the set (without deleting it). *Note*: 5 pts. are deducted for each wrong answer, with a minimum score of 0.

a) A hash table with about n buckets. Worst case: ________ Average case: ________

b) A binary search tree. Worst case: ________ Average case: ________

c) A heap (with $A[i/2] \geq A[i]$ for all i). Worst case: ________ Average case: ________

Problem 15: (30 points) Suppose P is an ML program (collection of ML function definitions). Consider the relation C, whose domain is the set of functions in P, such that fCg if and only if function f calls function g directly (i.e., in the text for f is at least one use of g).

Also, let I be the relation on the domain of functions of P representing "direct or indirect calls." That is, fIg if and only if f calls g, either directly or through one or more other calls. More precisely, we can define I from C by the following inductive definition:

BASIS: If fCg then fIg.

INDUCTION: If fCh and hIg, then fIg.

This problem asks you to tell whether C and I *must* have certain properties regardless of what the program P is and whether they *might* have certain properties, depending on the program P. In addition, we ask you to consider the same questions restricted to the class of programs P that are not recursive (abbreviated "nonrec." in the table below).

Fill out the following table with Y (yes) and N (no). Scoring: deduct 1 point for each blank, 2 points for each wrong answer, minimum of 0.

Property	C, any P	C, nonrec. P	I, any P	I, nonrec. P
Might be symmetric				
Must be symmetric				
Might be transitive				
Must be transitive				
Might be reflexive				
Must be antisymmetric				
Must be a partial order				
Might be a total order				