CS109A Class Quiz
Friday, March 1, 1996, 3:15–4:05PM

Directions

The exam is *open book*; any written materials may be used. Answer all 6 questions on the exam paper itself. The total number of points is 100. 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. Please do not leave answers in an unexpanded form; e.g., write 10 instead of $\binom{5}{2}$ or $5!/(3!2!)$. 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		

Problem 1: (15 points) Solve the linear recurrence $T(n) = \frac{5}{2}T(n-1) - T(n-2)$ with the basis $T(0) = 1$ and $T(1) = 4$. Show your reasoning and calculation.

Problem 2: (15 points) Six Giants fans and five A's fans line up for tickets. We shall assume for this problem that Giants fans are indistinguishable from one another, and A's fans are indistinguishable from one another, so you may think of the line as holding 6 letters `G` and 5 letters `A`. For each of these problems, you may give the answer alone or, if you want to be considered for partial credit, you may give a brief explanation of your answer, giving, e.g., the formula you used and the arithmetic involved.

a) How many different, distinguishable orders of the fans are there? Remember, we assume you cannot distinguish among fans of the same team.

b) In how many of these orders are the first three places held by two Giants fans and one A's fan?

c) In how many of these orders are the last five places held by two Giants fans and three A's fans?

d) In how many of these orders are *neither* two of the first three places held by Giants fans *nor* two of the last five places held by Giants fans?

Problem 3: (20 points) In Fig. 1 is the definition of an ML function `square` that squares a BI (Binary Integer).† It does not work on a BI with a first component of 0; i.e., it will not square `(0,nil)`. However, it correctly squares any other BI.

Function `square` uses certain functions that you wrote in the various projects:

`break(k,x)` breaks BI x into two BI's, one with the first k (low-order) bits in the list portion of x, and the second with the remaining bits.

`shift(m,x)` takes BI x and inserts m 0's at the front of its list portion, returning the resulting BI. This function has the effect of multiplying x by 2^m.

`mult(x,y)` returns a BI that is the product of BI's x and y.

`add(x,y)` returns a BI that is the sum of BI's x and y.

Define the *size* of a BI x to be the length of its list (second) component. Equivalently, the size is the value of the first component; if the BI is properly formed (which you may assume in this problem), these two quantities are the same. Then the running times of the four functions used by `square` are:

† A *Binary Integer* is the type that we have been using in programming assignments 2–5; it consists of an integer n paired with a list L. The intent is that n is the length of L and that L consists of a sequence of 0's and 1's representing some integer in binary, with the low-order bit at the front of the list.

```
(1)  fun square(1,[a]) = (1,[a*a])
(2)  |   square(n,L) =
         let
(3)          val halfn = n div 2;
(4)          val (A,B) = break(halfn,(n,L));
(5)          val AB = mult(A,B);
(6)          val Asquared = square(A);
(7)          val Bsquared = square(B);
(8)          val middle = shift(halfn+1,AB);
(9)          val last = shift(2*halfn,Bsquared);
         in
(10)         add(Asquared,add(middle,last))
         end;
```

Fig. 1. The function `square`.

`break` and `shift` each take time $O(n)$, where n is the size of the second argument (the BI).

`mult` is the ordinary multiplication from Assignment 4 and takes $O(n^2)$ time where n is the sum of the sizes of its two arguments.

`add` takes time $O(n)$, where n is the sum of the sizes of its two arguments.

a) In the space below, draw the structure tree for `square`. Indicate for each node of the tree what type of program structure it represents (choose from if-match, match, block, and "assignment"; the latter includes both "val" statements and expressions that are evaluated). Also indicate for each node the line or lines of the `square` function that the node represents.

b) Indicate on your structure tree from (a) the running time for each node of your tree, using $T(n)$ for the running time of the `square` function itself when its argument is a BI of size n.

c) We wish to set up a recurrence for $T(n)$, the running time of `square`. What is a suitable basis value for n? ______ Write the basis for your recurrence:

---

d) What is the inductive part of your recurrence for $T(n)$?

---

e) Solve your recurrence, by lookup in a table if you wish.

Problem 4: (15 points) Below is a datatype `'elt group` that allows singles, pairs, and triples of elements of some type `'elt`.

```
datatype 'elt group =
    One of 'elt |
    Two of 'elt * 'elt |
    Three of 'elt * 'elt * 'elt;
```

a) Write a function `first(x)` that takes an argument x of type `'elt group` and returns the first element of the single, pair, or triple.

---

b) What is the type of your function `first`?

Problem 5: (15 points) Consider the probability space of 36 points representing the toss of two (ordinary) six-sided dice. Let E be the event that the first die is 1 or 2. Let F be the event that the sum of the dice is 2, 3, or 4.

a) What is PROB(E)? ______

b) What is PROB(F)? ______

c) What is PROB(E/F)? ______

Problem 6: (15 points) Suppose $T(n)$ is defined, for n a power of 2, by:

$$T(1) = 1$$
$$T(n) = 4T(n/2) + n, \text{ for } n > 1$$

We wish to prove by complete induction that $T(n) \leq 2n^2 - n$ for all n a power of 2. The approach is a simplification of "guess and check," because here we have only concrete constants, rather than symbolic constants $a, b, \dots$, etc.

a) Write the appropriate statement $S(n)$ here, beginning "if n is a power of 2 then $\cdots$. Note that $S(n)$ is trivially true when n is not a power of 2.

---

b) What is your basis value n? ______ Prove the basis here:

---

c) Prove the inductive step here:
