## CS109A Final Examination Solutions

**Problem 1:** (20 points) In this question we shall ask about whether an element $x$ is in a set $S$, by giving conditions on $x$, $S$, and other sets. For each of the parts below, answer A, B, or C, with the following meanings.

A: $x$ is surely in the set $S$.
B: $x$ cannot possibly be in the set $S$.
C: It cannot be determined from what is given whether $x$ is or is not in $S$.

*Note*: The problem is scored 4 points for a correct answer, 0 for no answer, and $-2$ for an incorrect answer.

a) $x$ is in $S \cup T$ and $x$ is NOT in $T$. *Ans.*: A.

b) $x$ is NOT in $S \cap T$ and $x$ is NOT in $T$. *Ans.*: C.

c) $x$ is NOT in $T - S$ and $x$ is in $T$. *Ans.*: A.

d) $x$ is in $R - (R - T)$ and $S = R \cap T$. *Ans.*: A.

e) $x$ is NOT in $R - (R - T)$ and $S = R \cap T$. *Ans.*: B. Note that (d) and (e) exploit the law $\big(R - (R - T)\big) \equiv (R \cap T)$.

**Problem 2:** (20 points) The *symmetric difference* of sets $S$ and $T$ is the set of elements that are in exactly one of $S$ and $T$. Two expressions for the symmetric difference are $(S - T) \cup (T - S)$ and $(S \cup T) - (S \cap T)$. Prove these two expressions equivalent by showing that an element $x$ in one is also in the other. In writing your proof, you may use as shorthand the following reasons:

(1) Given.
(2) Definition of union.
(3) Definition of intersection.
(4) Definition of set difference.

*Suggestion*: If you need to consider two cases, divide the page left-to-right and show your reasoning for each case "in parallel."

a) Show that if $x$ is in $(S - T) \cup (T - S)$ then $x$ is in $(S \cup T) - (S \cap T)$. *Ans.*:

| | |
|---|---|
| a) $x$ is in $(S - T) \cup (T - S)$ (1) | |
| b) $x$ is in $S - T$ or $x$ is in $T - S$ (2) | |
| c) Case 1: $x$ is in $S - T$ | Case 2: $x$ is in $T - S$ |
| d) $x$ is in $S$ (4) | $x$ is in $T$ (4) |
| e) $x$ is not in $T$ (4) | $x$ is not in $S$ (4) |
| f) $x$ is in $S \cup T$ (2) with (d) | (in either case) |
| g) $x$ is not in $S \cap T$ (3) with (e) | (in either case) |
| h) $x$ is in $(S \cup T) - (S \cap T)$ (4) with (f,g) | |

b) Show that if $x$ is in $(S \cup T) - (S \cap T)$ then $x$ is in $(S - T) \cup (T - S)$. *Ans.*:

| | | |
|---|---|---|
| a) | $x$ is in $(S \cup T) - (S \cap T)$ (1) | |
| b) | $x$ is in $S \cup T$ (4) with (a) | |
| c) | $x$ is not in $S \cap T$ (4) with (a) | |
| d) | $x$ is in $S$ or $x$ is in $T$ (2) with (b) | |
| e) | Case 1: $x$ is in $S$ | Case 2: $x$ is in $T$ |
| f) | $x$ is not in $T$ (3) with (c,e) | $x$ is not in $S$ (3) with (c,e) |
| g) | $x$ is in $S - T$ (4) with (e,f) | $x$ is in $T - S$ (4) with (e,f) |
| h) | $x$ is in $(S - T) \cup (T - S)$ (2) with (g) | (in either case) |

**Problem 3:** (15 points) Suppose we use the hash function on integers: $h(i) = i^2$ `mod` 10 (i.e., the remainder when $i$ is squared and then divided by 10) and use a hash table with 10 buckets, numbered 0 to 9.

a) If we hash the twenty integers $1, 2, \ldots, 20$ into the table, what is the distribution of integers in the buckets? For each of the buckets, give the number of elements in that bucket. *Ans.*: $(2, 4, 0, 0, 4, 2, 4, 0, 0, 4)$.

b) If we were to hash the first thousand integers $1, 2, \ldots, 999$ (Comment: that should have been 1000; only one person noticed this!) into this hash table, would you expect a more even distribution? *Ans.*: No. Why or why not? *Ans.*: The value of $h(i)$ depends only on the last digit. Thus, the pattern of the first ten or twenty numbers will repeat, and you never get any in buckets 2, 3, 7, or 8.

**Problem 4:** (15 points) The following is an ML declaration suitable for the datatype `ttree`, a "ternary" tree in which each node has "slots" for three children: a left, middle, and right child. As with binary trees, any one or more of these slots may be "null," while the other slots indicate nodes (or in the ML style, indicate the trees rooted at those nodes).

```
datatype 'label ttree =
    null |
    node of
        ('label * ('label ttree) * ('label ttree) * ('label ttree));
```

The following describes a certain `ttree`.

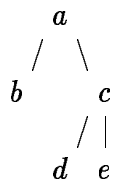
*i*) The root has label `"a"`, a left child described in (*ii*) below, no middle child, and a right child described in (*iii*) below.

*ii*) The left child of the root is a leaf with label `"b"`.

*iii*) The right child of the root has label `"c"`, a left child that is a leaf labeled `"d"`, a middle child that is a leaf labeled `"e"`, and no right child.

Answer the following about this `ttree`.

a) Draw a picture of the tree in the space below. A left child should extend to the left, a middle child be directly below, and a right child extend to the right. Represent each node by its label. *Ans.*:




b) Write an ML expression that denotes this `ttree`. *Ans.*:

```
node("a", node("b",null,null,null),
    null,
    node("c", node("d",null,null,null),
        node("e",null,null,null),
        null
    )
);
```

Note that we use a type constructor, such as `node` or `null`, by applying it as if it were a function to the value(s) of its fields. A number of people used the $*$ operator, which is needed to define types, in place of the comma that separates fields of a structure. In "ordinary" expressions, $*$ has its usual multiplicative meaning.

**Problem 5:** (20 points) Write an ML program `pre` that takes as argument a `ttree` (as defined in Problem 4) and produces a list of this tree's labels, in preorder. *Ans.*:

```
fun pre(null) = []
| pre(node(label,left,mid,right)) =
        label::pre(left)@pre(mid)@pre(right);
```

Some people used `::` in place of the `@` operator in the line above. Remember the distinction between cons and concatenation. You'd get a type error, as well as a logical error, if you tried to connect two lists of the same type by the `::` operator.

**Problem 6:** (20 points) Prove by structural induction that in a *nonempty* ternary tree the number of NIL pointers (nulls in the ML datatype) is one more than twice the number of nodes. *Warning*: You will receive no credit if your proof is not a structural induction, e.g., you will receive no credit if it is a complete induction on the height or the number of nodes. To organize your proof, answer the following questions.

a) What is the statement $S(T)$ to be proved? *Ans.*: Let $n(T)$ be the number of nodes of ternary tree $T$ and let $x(T)$ be the number of nils in $T$. Then $x(T) = 2n(T) + 1$.

b) What is a suitable basis case? *Ans.*: $T$ is a single node.

c) Prove the basis case. *Ans.*: If $T$ is a single node, then $n(T) = 1$ and $x(T) = 3$. Clearly $x(T) = 2n(T) + 1$ in this case.

d) Prove the inductive step. While there are some options regarding the level of detail appearing in the proof, a proof will not be considered correct unless it indicates clearly where and how the inductive hypothesis is used. *Ans.*: Let $T$ have root $r$. There are three cases depending on whether $r$ has 1, 2, or 3 children. (aside: note that if we

think of a missing child as a nil pointer, then the relationship nils = 2 * nodes + 1 holds, since $1 = 2 * 0 + 1$. However, the inductive hypothesis does not apply to an empty tree.)

Case 1: (1 nonnil child, $T_1$). Then $x(T) = x(T_1) + 2$ and $n(T) = n(T_1) + 1$. By the inductive hypothesis, $x(T_1) = 2n(T_1) + 1$. Thus, $x(T) = 2n(T_1) + 1 + 2 = 2n(T) + 1$.

Case 2: (2 nonnil children, $T_1$ and $T_2$). $x(T) = x(T_1) + x(T_2) + 1$ and $n(T) = n(T_1) + n(T_2) + 1$. By the inductive hypothesis, $x(T_1) = 2n(T_1) + 1$ and $x(T_2) = 2n(T_2) + 1$. Thus, $x(T) = 2n(T_1) + 1 + 2n(T_2) + 1 + 1 = 2n(T) + 1$.

Case 3: (3 nonnil children, $T_1$, $T_2$, and $T_3$). $x(T) = x(T_1) + x(T_2) + x(T_3)$ and $n(T) = n(T_1) + n(T_2) + n(T_3) + 1$. By the inductive hypothesis, $x(T_1) = 2n(T_1) + 1$, $x(T_2) = 2n(T_2) + 1$, and $x(T_3) = 2n(T_3) + 1$. Thus, $x(T) = 2n(T_1) + 1 + 2n(T_2) + 1 + 2n(T_3) + 1 = 2n(T) + 1$.

**Problem 7:** (20 points) We would like to find the longest common subsequences of the strings `babcc` and `abacbca`.

a) In the space below, create a table giving the length of the LCS between each pair of prefixes, one from each of the strings. Please use columns to correspond to prefixes of `abacbca` and rows to represent prefixes of `babcc`. *Ans.*:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| `c` | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 |
| `c` | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 |
| `b` | 0 | 1 | 2 | 2 | 2 | 3 | 3 | 3 |
| `a` | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 |
| `b` | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | `a` | `b` | `a` | `c` | `b` | `c` | `a` |

b) What is the length of the LCS of these strings? *Ans.*: 4.

c) What are all the LCS's of this length? Just give the LCS's; you need not indicate the positions of the two original strings from which they come. *Ans.*: `abcc`, `bacc`, and `babc`.

**Problem 8:** (20 points) Suppose strings of lengths 1 through 10 are equally likely, i.e., 10% probability each. Also, suppose that we wish to store strings in linked lists of cells consisting of room for $k$ bytes and a pointer to the next cell. Assume that the number of bytes taken by a cell with room for $k$ characters is $k + 2$ bytes (i.e., pointers take 2 bytes, and there is no requirement that fields begin at a position that is a multiple of 4).

a) Describe briefly how you would calculate the average number of bytes needed to store strings if $k = 5$. *Ans.*: Half the strings will take one cell and half will take two cells. Thus, the average space is the average of the space for 1 and 2 cells, that is $(7 + 14)/2 = 10.5$.

b) Give the average number of bytes required for the following values of $k$. *Ans.*: $k = 1$: 16.5. $k = 2$: 12. $k = 3$: 11. $k = 4$: 10.8.

**Problem 9:** (20 points) Let $S$ be the set of pairs $(c, j)$, where $c$ is one of the colors red, green, or blue (abbreviate $r$, $g$, and $b$, respectively), and $j$ is a nonnegative integer. We wish to show that $S$ is a countable set.

a) Define a one-to-one correspondence $f$ from the nonnegative integers to $S$. *Ans.*: Let $f(i) = (r, i \texttt{ div } 3)$ if $i \bmod 3 = 0$, let $f(i) = (g, i \texttt{ div } 3)$ if $i \bmod 3 = 1$, and let $f(i) = (b, i \texttt{ div } 3)$ if $i \bmod 3 = 2$.

b) Explain briefly why $f$ is a one-to-one correspondence. *Ans.*: $f$ is defined for all nonnegative integers $i$, and it has a unique value, so $f$ is a total function. For any $(c, j)$ in $S$, there is an $i$ such that $f(i) = (c, j)$; namely if $c = r$, then $i = 3j$, if $c = g$, then $i = 3j + 1$, and if $c = b$, then $i = 3j + 2$. Suppose there were two different $i$ and $i'$ such that $f(i) = f(i')$. Then $i - i'$ would have to be a multiple of 3, or else the "color" components of $f(i)$ and $f(i')$ would be different. However, if $i$ and $i'$ differ, they must thus differ by at least 3, and therefore, $i \texttt{ div } 3$ would not equal $i'$ div 3. Thus, $f^{-1}$ is also a total function.

**Problem 10:** (30 points) Let the domain $D$ be the integers from 1 to 10, and let $R$ be the relation on $D$ defined by $aRb$ if and only if $a$ and $b$ have a common divisor greater than 1. For example, $4R4$ and $4R6$, but the following are false: $1R5$, $2R5$, and $3R5$. Tell whether $R$ has each of the following properties and either give a brief proof if so or a counterexample if not.

a) $R$ is reflexive? *Ans.*: No; $1R1$ is false.

b) $R$ is symmetric? *Ans.*: Yes. If $a$ and $b$ have a common divisor greater than 1, then $b$ and $a$ have the same divisor.

c) $R$ is antisymmetric? *Ans.*: No; for example $2R4$ and also $4R2$.

d) $R$ is transitive? *Ans.*: No; for example $2R6$ and $6R3$, but not $2R3$.

e) $R$ is a partial order? *Ans.*: No; a partial order must be transitive.

f) $R$ is an equivalence relation? *Ans.*: No; an equivalence relation must be transitive.

**Problem 11:** (20 points) Prove by induction on $n$ that for all $n \geq 1$, the sum

$$\frac{1}{1 \times 2} + \frac{1}{2 \times 3} + \frac{1}{3 \times 4} + \cdots + \frac{1}{n \times (n+1)} \qquad \left(\text{that is, } \sum_{i=1}^{n} \frac{1}{i(i+1)}\right)$$

equals $\dfrac{n}{n+1}$. To organize your proof, answer the following questions.

a) What is the statement $S(n)$ to be proved? *Ans.*: $\sum_{i=1}^{n} \frac{1}{i(i+1)} = n/(n+1)$.

b) What is a suitable basis case? *Ans.*: $n = 1$. ($n = 0$ can be made to work also.)

c) Prove the basis case. *Ans.*: For $n = 1$ we have $\sum_{i=1}^{1} \frac{1}{i(i+1)} = 1/2$ and $n/(n+1) = 1/2$.

d) When we prove the inductive step, we assume $S(n)$. What must we prove, using this assumption? *Ans.*: $S(n+1)$, that is, $\sum_{i=1}^{n+1} \frac{1}{i(i+1)} = (n+1)/(n+2)$.

e) Prove the inductive step. While there are some options regarding the level of detail appearing in the proof, a proof will not be considered correct unless it indicates clearly where and how the inductive hypothesis is used. *Ans.*:

$$\sum_{i=1}^{n+1} \frac{1}{i(i+1)} = \sum_{i=1}^{n} \frac{1}{i(i+1)} + \frac{1}{(n+1)(n+2)}$$

By the inductive hypothesis, $\sum_{i=1}^{n} \frac{1}{i(i+1)} = n/(n+1)$. Thus,

$$\sum_{i=1}^{n+1} \frac{1}{i(i+1)} = n/(n+1) + 1/(n+1)(n+2)$$

By algebraic manipulation, $\sum_{i=1}^{n+1} \frac{1}{i(i+1)} = (n+1)/(n+2)$.

**Problem 12:** (20 points) We wish to represent sets $S$ and $T$ that consist of $n/2$ integers chosen from the integers 1 through $n$, and we shall consider three data structures to do so:

A: A linked list (unsorted, no duplicates).

B: A hash table with $n$ buckets.

C: A Boolean array indexed by integers `1..n`, used as a characteristic vector.

For each of the operations suggested below, give the big-oh running time of the operation, assuming the most efficient possible implementation. For the hash table, you should give the average running time, and in the other two cases give the worst-case running time.

a) Insertion of an element $x$ into a set $S$. *Ans.*: A: $O(n)$; B: $O(1)$; C: $O(1)$. Note that in case A, insertion cannot be accomplished in $O(1)$ time, because we need to check for duplicates; thus we must examine the whole list if necessary.

b) Computing the union of the two sets $S$ and $T$. *Ans.*: A: $O(n)$; B: $O(n)$ C: $O(n)$. A came as a surprise to many. Some people realized that you could sort the lists and use the algorithm described in the book to take the union of the lists in $O(n)$ time. But the $O(n \log n)$ time to sort dominates. A better approach is to use a characteristic vector of length $n$ for each set. It only takes $O(n)$ time to convert lists to characteristic vectors and back again *in this special case*, because the lengths of the lists and the sizes of the domains are within a constant factor of one another.

**Problem 13:** (20 points) Consider the following fragment of Pascal code. Note that `ODD` is a built-in Pascal function, not a procedure call, and can be assumed to take $O(1)$ time.

```
    begin
(1)     sum := 0;
(2)     for i := 1 to n do
(3)         if ODD(i) then
(4)             for j := 1 to i do
(5)                 sum := sum + A[j]
(6)         else
(7)             for j := i to n do
(8)                 sum := sum - A[j]
    end;
```

In the space below, draw a tree showing how the lines are grouped into progressively larger statements. For each statement, indicate the lines it covers and give a smooth and tight big-oh bound on its running time (as a function of the value of $n$). *Ans.*:

| | | | |
|---|---|---|---|
| | | Block 1–8 | $O(n^2)$ |
| $O(1)$ | Asgn 1 | For 2–8 | $O(n^2)$ |
| | | If 3–8 | $O(n)$ |
| $O(n)$ | For 3–4 | For 7–8 | $O(n)$ |
| $O(1)$ | Asgn 4 | Asgn 8 | $O(1)$ |

The most common error was thinking that 3–5 and/or 6–8 were statements. Surely an else-clause does not stand by itself, and the if-clause of 3–5, while it could stand as a statement by itself, is in this case evidently part of an if-then-else structure. Page 109 of the text covers the rules for statement construction in Pascal.

**Problem 14:** (20 points) For each of the following, tell whether the stated big-oh relationship holds. If not, you can just say "false." If the relationship does hold, then you must give witnesses $n_0$ and $c$ that are minimal in the sense that no smaller $n_0$ for the same $c$ works, and no smaller $c$ works for the same $n_0$. Note that there may be more than one minimal pair of witnesses for a given problem, so there is not always a unique correct answer.

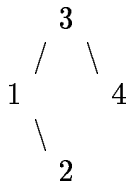
a) $7n + 50$ is $O(n^2)$. *Ans.*: $n_0 = 1$; $c = 57$. There are many other choices. However, most people who tried something else lost points because they forgot that there is no requirement that $c$ be an integer. Since we were looking for the least $c$ for a given $n_0$, if you — say — picked $n_0 = 12$, you had to say $c = 134/144$, not $c = 1$.

b) $10^n$ is $O(n^n)$. *Ans.*: $n_0 = 10$; $c = 1$.

c) Let $f(n) = n^3$ for $n = 0, 1, \ldots, 1000$, and $f(n) = n^2$ for $n > 1000$. $f(n)$ is $O(n^2)$. *Ans.*: $n_0 = 1001$; $c = 1$. Another reasonable choice is $n_0 = 0$; $c = 1000$. Note that $n_0 = 1$ with $c = 1000$ is not minimal because $n_0$ can be reduced to 0 for the same $c$.

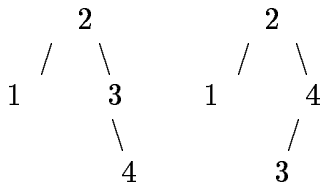
d) Let $f(n)$ be as in (c). Let $g(n) = n^3$ for even $n$ and $g(n) = 10n$ for odd $n$. $f(n)$ is $O(g(n))$. *Ans.*: False.

**Problem 15:** (20 points) The following is a binary search tree formed by inserting into an empty tree the integers 1, 2, 3, and 4, in some order.

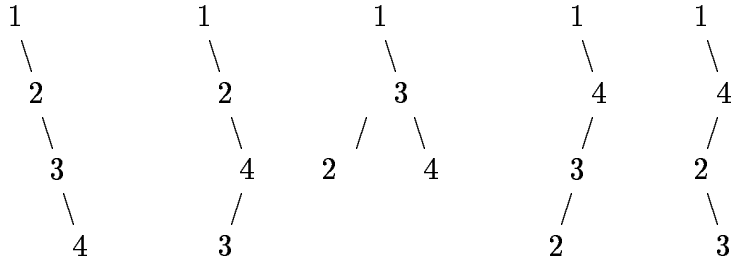



a) What are all the orders in which the four integers could have been inserted to form this particular binary search tree? *Ans.*: 3124, 3142, and 3412.

b) There are many other binary search trees that can be formed from the integers 1 through 4 by insertion in some other order. In the space on the next page, show all the possible BST's that have 2 at the root. *Ans.*:




c) In the space below, show all the BST's with integers 1 through 4 that have 1 at the root.




d) How many BST's with the integers 1 through 4 have 4 at the root? *Ans.*: 5