NAME: ______________________________

ID: ______________________________

C241 Test Three

1	/12
2	/12
3	/10
4	/10
5	/16
6	/20
7	/20
8	/12
9	/10
	/120

INSTRUCTIONS:

- *Put your name and ID number in the upper-right of this page.* Do not *detach any pages.*
- *Exam time is 120 minutes.*
- *Questions are differently weighted, see right.*
- *Blank answers get no credit, so even if you're not sure how to answer a question, show what you do know about questions of that kind.*
- *Place your final answers in the spaces provided. Scratch work is not graded, but neatness counts.*

1. (12 points) Recall:

Definition 6.1. Let $f, g\colon \mathbb{N} \to \mathbb{R}^+$. We say that f *is of order* g, written $f(n) \in O\big(g(n)\big)$, if there exist $N \in \mathbb{N}$ and $C \in \mathbb{R}$ such that for all $n \geq N$, $f(n) \leq C \cdot g(n)$.

(a) *Find witnesses* N *and* C *that show* $7n + 5 \in O\big(n\big)$.

$$\left.\begin{array}{lrcll} & 7n & \leq & 7n & \text{if } n \geq 1 \\ + & 5 & \leq & n & \text{if } n \geq 5 \\ \hline & 7n+5 & \leq & 8n & \text{if } n \geq 5 \end{array}\right\} \begin{array}{l} N = 5 \\ C = 8 \end{array}$$

(b) *Find witnesses* N *and* C *that show* $4n^2 + 3n + 1 \in O\big(n^2\big)$.

$$\left.\begin{array}{lrcll} & 4n^2 & \leq & 4n^2 & \text{if } n \geq 1 \\ + & 3n & \leq & n^2 & \text{if } n \geq 3 \\ + & 1 & \leq & n^2 & \text{if } n \geq 1 \\ \hline & 4n^2+2n+1 & \leq & 4n^2 & \text{if } n \geq 3 \end{array}\right\} \begin{array}{l} N = 3 \\ C = 4 \end{array}$$

(c) Prove: *If* $f_1(n) \in O\big(g_1(n)\big)$ *and* $f_2(n) \in O\big(g_2(n)\big)$ *then* $f_1(n) \times f_2(n) \in O\big(g_1(n) \times g_2(n)\big)$.

- Since $f_1(n) \in O\big(g_1(n)\big)$ there exist N_1 and C_1 such that for all $n \geq N_1$, $f_1(n) \leq C_1 \cdot g_1(n)$.
- Since $f_2(n) \in O\big(g_2(n)\big)$ there exist N_2 and C_2 such that for all $n \geq N_2$, $f_2(n) \leq C_2 \cdot g_2(n)$.
- All quantities are positive, so we can multiply inequalities to get $f_1(n) \times f_2(n) \leq C_1C_2 \cdot g_1(n) \times g_2(n)$ for all $n \geq C_1C_2$.
- Thus, $f_1 f_2 \in O\big(g_1 g_2\big)$ with witnesses $N = \max(N_1, N_2)$ and $C = C_1C_2$.

2. (12 points) Analysis of recursive "divide and conquer" algorithms may yield recursive performance estimates like T, to the right. To show that $T(n) \in O(n \log n)$ it is easier to restrict n to be a power of two:

Prove: *For all* $n \in \mathbb{N}$, $T(2^n) = 2^n(a + bn)$.

$$\boxed{\begin{array}{l} T\colon \mathbb{W} \to \mathbb{R} \\ T(1) \quad = \quad a \\ \text{and for } m > 1, \\ T(m) \quad = \quad 2 \cdot \left[T\left(\frac{m}{2}\right)\right] + mb \end{array}}$$

(BASE CASE)

$$T(2^0) = T(1) = a = 2^0 \cdot (a + 0 \cdot b)$$

(INDUCTION STEP) Assume $T(2^k) = 2^k(a + bk)$. Then

$$\begin{aligned} T(2^{k+1}) = \quad & 2\left[T\left(\frac{2^{k+1}}{2}\right)\right] + 2^{k+1}b \\ = \quad & 2\left[T\left(2^k\right)\right] + 2^{k+1}b \\ \overset{IH}{=} \quad & 2\left[2^k\left(a + bk\right)\right] + 2^{k+1}b \\ = \quad & 2\left[2^k a + 2^k bk\right] + 2 \cdot 2^k b \\ = \quad & 2\left[2^k a + 2^k bk + 2^k b\right] \\ = \quad & 2^{k+1}\left(a + b(k+1)\right) \end{aligned}$$

as needed. This completes the induction. $\square$

3. (10 points) Recall that a *substitution*, $F\begin{bmatrix}U_1,\dots,U_n\\ v_1,\dots,v_n\end{bmatrix}$ denotes the sentence obtained by simultaneously replacing all occurrences of variables v_i by the corresponding words U_i in formula F.

For $F \equiv x + (y - z) + wx$, write the results of the following substitutions:

(a) $F\begin{bmatrix}a,b\\ x,y\end{bmatrix} \equiv (a + (b - z) + wa$

(b) $F\begin{bmatrix}b\\ q\end{bmatrix} \equiv (x + (y - z) + wx$

(c) $F\begin{bmatrix}y,b\\ x,y\end{bmatrix} \equiv (y + (b - z) + wy$

(d) $F\begin{bmatrix}y\\ x\end{bmatrix}\begin{bmatrix}b\\ y\end{bmatrix} \equiv (b + (b - z) + wb$

(e) $F\begin{bmatrix}(y-w)\\ x\end{bmatrix}\begin{bmatrix}(y-z)\\ w\end{bmatrix} \equiv (y - (y - z)) + (y - z) + (y - z)(y - w)$

4. (10 points) Let A be the set of alphabetic characters, $A = \{\texttt{a}, \texttt{b}, \dots, \texttt{z}\}$ and let `#` and `$` be two operation symbols. Define a infix language in which

(a) `$` takes precedence over `#`,
(b) `$` associates to the right, and
(c) `#` associates to the left.

For instance, `a$b$c#d#e#f$g` would be parsed as $\boxed{\boxed{\boxed{\boxed{\texttt{a \$ }\boxed{\texttt{b \$ c}}}\ \texttt{\# d}}\ \texttt{\# e}}\ \texttt{\#}\ \boxed{\texttt{f \$ g}}}$

You may—but are not required to—use Backus-Naur notation (BNF).

$F \in (A \cup \{\texttt{\$}\})^+$
1. $A \subseteq F$
2. $x \in A \ \wedge\ v \in F \ \Rightarrow\ x\,\texttt{\$}\,v \in F$
3. nothing else

$E \in (F \cup \{\texttt{\#}\})^+$
1. $F \subseteq E$
2. $u \in E \ \wedge\ v \in F \ \Rightarrow\ u\,\texttt{\#}\,v \in F$
3. nothing else

or

$$
\begin{array}{ll}
\langle A\rangle & ::= \{\texttt{a, b \dots, z}\} \\
\langle F\rangle & ::= \langle A\rangle \\
 & ::= \langle A\rangle\,\texttt{\#}\,\langle F\rangle \\
\langle E\rangle & ::= \langle F\rangle \\
 & ::= \langle E\rangle\,\texttt{\$}\,\langle F\rangle
\end{array}
$$

5. (16 points) The language L and functions I, A, R and T, defined below, are the same as in Section 3.6.

	$L \subseteq \{\mathtt{a},\ \mathtt{b},\ \bullet\}^+$
1.	$\bullet \in L$
2a.	$u \in L \Rightarrow \mathtt{a}u \in L$
2b.	$u \in L \Rightarrow \mathtt{b}u \in L$
3.	nothing else

	$I\colon L \to L$
1.	$I(\bullet) = \bullet$
2a.	$I(\mathtt{a}u) = \mathtt{b}I(u)$
2b.	$I(\mathtt{b}u) = \mathtt{a}I(u)$

	$R\colon L \to L$
1.	$R(\bullet) = \bullet$
2a.	$R(\mathtt{a}u) = A(R(u), \mathtt{a}\bullet)$
2b.	$R(\mathtt{b}u) = A(R(u), \mathtt{b}\bullet)$

	$A\colon L^2 \to L$
1.	$A(\bullet, v) = v$
2a.	$A(\mathtt{b}u, v) = \mathtt{b}A(u, v)$
2b.	$A(\mathtt{a}u, v) = \mathtt{a}A(u, v)$

	$T\colon L^2 \to L$
1.	$T(\bullet, v) = v$
2a.	$T(\mathtt{a}u, v) = T(u, \mathtt{a}v)$
2b.	$T(\mathtt{b}u, v) = T(u, \mathtt{b}v)$

(a) Indicate which of the following are true:

i) $\boxed{\text{T}}$ $\bullet \in L$

ii) $\boxed{\text{F}}$ $\mathtt{a} \in L$

iii) $\boxed{\text{T}}$ $A(R(\mathtt{ba}\bullet), \mathtt{a}\bullet) = A(A(R(\mathtt{a}\bullet), \mathtt{b}\bullet), \mathtt{a}\bullet)$

iv) $\boxed{\text{T}}$ $R(I(\bullet)) = \bullet$

v) $\boxed{\text{F}}$ $A(I(u), R(v)) = A(I(u), v)$

vi) $\boxed{\text{F}}$ $T(u, \bullet) = A(\bullet, u)$

vii) $\boxed{\text{F}}$ $I(u) = u \begin{bmatrix}\mathtt{a}\\ \mathtt{b}\end{bmatrix} \begin{bmatrix}\mathtt{b}\\ \mathtt{a}\end{bmatrix}$

viii) $\boxed{\text{T}}$ $I(u) = u \begin{bmatrix}\mathtt{a},\mathtt{b}\\ \mathtt{b},\mathtt{a}\end{bmatrix}$

(b) Use structural induction to prove: *For all* $u \in L$, $A(u, \bullet) = u$.

Base Case. $A(\bullet, \bullet) \overset{A.1}{=} \bullet$.

Induction: Assume $IH \equiv A(u, \bullet) = u$.

$$A(\mathtt{a}u, \bullet) \overset{A.2a}{=} \mathtt{a}\ A(u, \bullet) \overset{IH}{=} \mathtt{a}u$$

Similarly, for $\mathtt{b}u$,

$$A(\mathtt{b}u, \bullet) \overset{A.2b}{=} \mathtt{b}\ A(u, \bullet) \overset{IH}{=} \mathtt{b}u$$

6. (20 points) This question has four parts on this and the next page.

The *language* TREE, defined below, is a *symbolic representation* of binary tree structures.

$\text{TREE} \subseteq \{\texttt{N}, \texttt{L}, \texttt{(}, \texttt{)}\}^+$
1. $\texttt{L} \in \mathit{Tree}$
2. if $t_1, t_2 \in \mathit{Tree}$ then $\texttt{(N}\hat{\ }t_1\hat{\ }t_2\texttt{)} \in \text{TREE}$
3. nothing else

(a) Indicate which of these sentences are words in the language TREE?

(i)	Y	`L`	(iv)	N	`(L)`
(ii)	Y	`(N L L)`	(v)	N	`(N N L)`
(iii)	Y	`(N (N L L) L)`	(vi)	Y	`(N (N L L) (N L (N L L)))`
(iii)	N	`(N ((N L L)) L)`	(vii)	N	`(N (N L L (N L)))`

(b) The *size* of any word is be number of characters it contains. Fill in the missing information to define a recursive function size: TREE $\to \mathbb{N}$ that gives the size of any word $t \in \text{TREE}$.

1. $\mathsf{size}(\texttt{L}) = \boxed{1}$
2. $\mathsf{size}(\texttt{(N}\ t_1\ t_2\texttt{)}) = \boxed{3} + \mathsf{size}(t_1) + \mathsf{size}\left(\boxed{t_2}\right)$

Full credit if you entered 1 instead of 3 above.

(c) The *depth* of a tree is defined to be *one plus* the length of the longest path from its root to a leaf. Define a recursive function depth: TREE $\to \mathbb{N}$ that gives the depth of the tree it represents.

$$\begin{aligned} \mathit{depth}(\texttt{L}) &= 1 \\ \mathit{depth}(\texttt{(N}\hat{\ }t_1\hat{\ }t_2\texttt{)}) &= 1 + \max(\mathit{depth}(t_1), \mathit{depth}(t_2)) \end{aligned}$$

(d) Prove by induction that the size of a TREE is exponential in its depth. That is, *for some constant C, and for any $t \in$ TREE, $\mathsf{size}(t) < C \cdot 2^d$, where $d = \mathsf{depth}(t)$.*

COMMENT. *This is way too tricky to be a test question. Ample credit was given for any reasonable attempt. You had to answer parts (b) and (c) correctly to have much chance at a proof. If $\mathsf{size}(t)$ were defined to be the number of `N` s and `L` s, it is pretty easy to prove that $\mathsf{size}(t) \leq 2^{\mathsf{depth(t)}} - 1$. The purpose of C is to account for the parentheses. But the theorem as stated does not support an induction. Instead, one has to prove something stronger:*

Lemma. *For all $t \in$ TREE, $\mathsf{size}(t) \leq 3(2^d - 1)$, where $d = \mathsf{depth}(r)$*

PROOF.

(BASE CASE) $\mathsf{size}(\texttt{L}) = 1 \leq 3 \cdot 1 = 3(2^1 - 1) = 3(2^{\mathsf{depth}(\texttt{L})} - 1)$

(INDUCTION) Let $t =$ `(N` t_1 t_2`)` and

$$\begin{array}{rclrcl} s & = & \mathsf{size}(t) & d & = & \mathsf{depth}(t) = 1 + \max(d_1, d_2) \\ s_1 & = & \mathsf{size}(t_1) & d_1 & = & \mathsf{depth}(t_1) \\ s_2 & = & \mathsf{size}(t_2) & d_2 & = & \mathsf{depth}(t_2) \end{array}$$

Assume by induction that

$$\begin{array}{rcl} s_1 & \leq & 3(2^{d_1} - 1) \\ s_2 & \leq & 3(2^{d_2} - 1) \end{array}$$

Let $\bar{\bar{d}} = \max(d_1, d_2)$. Adding the inequalities above, we get

$$s_1 + s_2 \leq 3(2^{\bar{\bar{d}}} - 1) + 3(2^{\bar{\bar{d}}} - 1) = 3(2^{\bar{\bar{d}}+1} - 2) = 3(2^d - 2)$$

where $d = \mathsf{depth}(t)$, as defined above. Hence,

$$\begin{aligned} \mathsf{size}(t) &= 3 + s_1 + s_2 \\ &\leq 3 + 3(2^d - 2) \quad \textit{(It is at this point that we learn } C \textit{ must equal 3)} \\ &= 3 + 3 \cdot 2^d - 6 \\ &= 3 \cdot 2^d - 3 \\ &= 3(2^d - 1) \end{aligned}$$

This completes the induction step. □

Corollary. *For some constant C and for all $t \in$ TREE, $\mathsf{size}(t) < C \cdot 2^d$, where $d = \mathsf{depth}(t)$.*

PROOF. With $C = 3$ and by the Lemma, $\mathsf{size}(t) \leq 3(2^d - 1) < C \cdot 2^d$. □

7. (20 points) The *Theorem on Loop Invariants* from Chapter 5 says that to prove an assertion POST holds after the while-loop executes,

{PRE} **while** TEST **do** {INV} BODY {POST}	it suffices to prove:	INITIALIZATION: PRE $\Rightarrow$ INV. INVARIANCE: {INV $\wedge$ TEST} BODY {INV} TERMINATION: INV $\wedge$ $\neg$TEST $\Rightarrow$ POST.

Use the Theorem on Loop Invariants to prove the program below computes A^B.

```
{x = A ∧ y = B}
begin
z := 1;
while y ≠ 0 do {z · x^y = A^B}
   begin
   while even?(y) do {z · x^y = A^B ∧ y ≠ 0}
      begin y := ½y; x := x × x end;
   z := z * x;
   y := y − 1
   end
end
{z = A^B}
```

There are two loops in the program, so there are two initialization, invariance and termination arguments.

(INNER LOOP)

INITIALIZATION. The each time the program reaches the inner loop, the outer loop's invariant is true, and this, together with the loop test *even*?(y) is just the inner loop's invariant.

INVARIANCE. If y is even, the inner loop body computes new values, $x' = x^2$, $y' = \frac{1}{2}y \neq 0$ and $z' = z$.. So $z' \cdot x'^{y'} = z \cdot (x^2)^{\frac{1}{2}y} = z \cdot x^{2 \cdot \frac{1}{2}y} = z \cdot x^y = A^B$. So the inner invariant is preserved.

TERMINATION. When the inner loop terminates, its invariant is true and y is no longer an even number. Hence it is also the case that the outer invariant remains true.

(OUTER LOOP)

INITIALIZATION. When the program first reaches the outer loop, we have $x = A$, $y = B$ and $z = 1$, so $z \cdot x^y = 1 \cdot A^B = A^B$.

INVARIANCE. The outer invariant still holds after the inner loop. The body of the outer loop assigns new values $x' = x$, $y' = y - 1$ and $z' = zx$. Thus, $z' \cdot x'^{y'} = zx \cdot x^{(y-1)} = z \cdot xx^{(y-1)} = z \cdot x^y = A^B$ and the invariant is preserved.

TERMINATION. On termination we still have $z \cdot x^y = A^B$, but $y = 0$, so $x^y = 1$. Therefore the condition $z = A^B$ holds as desired.

8. (12 points) Let A be a set and $R \subseteq A \times A$ a relation. On the left below is an inductive definition of a relation $R^\star \subseteq A \times A$. The statements on the right explain what this inductive definition means. They define a sequence of relations, $R_i \subseteq A \times A$, whose union (or limit) is $R^\star$.

1. $R \subseteq R^\star$	1. $R_0 = R$
2. $(a, b),\ (b, c) \in R^\star \ \Rightarrow (a, c) \in R^\star$	2. $R_{k+1} = R_k \cup \{(a, c) \mid \exists b \in A\colon (a, b),\ (b, c) \in R_k\}$
3. nothing else	3. $R^\star = \bigcup_{i=0}^{\infty} R_i$

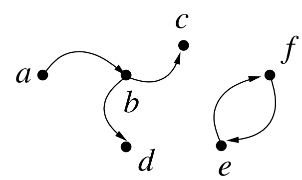
(a) For the relation depicted below-right, what are the sets R_0, R_1 R_2 R_3?

$R_0 = \{(a, b), (b, d), (b, c), (e, f), (f, e)\}$

$R_1 = \{(a, b), (b, d), (b, c), (e, f), (f, e), (a, d), (a, c), (f, f), (e, e)\}$

$R_2 =$ *(same as R_1)*

$R_3 =$ *(same as R_2)*



(b) The "nothing else" clause means that there are no unnecessary ordered pairs in $R^\star$. **Prove:** *If S is any transitive relation that contains R, then $R^\star \subseteq S$.* HINT: use the more primitive definition of $R^\star = \bigcup R_i$.

Proposition. For all $n \in \mathbb{N}$, $R_n \subseteq S$.

PROOF. The proof is by induction on n. In the base case, it is given that $R_0 \subseteq S$.

(INDUCTION) suppose $R_k \subseteq S$ ... and let (x, y) be any element of R_{k+1} ... Therefore $R_{k+1} \subseteq A$.

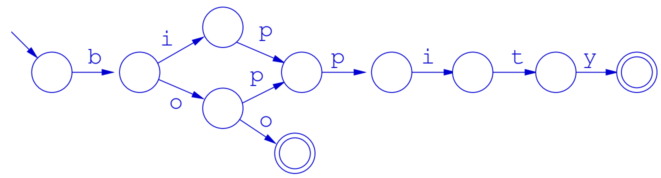
□

Now let (x, y) be any element of $R^\star$. Then because $R^\star = \bigcup R_k$, there must be some k for which $(x, y) \in R_k$. By the proposition, $R_k \subseteq S$, so $(x, y) \in S$. Since (x, y) was arbitrary, we have shown that $R^\star \subseteq S$.

9. (10 points)

(a) Draw an automaton over $A = \{$ `b` , `i` , `p` , `t` , `y` , `o` $\}$ that accepts the language $\{$ `bippity` , `boppity` , `boo` $\}$. You may use empty and nondeterminstic transitions if you wish.

One that works is



(b) Draw an automaton over $\{$ `0` , `1` $\}$ that accepts a language whose words all start and end with the same two letters, such as `101110` and `000`, but not `1001` . You may use empty and nondeterminstic transitions if you wish.

One that works is