TEST #2 - Spring 2008
Answer all 6 questions. Provide all reasoning and show all working. An unjustified answer will receive little or no credit. Begin each question on a separate page.

(15) 1. Let L be the language accepted by the NFA shown on the right. Find NFAs which accept
(a) L^c    (b) (L^c)^R.

(15) 2. (a) Find an NFA which is equivalent to the RLG given below.
   G: S->A, S->01B, A->01A, A->0C, B->A,
   B->1D, C->01, C->D, D->01S, D->11.
   (b) Convert the NFA shown below on the right in Qu.#3 into an equivalent RLG.

(18) 3(a) Find a regular expression for
   the language accepted by the NFA shown on the right.
   (b) Write down what the Halting Problem says and define what is
   the Busy-beaver function.

(18) 4(a) Define what are the initial functions and what is the
   operation known as primitive recursion.
   (b) Show that f(x,y) = 3x+4y+1 is a primitive recursive function
   by finding primitive recursive functions g and h such that
   f = prim.rec.(g,h).
   [You must show that your g and h are primitive recursive.]

(16) 5(a) Define what is a Turing computable function with domain D.
   (b) Show what happens at each step if
   01010 is the input for the TM, M shown on the right.
   (c) Find the language accepted by M.

(18) 6. Determine which of the following languages are regular and
   which are not.
   (a) L_1=\{a^k.b^k: k=n^2+1 (mod 3)\}    (b) L_2=\{b^k.c^n: k<n^2+1\}
   [If you say that it is regular, you must find a regular
   expression for it; if you say it is non-regular, you must give a
   complete proof. You may use the Pumping Lemma, if you so desire.]
Solutions to Test #2

1. \( \{A\} \xrightarrow{o} \{B\} \xrightarrow{o} \emptyset \xrightarrow{o} \emptyset \) 
\( \{A\} \xrightarrow{1} \{A, B\} \xrightarrow{0} \{B\} \xrightarrow{1} \{A, B\} \)

Note: DFAs are special NFAs

DFA for \( L \)

OAS - NFA for \( L^c \)

2(a) \( S \xrightarrow{0} B \xrightarrow{0} C \xrightarrow{0} Z \)

\( S \xrightarrow{1} A \xrightarrow{0} C \xrightarrow{1} D \xrightarrow{0} E \xrightarrow{1} \)

\( B \xrightarrow{1} D \xrightarrow{0} E \)

[\( \rightarrow C \) means \( C \) is the starting variable]

(b) \( \rightarrow C \), \( C \rightarrow 1A \), \( A \rightarrow oA \), \( A \rightarrow 1B \), \( B \rightarrow oC \)
\( B \rightarrow 0D \), \( D \rightarrow 0E \), \( E \rightarrow 1B \), \( E \rightarrow 0C \), \( E \rightarrow \lambda \)

3(a) Eliminate A to get —

Eliminate D to get —

Eliminate B to get —

\( L(M) = r_1^* r_2 (r_4 + r_3 r_1^* r_2)^* \)
\( = (10^*10)^* 10^*100 \cdot (100 + (0+10)(10^*10)^* 10^*100)^* \)
3(b) The Halting Problem asks if there is a TM such that for an arb. TM $M$ and an arb. input $w$ for $M$,
$H$ halts on $c(M)\#q(w)$ in an acc. state if $M$ halts on $w$ &
$H$ halts on $c(M)\#q(w)$ in a non-acc. state if $M$ does not halt on $w$.
Let $\mathcal{H}_n$ = set of all TMs with $n$ states & tape
alphabet $\{0,1\}$ which halts when started on the blank tape.
$L(n)$ = maximum number of 1's that a TM in $\mathcal{H}_n$ can produce.

4(a) The initial functions are: the constant 0, the zero function $z(x) = 0$,
the successor function $s(x) = x + 1$, and the projective functions
$I^{(n)}_k$ which are defined by $I^{(n)}_k(x_1, \ldots, x_n) = x_k$; $1 \leq k \leq n$.

Primitive recursion is the operation which produces a
function $f: \mathbb{N}^n \to \mathbb{N}$ from the functions $g: \mathbb{N} \to \mathbb{N}$ & $h: \mathbb{N}^n \to \mathbb{N}$
by putting $f(x, 0) = g(x)$ & $f(x, y+1) = h(x, y, f(x, y))$.

(b) $f(x, 0) = 3x + 1 \iff g(x)$
$g(0) = 3(0) + 1 = 1$
$f(x, y+1) = 3x + 4(y+1) + 1$
$\quad = (3x + 4y + 1) + 4$
$\quad = f(x, y) + 4 \iff h(x, y, f(x, y))$
$\quad = (3y + 1) + 3$
$\quad = g(y) + 3$

$g$ = prim. rec. $(s_0, s_0, s_0, I^{(2)}_2)$ & $h = s_0, s_0, s_0, I^{(3)}_2$
$f$ = prim. rec. (prim. rec. $(s_0, s_0, s_0, I^{(2)}_2)$, $s_0, s_0, s_0, I^{(3)}_2$)

5(a) A function with domain $\mathcal{D}$ is said to be Turing-computable
if we can find a TM $M$ such that for each $x \in \mathcal{D}$,
$(g_x, w) \vdash^{*} (g_y, f(w))$ is a halted computation in $M$ with $g_y \in \mathcal{A}$.

(b) $\langle A, 0101 \rangle \vdash^{*} \langle D, 1101 \rangle \vdash^{*} \langle B, 1110 \rangle \vdash^{*} \langle C, 1100 \rangle$
$\quad \vdash^{*} \langle B, 1110 \rangle \vdash^{*} \langle C, 1101 \rangle \vdash^{*} \langle F, 1110 \rangle$

(c) $L(M) = 0^* + 0^*10(10)^* + 10(10)^* = 0^* + (00 + \lambda)10(10)^*$
6(a) If \( N = 0 \pmod{3} \), then \( k = 0^2 + 1 = 1 \pmod{3} \); if \( N = 1 \pmod{3} \), then \( k = 1^2 + 1 = 2 \pmod{3} \); & if \( N = 2 \pmod{3} \), \( k = 2^2 + 1 = 2 \pmod{3} \).
So \( L_1 = a(aaa)^*bbb^* + a(aaa)^*bbb^* + a(aaa)^*bbbbb^*\)
and is therefore a regular language.

(b) Suppose \( L_2 \) was a regular language. Then we can find
a DFA \( M \) such that \( L(M) = L_2 \). Let \( N \) be the number
of states in \( M \) and consider the string \( b^{N^2}c^N \). Since
\( N^2 < (N+1)^2 \), \( b^{N^2}c^N \in L_2 \) & will be accepted by \( M \).
Since it takes \( N+1 \) states to process the \( c^N \), the acceptance
track of \( b^{N^2}c^N \) must have a loop as shown below with
\( j \geq 1 \).

Now if we skip this loop, we will see that \( M \) accepts
\( b^{N^2}c^i c^{N-i-j} = b^{N^2}c^{N-j} \).
But \( N^2 \neq (N-j)^2 + 1 \), so this contradicts the fact
that \( L(M) = L_2 \). Hence \( L_2 \) cannot be a regular lang.