COSC170 - Compiler Construction
Homework 1
Due: 2009 Feb 06, beginning of lecture

1. Draw the finite state machines, and write regular expressions that denote the strings recognized by the following finite automata:

   (a) s_1 is the start state; s_2 is the final (accepting) state

	a	b
s_1		s_2
s_2	s_3	
s_3		s_2

   (b) s_1 is the start state; s_4 is the final state

	a	b
s_1	s_2	s_3
s_2		s_4
s_3	s_4	
s_4		s_4

   (c) s_1 is both the start state and the final state

	a	b	c
s_1	s_2	s_3	s_4
s_2	s_1		
s_3		s_1	
s_4			s_1

   (d) s_1 is the start state; s_1 and s_3 are final states

	a	b	c
s_1	s_2		
s_2		s_2	s_3
s_3			

2. Write DFA's that recognize the following languages. Be careful to note starting and accepting states in your automata.

   (a) $\{w \in \{a, b\}^* :$ each a in w is followed by at least one b $\}$

   (b) $\{w \in \{a, b\}^* : w$ contains precisely three a's $\}$

   (c) $\{w \in \{a, b\}^* : w$ has an odd number of a's and an even number of b's $\}$

   (d) $\{w \in \{a, b\}^* : w$ has bab as a substring $\}$

3. Consider the following regular expression:

$$(abc^* \mid a^*bc)$$

(a) As described in class, construct an NFA for this expression.

(b) Using subset construction, convert the NFA into a DFA.

(c) Optimize the DFA to minimize the number of states.

4. Consider the grammar below:

S	::=	S `;` S
	|	`<ID> =` E
	|	`print (` L `)`
E	::=	E `+` E
	|	`(` S `,` E `)`
	|	`<ID>`
	|	`<NUM>`
L	::=	E
	|	L `,` E

(a) Write a grammar that accepts the same language as the grammar above, but that is suitable for LL(1) parsing.

(b) Find $FIRST$ and $FOLLOW$ sets for your grammar.

(c) Show the LL(1) parsing table for your grammar.

(d) Given the input string, "`x = 5; print(x, 6)`", use your parse table to expand the goal into a parse for the expression. Note which grammar production is applied at each step.