NOTE: \( n_a(w) \) means the number of a’s in the string \( w \).

1. (14 pts) Complete or answer the following.

\[
L_1 = \{abb, ac, a, b\}, \Sigma = \{a, b, c\} \\
L_2 = \{ac, aa\}, \Sigma = \{a, c\} \\
L_3 = (ab)^*c(c)^*, \Sigma = \{a, b, c\} \\
L_4 = \{w \in \Sigma^* \mid n_a(w) + n_b(w) = n_c(w)\}, \Sigma = \{a, b, c\}
\]

(a) \( L_1 \cap L_2 = \)  
(b) \( L_3 \cap L_4 = \)  
(c) \( L_2 \circ L_2 = \)  
(d) \( L_2 \times L_2 = \)  
(e) \( \overline{L_4} = \)  
(f) \( |2^{L_1}| = \)  
(g) \( L_2 - L_1 = \)

2. (22 pts) Answer TRUE or FALSE to each of the statements below.

(a) \( \{\} \in \{a, \{b\}\} \) (TRUE or FALSE?)

(b) Given a DFA \( M \), if every state except the start state is a final state, then \( L(M) = \Sigma^* - \{\lambda\} \). (TRUE or FALSE?)

(c) If \( M \) is a DFA, then there exists an NPDA \( M' \) such that \( L(M) = L(M') \). (TRUE or FALSE?)

(d) If \( G \) is a CFG with only 1 variable and two productions, then there exists a regular expression \( R \) such that \( L(G) = L(R) \). (TRUE or FALSE?)

(e) If \( G \) is a DCFG, then there exists an NPDA \( M \) such that \( L(G) = L(M) \). (TRUE or FALSE?)

(f) If \( G \) is a CFG that includes rules of the form \( A \to \lambda \), then using the brute-force parser with a string that is not in \( L(G) \) may never halt. (TRUE or FALSE?)

(g) The following grammar \( G \) is a regular grammar. (TRUE or FALSE?)

\[
S \rightarrow aS \mid B \\
B \rightarrow bB \mid \lambda
\]

(h) \( L = \{w \in \Sigma^* \mid |w| \text{ is an even number}\} \Sigma = \{a, b\} \). \( L \) is regular. (TRUE or FALSE?)
(i) \( L = \{ a^n b^m | n \text{ mod } 3 = 0, m > 10 \}, \Sigma = \{a, b\}. \) L is regular. (TRUE or FALSE?)

(j) \( L = \{ a^n b^m c^p | m > n + p, n > 0, p > 0 \}, \Sigma = \{a, b, c\}. \) L is regular. (TRUE or FALSE?)

(k) \( L = \{ w \in \Sigma^* | n_a(w) < n_b(w) \text{ and } n_b(w) \text{ is an odd number } \} \Sigma = \{a, b\}. \) L is regular. (TRUE or FALSE?)

3. (4 pts) Consider the following grammar with start variable S.

\[
S \rightarrow aSBc | aB \\
B \rightarrow bB | \lambda
\]

A) Give a right-most derivation for the string \( aaabcc \).

B) Give a parse tree for the string \( aaabcc \).

4. (5 pts) Write a CFG G for the following language:

\( L = \{ a^{2n} c b^m | n > 0 \}, \Sigma = \{a, b, c\}. \)

For example, \( aacbbbc \in L \).

5. (8 pts) Draw a DFA for the following language. You do not have to show trap states (show the transition diagram, indicate the start state by a short arrow, and final states by double circles.)

\( L = \{ w \in \Sigma^* | n_b(w) \text{ mod } 3 = 2 \text{ and } w \text{ has the substring } aab \}, \Sigma = \{a, b\}. \)

For example, \( bbab, baaba, aabaabbb, \) and \( baabbbbaa \) are in \( L \).

6. (6 pts) Consider the following DFA.

a) Show states \( q1 \) and \( q6 \) are distinguishable with an appropriate string. Explain.

b) Give the states in the minimal state DFA (you do not need to show the arcs). Each state should indicate which states it represents from the original DFA. For example you could list one state as 0,1,2 if states 0, 1 and 2 in the original DFA can be combined to form a state in the minimal state DFA.

7. (10 pts) Consider \( L = \{ a^{2n} c b^m | n > 0 \}, \Sigma = \{a, b, c\}. \) Draw the transition diagram for a nondeterministic pushdown automaton M that accepts \( L \) by final state. (Remember to identify the start state by an arrow and final states by double circles. Format of labels are \( a, b; cd \) where \( a \) is the symbol on the tape, \( b \) is the symbol on top of the stack that is popped, and \( cd \) are pushed onto the stack (with \( c \) on top of \( d \)). \( Z \) is on top of the stack when M starts.)

(a) First list 3 strings in \( L \).
8. (6 pts) **Pumping Lemma:** Let \( L \) be an infinite regular language. \( \exists \) a constant \( m > 0 \) such that any \( w \in L \) with \( |w| \geq m \) can be decomposed into three parts as \( w = xyz \) with

\[
\begin{align*}
|xy| &\leq m \\
|y| &\geq 1 \\
xy^iz \in L &\text{ for all } i \geq 0
\end{align*}
\]

Use the Pumping Lemma to prove the language \( L \) below is not regular.

\( L = \{ w \in \Sigma^* \mid n_a(w) > 2 \cdot n_b(w) \} \) \( \Sigma = \{a, b, c\} \).

**Proof:** (SHOW ALL STEPS! Some have been started for you.)

Assume \( \) \( \) \( \)

Choose \( w = \) \( \) \( \)

Show there is no way to partition this string \( w = xyz \) such that the properties of the pumping lemma hold.

9. (8 pts) Consider the following property, RemoveSomeAA. If \( L \) is a regular language, then \( \text{RemoveSomeAA}(L) = \) strings from \( L \) that have at least two adjacent \( a \)'s, and one such occurrence of \( aa \) is removed. If there is a string \( w \) in \( L \) that does not have two adjacent \( a \)'s in the string, then that does not put a string in \( \text{RemoveSomeAA}(L) \).
For example, if $aabaaba$ is in $L$, then $baabaa$ is in $\text{RemoveSomeAA}(L)$. (If a different $aa$ is replaced you could have $aabbba$ and $aabaab$ also in $\text{RemoveSomeAA}(L)$).

If $abab$ is in $L$, then $abab$ does not generate a string in $\text{RemoveSomeAA}(L)$.

Start the proof to show that $\text{RemoveSomeAA}(L)$ is a regular language. You need to explain how to convert a DFA $M$ for $L$ into a DFA or NFA that represents the language $\text{RemoveSomeAA}(L)$.