1. (10 pts) Consider the following languages. Write “REG” if it is regular, “CFL” if it is a CFL and not regular, and write “NOT” if it is not a CFL.

(a) \( L = \{ a^m b^{2n} c^n d^{3m} \mid n > 0, m > 0 \} \)  
(b) \( L = \{ a^n \mid n > 0, n \text{ is a multiple of 10} \} \)  
(c) \( L = \{ a^n b^n c^p \mid 0 < n < 500, p > 2n \} \)  
(d) \( L = \{ w \in \Sigma^* \mid n_a(w) > 2 \ast n_b(w) \text{ and } n_b(w) \text{ is even} \}, \Sigma = \{ a, b, c \}. \)  
(e) \( L = \{ w \in \Sigma^* \mid \text{for every } a \text{ there is a } b \text{ to the right of it that can be matched only with this } a \}, \Sigma = \{ a, b, c \}. \) For example \( ababba \) and \( abbaab \) are NOT in \( L, aababb \) is in \( L. \)

2. (12 pts) Answer TRUE or FALSE to each of the statements below.

(a) The class of languages that are SLR(1) is a smaller class than the class of languages that are CFL. (TRUE or FALSE?)
(b) In the SLR parsing stack, both states and symbols are put on the stack. The parsing could easily be done without the symbols on the stack. Only the state numbers are needed on the stack. (TRUE or FALSE?)
(c) Given a language \( L_1 \) and \( L_2 \) such that \( L_1 \) is a regular language and \( L_2 \) is a regular language, then \( L_1 \cap L_2 \) is a CFL. (TRUE or FALSE?)
(d) A Turing machine defined over an alphabet that has only one symbol will always halt. (TRUE or FALSE?)
(e) A brute force parser applied to a CFG can stop and reject a string \( w \) if it has already tried all derivations of length \( <= 2 \ast | w |. \) (TRUE or FALSE?)
(f) The FOLLOW set of a variable in a grammar can contain \( \lambda. \) (TRUE or FALSE?)

3. (3 pts) Explain the difference between \( \Sigma \) and \( \Gamma \) in the formal definition of a Turing machine.

4. (2 pts) Which grammars work better with the LL(1) parsing method, left-recursive or right-recursive?
5. (3 pts) Explain the difference between Turing machines and pushdown automata in the acceptance (or not acceptance) of strings.

6. (3 pts) Can an SLR parser have a shift/shift conflict? Give an example or explain why it cannot.

7. (3 pts) The following grammar is LL(k) for what value of k?

\[
\begin{align*}
S & \rightarrow aABBc \\
A & \rightarrow aBa \mid abac \\
B & \rightarrow bb \mid b
\end{align*}
\]

8. (11 pts) Consider the following grammar (DO NOT change the grammar):

\[
\begin{align*}
S & \rightarrow BSaB \mid a \mid \lambda \\
A & \rightarrow BSc \\
B & \rightarrow bBA \mid \lambda
\end{align*}
\]

For this problem you will construct the LL(1) parse table.

(a) Calculate FIRST and FOLLOW for the variables in the grammar.

<table>
<thead>
<tr>
<th></th>
<th>FIRST</th>
<th>FOLLOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Calculate all entries in the LL(1) Parse Table. If there are multiple rules for an entry, write in all the rules.
9. (16 pts) Construct the LR parsing table for the following grammar (DO NOT change the grammar.) A new start symbol $S'$ and production have already been added to the grammar.

1) $S' \rightarrow S$
2) $S \rightarrow aAB$
3) $A \rightarrow Bc$
4) $A \rightarrow \lambda$
5) $B \rightarrow bB$
6) $B \rightarrow \lambda$

(a) Calculate the FIRST and FOLLOW sets of variables.

<table>
<thead>
<tr>
<th></th>
<th>FIRST</th>
<th>FOLLOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$A$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Construct the transition diagram of the DFA that models the stack. Number the states, show marked productions, and identify final states by two circles.

(c) Construct the LR parse table that corresponds to the transition diagram drawn in part b. (Note: all the rows and columns given may not be needed. If there are multiple items for an entry, write all in the entry.)
10. (8 pts) Consider the following L-system.

Axiom: g X g
X → g Y [ + X + g ] Y
Y → f g

angle 45
color black
lineWidth 2
distance 10

Recall that g is for drawing a line, f is for moving forward, + means change the direction by the angle clockwise, and [ ] are used for stacking operations.

Assume a g drawn with distance 10 and lineWidth 2 is about this size | a. Give the next two strings in the language.
b. Render the L-system showing the picture for the axiom and the first two strings if there is a visual picture.

11. (6 pts) **Pumping Lemma for CFL’s** Let \( L \) be any infinite CFL. Then there is a constant \( m \) depending only on \( L \), such that for every string \( w \) in \( L \), with \( |w| \geq m \), we may partition \( w = uvxyz \) such that:

\[
|vxy| \leq m, \text{ (limit on size of substring)} \\
|vy| \geq 1, \text{ (} v \text{ and } y \text{ not both empty)} \\
\text{For all } i \geq 0, \ uv^i xy^i z \in L
\]

Prove that \( L = \{a^nb^kc^p \mid k > n, k > p\} \), is not a context-free language.
You only have to fill in the parts below. Assume \( L \) is a context-free language.

(a) Choose \( w = \)

(b) Prove the case when \( v = a^{t_1} \) and \( y = a^{t_2} \) (both are strings of a’s)

(c) Prove the case when \( v = b^{t_1} \) and \( y = c^{t_3} \) and there must be \( b \)'s and \( c \)'s (thus \( t_1 > 0 \) AND \( t_3 > 0 \), when either one is 0, another case is formed)

12. (5 pts) Here is the algorithm for transforming a CFG into CNF.

(a) Remove \( \lambda \)-productions, unit productions, and useless productions.

(b) For every rhs of length > 1, replace each terminal \( x_i \) by a new variable \( C_j \) and add the production \( C_j \rightarrow x_i \).

(c) Replace every rhs of length > 2 by a series of productions, each with rhs of length 2. QED.

Convert the following grammar into CNF. Note that the first step has already been done for you, there are no \( \lambda \)-productions, unit productions or useless productions.

\[
S \rightarrow ABAB \\
A \rightarrow a \\
B \rightarrow bBb \mid b
\]

13. (9 pts) Construct a one-tape TM acceptor (using a transition diagram) for the following language. \( L = \{w \in \Sigma^* \mid \text{for every } a \text{ there is a } b \text{ to its right that can be matched only with this } a\} \). \( \Sigma = \{a, b\} \).

For example, \( ababba \) is NOT in \( L \) since the last \( a \) does not have a \( b \) to its right it can be matched with. Also NOT in \( L \) are \( ba, abaab \) and \( bbabaa \). In \( L \) are \( aaabbb, aabbab \) and \( aabbab \).

In drawing the transition diagram, remember to identify the start state by an arrow and final states by double circles. Format of labels are \( a; b, R \) where \( a \) is the symbol read on the tape, \( b \) is the symbol written to the tape and \( R \) is the direction moved (you can use \( L \) and \( R \) for directions.) Make sure the tape head is pointing to the leftmost symbol of the output. Assume \( |w| = n \). What is the worst case running time (big-Oh) of your TM?
14. (9 pts) Construct a TM (using building blocks) for the following function. \( f(w) = w' \)
where \( w' \) is \( w \) with each \( a \) doubled. For example, \( f(aba) = (aabaa) \) and \( f(baab) = baaaab \).

See the building block notation on the next page.

Assume \(|w| = n\). What is the running time in terms of \( n \) (big-Oh) of your TM?

Notation for Simplifying Turing Machines

Suppose \( \Gamma = \{a,b,c,B\} \)

\( z \) is any symbol in \( \Gamma \)

\( x \) is a specific symbol from \( \Gamma \)

1. \( s \) - start
2. \( R \) - move right
3. \( L \) - move left
4. \( x \) - write \( x \) (and don’t move)
5. \( R_a \) - move right until you see an \( a \)
6. \( L_a \) - move left until you see an \( a \)
7. \( R_{\neg a} \) - move right until you see anything that is not an \( a \)
8. \( L_{\neg a} \) - move left until you see anything that is not an \( a \)
9. \( h \) - halt in a final state
10. \( a,b \rightarrow \{w\} \rightarrow \)

If the current symbol is \( a \) or \( b \), let \( w \) represent the current symbol.