1. (12 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) \( L = \{a^n b^{2n} c^{3n} \mid n > 0 \},\ \Sigma = \{a, b, c\}. \) \( \text{NOT} \)

(b) \( L = \{a^n b^{2m} \mid m > 0, n > 0\},\ \Sigma = \{a, b\}. \) \( \text{REG} \)

(c) \( L = \{b^na^m c^p \mid m > n+p, n > 0, p > 0, \text{and } n \text{ is even} \},\ \Sigma = \{a, b, c\}. \) \( \text{CFL} \)

(d) \( L = \{b^n c^p \mid p > 2n, n > 0, p < 500\},\ \Sigma = \{b, c\}. \) \( \text{REG} \)

(e) \( L = \{w \in \Sigma^* \mid 10 + n_b(w) > n_a(w) > n_b(w)\},\ \Sigma = \{a, b\}. \) \( \text{CFL} \)

(f) \( L = \{w \in \Sigma^* \mid n_a(w) \neq n_b(w)\},\ \Sigma = \{a, b\}. \) \( \text{CFL} \)

2. (14 pts) Answer TRUE or FALSE to each of the statements below.

(a) Suppose \( G \) is a CFG with 3 or more variables. Then there exists a TM \( M \) such that \( L(M) = L(G) \) and \( M \) halts on all inputs. (TRUE or FALSE?)

(b) Consider a CFG \( G \). Removing unit productions could create useless productions in the revised grammar. (TRUE or FALSE?)

(c) If \( M \) is a TM with fewer than 10 transitions, then there exist a CFG \( G \) such that \( L(M) = L(G) \). (TRUE or FALSE?)

(d) In a TM, the blank symbol \( \in \Sigma \). (TRUE or FALSE?)

(e) A DFA \( M \) on input string \( w, \ | w | = n, \) takes worst case time \( O(n) \) to decide to accept or reject \( w \). (TRUE or FALSE?)

(f) In \( LR(1) \) parsing, the \$ is always shifted onto the parsing stack exactly once. (TRUE or FALSE?)

(g) Given a CFG \( G \), suppose you are told that either LL or LR parsing is used to parse a string in \( L(G) \). If the derivation is \( S \Rightarrow AcB \Rightarrow Acb \Rightarrow acb \) then LR was the parsing method used. (TRUE or FALSE?)
3. (4 pts) Let M be a TM, M=(Q,Σ,Γ,δ,q₀,B,F). Explain how this formal definition differs from the formal definition of a DFA.

- Π is the tape alphabet that may include symbols not in the input alphabet. A DFA only has one alphabet Σ.
- The B is the blank symbol which is needed for the TM. For TM, $S$ is different. It includes the symbol to write and the direction to move.

4. (4 pts) Consider the following grammar. Using this grammar, give one string and its derivation that shows both $c$ immediately following $C$ and $b$ immediately following $A$ at different parts of the derivation.

\[ S \rightarrow AAB \]
\[ A \rightarrow aAC \mid \lambda \]
\[ B \rightarrow bB \mid c \]
\[ C \rightarrow BB \]

\[ S \Rightarrow AAB \Rightarrow aACAB \Rightarrow aACB \Rightarrow aACB \square aABBc \Rightarrow aABBBCc \]

5. (3 pts) Give a context-free grammar that is in Chomsky Normal Form (CNF) that has three rules and shows an example of all types of right-hand sides for this form.

\[ S \rightarrow AB \]
\[ A \rightarrow a \]
\[ B \rightarrow b \]

6. (3 pts) The following grammar is LL(k) for what value of k? Give the value of k and an example of a string that needs that many lookaheads.

\[ S \rightarrow BAC \mid bcb \]
\[ A \rightarrow aA \mid \lambda \]
\[ B \rightarrow Bb \mid bc \]
\[ C \rightarrow aa \]

\[ k=4 \quad \text{bcb} \]
\[ \text{or} \quad \text{bcbaa} \]
7. (10 pts) Consider the following grammar (DO NOT change the grammar):

\[
S \rightarrow ABcB \\
A \rightarrow aA \mid \lambda \\
B \rightarrow Ab \mid cBb
\]

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>a, b, c</td>
<td>$</td>
</tr>
<tr>
<td>A</td>
<td>a, \lambda</td>
<td>a, b, c</td>
</tr>
<tr>
<td>B</td>
<td>a, b, c</td>
<td>b, c, $</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.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th></th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>AbCB</td>
<td>AbCB</td>
<td>AbCB</td>
<td></td>
<td>$</td>
</tr>
<tr>
<td>A</td>
<td>aA, \lambda</td>
<td>\lambda</td>
<td>\lambda</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>Ab</td>
<td>Ab</td>
<td>cBb</td>
<td></td>
<td>$</td>
</tr>
</tbody>
</table>

(c) Is this grammar an LL(1) grammar? Explain.

no, there are two entries in the table in T[\{A, a\}].
8. (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.

0) S' → S  
1) S → BAc  
2) A → Ba  
3) A → a  
4) A → λ  
5) B → b

(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>b</td>
<td>#</td>
</tr>
<tr>
<td>A</td>
<td>a, b</td>
<td>c</td>
</tr>
<tr>
<td>B</td>
<td>b</td>
<td>a, c</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.)

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>$</th>
<th>5</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td>s3</td>
<td></td>
<td>1</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>acc</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>s7</td>
<td>s3</td>
<td>r4</td>
<td>4</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>r5</td>
<td>r5</td>
<td>r5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>s5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>r1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>s8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td>r3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9. (8 pts) Consider the following L-system.

Axiom: \( g \ f \ g \ X \)
\( X \rightarrow X \ [\ - \ g \ ] \ g \ + \ Y \)
\( Y \rightarrow g \)

angle 90
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, \(-\) means change the direction by the angle counterclockwise and \([\ ]\) are used for stacking operations.

Assume a \( g \) drawn with distance 10 and lineWidth 2 is about this size |

a. Render the L-system and draw the axiom if there is a visual picture for it.

\[
\begin{array}{c}
gfgX \\
| \\
| \end{array}
\]

b. Give the first string in the language (after the axiom) and draw it.

\[
\begin{array}{c}
gfgX[-g]g+Y \\
| \\
| \end{array}
\]

b. Give the second string in the language (after the axiom) and draw it.

\[
\begin{array}{c}
gfgX[-g]g+Y[-g]g+g \\
| \\
| \end{array}
\]
10. (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:

- $|vxy| \leq m$, (limit on size of substring)
- $|vy| \geq 1$, ($v$ and $y$ not both empty)
- For all $i \geq 0$, $uv^ixy^iz \in L$

Consider $L = \{a^pb^nc^p \mid p < n, n > 0\}$, $\Sigma = \{a, b, c\}$.

Prove $L$ is not a context-free language.

You only have to fill in the parts below. Assume $L$ is a context-free language.

(a) Choose $w = a^m b^2a c^{m-1}$

(b) Prove the case when $v = a^{t_1}$ and $y = b^{t_2}$

$$i = 0 \Rightarrow u^i v^i x^i y^i z^i = a^{m-t_1} b^{2m+t_2} c^{m-1} \notin L$$

$t_1 \neq 0$ since $n_a(w) \leq n_c(w)$

$t_1 = 0$ $2 \times n_a(w) \neq n_b(w)$

(c) Prove the case when $v = b^{t_1}$ and $y = c^{t_3}$

$$i = 2 \Rightarrow u^i v^i x^i y^i z^i = a^m b^{2m+t_1} c^{m-1+t_3} \notin L$$

$t_2 > 0$ since $n_a(w) \geq n_a(w)$

$t_2 = 0$ since $2 \times n_a(w) \neq n_b(w)$
11. (10 pts) Construct a one-tape TM (using a transition diagram) that accepts the following language:

$L = \{ w \in \Sigma^* \mid w \text{ is of even length, } |w| \geq 2, \text{ and the middle two characters are not the same } \} \Sigma = \{a, b\}$. For example, $abbabb$ would be accepted (the middle two characters are $ba$ and they are different), $aaabbbaa$ would be rejected (the middle two characters are the same $bb$), and $aaa$ would be rejected (odd length).

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.)

Assume $|w| = n$. What is the worst case running time (big-Oh) of your TM? $\mathcal{O}(n^2)$
12. (10 pts) Construct a TM (using building blocks) for the following function. \( w \in \{a, b\}^*, \quad |w| \geq 2, \ f(w) = w' \) (where \( w' \) is the same as \( w \) but with the first and last letters switched).

For example, \( f(ababb) = bbaba \) and \( f(aabba) = aabba \).

See the building block notation on the next page. Make sure the tape head is pointing to the leftmost symbol of the output.

Another Soln:

Assume \(|w| = n\). What is the running time in terms of \( n \) (big-Oh) of your TM? \( O(n) \)
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 $a$ (note that this moves right at least one square before it checks for $a$).
6. $L_a$ - move left until you see an $a$
7. $R_a|b$ - move right until you see an $a$ or $b$
8. $L_a|b$ - move left until you see an $a$ or $b$
9. $R_{\neg a}$ - move right until you see anything that is not an $a$
10. $L_{\neg a}$ - move left until you see anything that is not an $a$
11. $h$ - halt in a final state
12. $\frac{a,b}{\Rightarrow}$

If the current symbol is $a$ or $b$, let $w$ represent the current symbol.

13. $C$ - copy a string, $F(w) = w0w$, makes a copy of the string and inserts a 0 between the original and the copy.

14. $S_L$ - Shift the string that is to the right of the tape head (up to a blank), to the left, writing over the symbol the tape head is pointing to.

15. $S_R$ - Shift the string that is to the left of the tape head (up to a blank), to the right, writing over the symbol the tape head is pointing to.