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

(b) \( L = \{(a^n b^n)^* \mid n > 0\}, \Sigma = \{a, b\}. \quad \text{NOT} \)

(c) \( L = \{a^n b^m c^p d^t \mid n = p, m = t, n > 0, m > 0, p > 0, t > 0\}, \Sigma = \{a, b, c, d\}. \quad \text{NOT} \)

(d) \( L = \{w \in \Sigma^* \mid n_a(w) > n_b(w) > n_c(w), n_a(w) > 0, 50 > n_b(w) > 0, n_c(w) > 0\}, \Sigma = \{a, b, c\}. \quad \text{REG} \)

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

(f) \( L = \{a^n b^m c^{2n} \mid n > 100 \text{ and } m \text{ is divisible by } n\}, \Sigma = \{a, b, c\}. \quad \text{NOT} \)

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

(a) Suppose \( M_1 \) is an NPDA and \( M_2 \) is a Turing machine, then \( L(M_1 \cap M_2) \) is a context-free language. (TRUE or FALSE?)

(b) Suppose \( M_1 \) is a CFG, then there exists a Turing machine \( M_2 \) such that \( L(M_1) = L(M_2) \) and \( M_2 \) always halts. (TRUE or FALSE?)

(c) A bottom-up parser shifts the symbols from the input string on the parsing stack from right to left. (TRUE or FALSE?)

(d) If the terminal \( b \) is in the Follow set of the variable \( A \), then there must be a production in which \( Ab \) is a substring of the right hand side of the production. (TRUE or FALSE?)

(e) A production can appear in one LR(1) item set more than once with the marker in different places. (TRUE or FALSE?)
(f) Removing unit productions from a grammar can generate useless productions.  
\( \text{TRUE} \) or FALSE?

3. (3 pts) Given a context-free grammar that is an LR(1) grammar, explain why the equivalent PDA may be nondeterministic while the LR(1) parsing method for the grammar is deterministic.

\[ \text{The LR(1) parsing method uses additional information, a look-ahead, to decide which rule to apply. This extra information makes the parsing process deterministic. The PDA doesn't use the look-aheads.} \]

4. (4 pts) Consider the following grammars and list all forms they are in (CFG, GNF, CNF, and/or regular grammar).

\[
\begin{align*}
(a) & \quad S \rightarrow aB | b \\
& \quad B \rightarrow bBB | c \\
B & \quad S \rightarrow ABC \\
& \quad A \rightarrow AA | a \\
& \quad B \rightarrow b \\
& \quad C \rightarrow c
\end{align*}
\]

5. (3 pts) Explain why there are two alphabets in the formal definition of a TM and what the difference is between the two.

One alphabet is the tape alphabet, any symbol that can write to the tape. The other alphabet is the input alphabet, only symbols that can be in the input string.

6. (3 pts) The following grammar is LL(k) for what value of k?

\[
\begin{align*}
S & \rightarrow aAc | BaAdBA \\
A & \rightarrow aA | \lambda \\
B & \rightarrow bB | c
\end{align*}
\]

7. (3 pts) Context-free languages are not closed under intersection. Give an example of two different CFL's L1 and L2 such that L1 \( \cap \) L2 is NOT a CFL.

\[
L_1 = \{a^m b^n c^m | m \geq n, m \geq 0\} \quad L_2 = \{a^n b^m c^n | n > m, n \geq 0\}
\]

8. (10 pts) Consider the following grammar (DO NOT change the grammar):
\[ S \rightarrow ABC \]
\[ A \rightarrow aSBa | \lambda \]
\[ B \rightarrow bB | dAd | \lambda \]

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

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

\[
\begin{array}{c|c|c}
 & \text{FIRST} & \text{FOLLOW} \\
\hline
S & a, b, d, c & \$, d, b, a \\
A & a, \lambda & b, d, c \\
B & b, d, \lambda & c, a \\
\end{array}
\]

(b) Calculate all entries in the LL(1) Parse Table. If there are multiple rules for an entry, write in all the rules.

\[
\begin{array}{c|c|c|c|c|c}
 & a & b & c & d & \$
\hline
S & ABC & ABC & ABC & ABC \\
A & aSBa & \lambda & \lambda & \lambda \\
B & \lambda & bB & \lambda & dAd \\
\end{array}
\]

(c) Is this grammar an LL(1) grammar? Explain. \textit{yes, there are no conflicts in the table}

9. (15 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 AB  
3) S \rightarrow a  
4) A \rightarrow SSd  
5) B \rightarrow b  
6) B \rightarrow \lambda

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

\[
\begin{array}{c|c|c|c|c}
 & \text{FIRST} & \text{FOLLOW} \\
\hline
S & a & \$, d, a \\
A & a & b, d, d, a \\
B & b, \lambda & \$, d, a \\
\end{array}
\]

3
(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>d</th>
<th>$</th>
<th>A</th>
<th>B</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>s3</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>r6</td>
<td>s5</td>
<td>r6</td>
<td>r6</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>s3</td>
<td></td>
<td>acc</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>r3</td>
<td>r3</td>
<td>r3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>r2</td>
<td>r2</td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>r5</td>
<td>r5</td>
<td>r5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>s3</td>
<td>s7</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>r4</td>
<td>r4</td>
<td>r4</td>
<td>r4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></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>

10. (8 pts) Consider the following L-system.

Axiom: X g + g  
X → g Y [+ g] [- g] X  
Y → g 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, $-\,$ 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.

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

\[ g, \gamma [+g][-g] \times g + g \]

C. \(X\). Give the second string in the language and draw it. \((\text{see next page})\)

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:

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

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

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 = $ see next page

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

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

12. (9 pts) Construct a one-tape TM transducer (using a transition diagram) that computes the following function. Assume $w \in \{a, b\}^+$ and is of even length, $f(w) = w'$ where $w'$ is $w$ with every pair of symbols swapped. For example, $f(abbab) = baabbb$ (the first two symbols $ab$ are swapped to $ba$, the next two symbols $ba$ are swapped to $ab$ and the last two symbols $bb$ are swapped to $bb$), and $f(aaabbba) = aababbab$.

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?
10c. \( gg + g [-g] g Y [+g][-g] X g + g \)

11. a) \( w = a^m b^{3m} c^{m-1} \)
   
   b) \( v = a^{t_1}, \; y = b \)

   \( i = 0 \)

   \( a^m b^{3m-t_2} c^{m-t_2} \) 
   
   Since \( n_a(w) \leq n_c(\omega) \)

   \( i = 2 \)

   \( a^m b^{3m+t_1} c^{m+t_3-1} \) 
   
   Since either \( t_1 > 0 \) and

   \( n_a(w) \neq 3 \times n_b(w) \) or

   \( t_1 = 0 \) and \( n_c(w) \geq n_a(w) \)
12.

\[ O(n) \]
13. (9 pts) Construct a TM (using building blocks) for the following function. $w \in \{0, 1\}^+$ and contains more 0's than 1's, $f(w) = w\#p$ where $p$ is the number (in unary) of 0's more than 1's in the original string.

For example, $f(10010) = (10010\#1)$ (1 more 0 than 1's) and $f(011000101100) = 011000101100\#11$ (2 more 0's than 1's).

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. $a \rightarrow b \rightarrow w$

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

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

12. $S_L$ - 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.

13. $S_R$ - 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.