1. Answer TRUE or FALSE to each of the following questions.

(a) (2 pts) Every CFG grammar is an LR(1) grammar. (TRUE or FALSE?)
(b) (2 pts) If $L$ is a regular language then there exists a DPDA $M$ such that $L=L(M)$. (TRUE or FALSE?)
(c) (2 pts) If $L$ is a context-free language then there exists a DPDA $M$ such that $L=L(M)$. (TRUE or FALSE?)
(d) (2 pts) An LR(k) parser produces a leftmost derivation in reverse order. (TRUE or FALSE?)
(e) (2 pts) Every context-free grammar that is ambiguous can be rewritten as an unambiguous grammar. (TRUE or FALSE?)

2. (12 pts) Write a CFG for each of the following languages.

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

3. (10 pts) Here is the algorithm for removing $\lambda$-productions from a CFG.

**To Remove $\lambda$-productions**

(a) Let $V_n = \{A | \exists \text{ production } A \rightarrow \lambda \}$
(b) Repeat until no more additions
   - if $B \rightarrow A_1 A_2 \ldots A_m$ and $A_i \in V_n$ for all $i$ then put $B$ in $V_n$
(c) Construct $G'$ with productions $P'$ s.t.
   - If $A \rightarrow x_1 x_2 \ldots x_m \in \{\Gamma \}$ then put all productions formed when $x_j$ is replaced by $\lambda$ (for all $x_j \in V_n$) s.t. $|\text{rhs}| \geq 1$ into $P'$.

**Using the algorithm above, remove $\lambda$-productions** from the following grammar.

Give the set $V_n$ and the resulting grammar.

- $S \rightarrow ABD | aSC$
- $A \rightarrow abA | \lambda$
- $B \rightarrow baB | b | \lambda$
- $C \rightarrow AD | AC$
- $D \rightarrow ABA | d$
4. (12 pts) Consider \( L = \{a^n b^m c^n \mid n > 0, m > 0\} \). 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 the two smallest strings in \( L \).

(b) Now draw the transition diagram.

5. (15 pts) Following is the definition of \( \delta \) for an NPDA that we saw in Chapter 7.

\[ \text{Definition 1.} \quad \delta : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow \text{finite subsets of } Q \times \Gamma^* \]

(a) Explain each symbol in the above definition (\( Q, \Sigma, \Gamma \) and \( \delta \)) and explain what the definition means?

(b) \textbf{Definition 2.} Suppose the definition of \( \delta \) above is changed to allow one to pop zero or more symbols from the stack. Give the new definition (in formal notation as above).

(c) Prove that for any NPDA \( M \) constructed using Definition 2 there exists an NPDA \( M' \) constructed using Definition 1 such that \( L(M) = L(M') \).

6. (6 pts) An NPDA \( M = (Q, \Sigma, \Gamma, \delta, q_0, \emptyset, F) \) is a 7-tuple where each part was defined in lecture and \( \delta \) was defined in the previous problem (Definition 1).

Suppose we create a new machine \( \Gamma \) a PDA with 2 stacks. Define a 2-stack NPDA \( \Gamma \) as a \( k \)-tuple (what is \( k \)?) and explain what each of the \( k \) parts is including giving the new definition of \( \delta \) using formal notation.

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

\[
\begin{align*}
S & \rightarrow aB \mid aC \\
B & \rightarrow aB \mid bc \\
C & \rightarrow abC \mid a
\end{align*}
\]

8. (12 pts) Consider the following grammar (DO NOT change the grammar):

\[
\begin{align*}
S & \rightarrow BSA \mid Ac \\
A & \rightarrow aS \mid \lambda \\
B & \rightarrow bB \mid d
\end{align*}
\]

(a) Calculate FIRST and FOLLOW for the variables in the grammar.
(b) Calculate all entries in the LL(1) Parse Table. If there are multiple rules for an
case, write in all the rules.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

9. (20 pts) Construct the LR parsing table for the following grammar. A new start symbol
S' and production have already been added to the grammar.

1) S' → S 3) S → aAS
2) S → λ 4) A → bb

(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>
</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, put both.)