Instructions: Do not put your name on the exam, please answer all the questions directly on the exam itself. Answer all the questions. Explain answers as fully as possible, give examples or define terms, if appropriate.

1. Do all reasonable programming languages have a LALR(1) grammar? Explain.
2. Convert the following NFA over the $\Sigma = \{a, b\}$ to a DFA using the subset construction. The start state of the NFA, marked by a triangle, is 0; the only final state, marked by double lines, is 13. Your result should have five states. Label your states with capital letters $A$, $B$, $C$, $D$, and $E$ and fill in the table below so that the correspondence is clear between the states of your DFA and sets of the NFA’s state labels. Fill in the table with the transition on each state of your DFA. Do not simplify.

<table>
<thead>
<tr>
<th>$DFA$</th>
<th>$NFA$</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$D$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$E$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]

\[ \epsilon \]
3. Rewrite the entire grammar below eliminating left recursion.

\begin{align*}
1 \quad S & \rightarrow A \\
2 \quad A & \rightarrow AaBd \\
3 \quad A & \rightarrow AbB \\
3 \quad A & \rightarrow Ac \\
4 \quad A & \rightarrow aA \\
5 \quad A & \rightarrow Bd \\
6 \quad B & \rightarrow b \\
7 \quad B & \rightarrow Bb
\end{align*}
4. Consider the following augmented grammar over the alphabet \{a, b, c\}.

\[
\begin{align*}
0 & \quad S' \to S \$
1 & \quad S \to A \\
2 & \quad S \to cb \\
3 & \quad A \to aAb \\
4 & \quad A \to B \\
5 & \quad B \to c
\end{align*}
\]

(a) Compute nullable, FIRST, and FOLLOW for all nonterminals of the grammar.

<table>
<thead>
<tr>
<th>nullable</th>
<th>FIRST</th>
<th>FOLLOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>S'</td>
<td></td>
<td></td>
</tr>
<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) For all productions, compute the FIRST of the right-hand side, or the FOLLOW of the left-hand side, as appropriate for computing the LL(1) parsing table.

<table>
<thead>
<tr>
<th>N \to \alpha</th>
<th>null(N)?</th>
<th>FIRST(\alpha)</th>
<th>FOLLOW(N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 S' \to S $</td>
<td>null(N)?</td>
<td>FIRST(\alpha)</td>
<td>FOLLOW(N)</td>
</tr>
<tr>
<td>1 S \to A</td>
<td>null(N)?</td>
<td>FIRST(\alpha)</td>
<td>FOLLOW(N)</td>
</tr>
<tr>
<td>2 S \to cb</td>
<td>null(N)?</td>
<td>FIRST(\alpha)</td>
<td>FOLLOW(N)</td>
</tr>
<tr>
<td>3 A \to aAb</td>
<td>null(N)?</td>
<td>FIRST(\alpha)</td>
<td>FOLLOW(N)</td>
</tr>
<tr>
<td>4 A \to B</td>
<td>null(N)?</td>
<td>FIRST(\alpha)</td>
<td>FOLLOW(N)</td>
</tr>
<tr>
<td>5 B \to c</td>
<td>null(N)?</td>
<td>FIRST(\alpha)</td>
<td>FOLLOW(N)</td>
</tr>
</tbody>
</table>
(c) Complete the *entire* LL(1) parse table below.

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

(d) Is the grammar LL(1)? Please circle the correct answer: yes / no. Explain.
5. Consider the algorithm to compute \( \text{CLOSE}[I] \) for the set of LR(1) items \( I \) for some grammar. Suppose the grammar contains the production \( X \rightarrow \gamma \) where \( X \) is some non-terminal and \( \gamma \) is some string of terminals and non-terminals. Answer the following questions assuming \( A \) is some non-terminal, \( \alpha \) and \( \beta \) are strings of terminals and non-terminals, and \( y \) and \( z \) are terminal symbols.

(a) If \( A \rightarrow \alpha \bullet X, z \) is in \( I \), which item or items (if any) would be added to \( \text{CLOSE}[I] \)?

(b) If \( A \rightarrow \alpha \bullet X y, z \) is in \( I \), which item or items (if any) would be added to \( \text{CLOSE}[I] \)?

(c) If \( A \rightarrow \alpha \bullet X \beta, z \) is in \( I \), which item or items (if any) would be added to \( \text{CLOSE}[I] \)?
6. Consider the following grammar.

1   \( S \rightarrow \text{id} \)
2   \( S \rightarrow V := E \)
3   \( V \rightarrow \text{id} \)
4   \( E \rightarrow V \)
5   \( E \rightarrow n \)

Add the augmenting production. Build the state transition diagram and parsing table for LR(1) parsing. Is the grammar LR(1)? Please circle the correct answer: yes / no.