NAME:__________________________________________

ANSWER ALL QUESTIONS ON THE EXAM SHEETS. IF YOU ARE UNSURE ABOUT THE MEANING OF A QUESTION, INDICATE THE ASSUMPTIONS YOU MAKE IN ANSWERING IT. APPROPRIATE CREDIT WILL BE GIVEN FOR REASONABLE ASSUMPTIONS. POINT VALUES FOR EACH QUESTION ARE GIVEN IN PARENTHESES.

Note: in all example grammars, the terminal symbols are given in boldface.
1. a. (5) Explain why the following grammar is not LL(k) for any k:

   \[
   \begin{align*}
   S & \rightarrow A \mid B \\
   A & \rightarrow aaA \mid aa \\
   B & \rightarrow aaB \mid a \\
   \end{align*}
   \]

b. (5) Explain why the following grammar is LL(1) but not SLR(1):

   \[
   \begin{align*}
   S & \rightarrow AaAb \mid BbBa \\
   A & \rightarrow \varepsilon \\
   B & \rightarrow \varepsilon \\
   \end{align*}
   \]

c. (5) Which of LL(1), SLR(1), and LR(1) can parse strings in the following grammar, and why?

   \[
   \begin{align*}
   E & \rightarrow A \mid B \\
   A & \rightarrow a \mid c \\
   B & \rightarrow b \mid c \\
   \end{align*}
   \]
2. (10) Consider the following two BNF grammars. In both cases:
- the terminals are \{ A, B, C, D, E, F, * \},
- the non-terminals are \{ <expr>, <term>, <identifier> \}, and
- the start symbol is <expr>.

For each grammar draw a parse tree for the expression

\[ A \times B \times C \times D \times E \times F \]

If the grammar is ambiguous, state that it is ambiguous and give the reason for the ambiguity.

Grammar A:
\[
<\text{expr}> ::= <\text{term}> \mid <\text{expr}> <\text{term}>
\]
\[
<\text{term}> ::= <\text{identifier}> \mid <\text{identifier}> * <\text{term}>
\]
\[
<\text{identifier}> ::= A \mid B \mid C \mid D \mid E \mid F
\]

Grammar B:
\[
<\text{expr}> ::= <\text{term}> \mid <\text{term}> <\text{expr}>
\]
\[
<\text{term}> ::= <\text{identifier}> \mid <\text{term}> * <\text{term}>
\]
\[
<\text{identifier}> ::= A \mid B \mid C \mid D \mid E \mid F
\]
3. (10) Write the actions of an LR parse for the following string, for the grammar and parse table shown below:

aa1bbbb

Grammar:

(1) \( S \rightarrow A \)
(2) \( S \rightarrow B \)
(3) \( A \rightarrow aA \ b \)
(4) \( A \rightarrow 0 \)
(5) \( B \rightarrow aB \ b \ b \)
(6) \( B \rightarrow 1 \)

<table>
<thead>
<tr>
<th>State</th>
<th>a</th>
<th>b</th>
<th>0</th>
<th>1</th>
<th>$</th>
<th>S</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>S1</td>
<td></td>
<td>S2</td>
<td>S3</td>
<td>11</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>S1</td>
<td>r4</td>
<td></td>
<td>r4</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>r6</td>
<td>r6</td>
<td>r6</td>
<td>r6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>r1</td>
<td>r1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>r2</td>
<td>r2</td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>S8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>S9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>r3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>S10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>r5</td>
<td></td>
<td>r5</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>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td>acc</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Stack</th>
<th>Input</th>
<th>Action</th>
</tr>
</thead>
</table>
4. This question concerns the context-free grammar given below (capital letters denote non-terminals, small letters denote terminals, $\epsilon$ denotes the empty string, and $S$ denotes the start non-terminal)

(1) $S \rightarrow W A B \mid A B C S$
(2) $A \rightarrow B \mid W B$
(3) $B \rightarrow \epsilon \mid y B$
(4) $C \rightarrow z$
(5) $W \rightarrow x$

a. (10) Fill in the following table with the First and Follow sets for the grammar (i.e., the First sets of the non-terminals, production right-hand sides, and suffixes of right-hand sides, and the Follow sets of the non-terminals).

<table>
<thead>
<tr>
<th>X</th>
<th>First(X)</th>
<th>Follow(X)</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>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WAB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABCS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>yB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
b. (10) Using the First and Follow sets in part (a), fill in the LL(1) parse table below:

<table>
<thead>
<tr>
<th></th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>EOF</th>
</tr>
</thead>
<tbody>
<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>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

c. (5) Is the grammar LL(1)? Justify your answer.
5. You are given the following grammar with the terminal symbols (, ), and `term` and non-terminals `S`, `E` and `L`.

   (1) \( S \rightarrow E \) $  
   (2) \( E \rightarrow \text{term} \)  
   (3) \( E \rightarrow ( L ) \)  
   (4) \( L \rightarrow \epsilon \)  
   (5) \( L \rightarrow E \ L \)  

   a. (5) If the terminal `term` matches the character \( x \), give three well formed strings in this grammar.

   b. (10) A nearly complete collection of LR(0) sets of items for the above grammar is given below. Fill in the missing items and GOTO information. Be sure you complete the entire set, including all necessary GOTOS.

   \[
   \begin{align*}
   S: \{ & S \rightarrow \cdot E \\
   & \ E \rightarrow \cdot \text{term} \\
   & \ E \rightarrow \cdot ( L ) \} \\
   S: \text{GOTO}(S,E) \\
   \} \quad & S: \text{GOTO}(S,E) \\
   S: \text{GOTO}(S,\text{term}) \\
   \} \quad & S: \text{GOTO}(S,\text{term}) \\
   S: \text{GOTO}(S,\text{ (}) \\
   \} \quad & S: \text{GOTO}(S,\text{ (}) \\
   S: \text{GOTO}(S,\text{ )}) \\
   \} \quad & S: \text{GOTO}(S,\text{ )}) \\
   \text{GOTO}(S,\text{ (}) = S.
   \end{align*}
   \]
c. **(10)** Using the items in part (b), fill in the missing information for states 5 and 7 in the SLR(1) parse table. Note that the numbers in the reductions refer to the productions (1 – 5) as given above.

<table>
<thead>
<tr>
<th></th>
<th>(   )</th>
<th>term</th>
<th>$</th>
<th>E</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Shift 4</td>
<td>error</td>
<td>Shift 3</td>
<td>error</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>error</td>
<td>error</td>
<td>error</td>
<td>Accept</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Reduce 2</td>
<td>Reduce 2</td>
<td>Reduce 2</td>
<td>Reduce 2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Shift 4</td>
<td>Reduce 4</td>
<td>Reduce 4</td>
<td>Shift 3</td>
<td>Reduce 4</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>error</td>
<td>Shift 7</td>
<td>error</td>
<td>error</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Reduce 5</td>
<td>Reduce 5</td>
<td>Reduce 5</td>
<td>Reduce 5</td>
<td></td>
</tr>
</tbody>
</table>
6. Consider the following grammar and associated semantic actions. In the actions, the operations \texttt{And}, \texttt{Or}, and \texttt{Not} are constructors for an abstract syntax tree data type.

\[
\begin{array}{c|c}
G \rightarrow F & G.p = F.p \\
F \rightarrow F_1 \land F_2 & F.p = \texttt{And}(F_1.p, F_2.p) \\
F \rightarrow F_1 \lor F_2 & F.p = \texttt{Or}(F_1.p, F_2.p) \\
F \rightarrow \neg F_1 & F.p = \texttt{Neg}(F_1.p) \\
F \rightarrow F_1 \Rightarrow F_2 & F.p = \texttt{Or}(\texttt{Not}(F_1.p), F_2.p) \\
F \rightarrow (F_1) & F.p = F_1.p \\
F \rightarrow \text{id} & F.p = \text{id.lexeme}
\end{array}
\]

a. (5) Say whether each attribute of a non-terminal is \textit{inherited} or \textit{synthesized} and why.

b. (5) Give the value of the attributes of \(G\) after parsing \(\neg (A \land (A \Rightarrow B))\).
7. (5) Consider the following grammar:

\[
\begin{align*}
S' & \rightarrow S \\
S & \rightarrow E = E \\
S & \rightarrow i \\
E & \rightarrow E + i \\
E & \rightarrow i 
\end{align*}
\]

The start state (set of LR(0) items) for this grammar is as follows:

\[
S_0 : \{ \quad S' \rightarrow \bullet S \\
S \rightarrow \bullet E = E \\
S \rightarrow \bullet i \\
E \rightarrow \bullet E + i \\
E \rightarrow \bullet i \quad \}
\]

Also we have

\[
S_3 = \text{GOTO}(S_0, i) \\
\{ \quad S \rightarrow \bullet i \\
E \rightarrow \bullet i \quad \}
\]

Because \( \text{FOLLOW}(S) = \{\$\} \) and \( \text{FOLLOW}(E) = \{\$,+,,\} \), we have a REDUCE-REDUCE conflict for state 3 in the SLR(1) parse table.

Give the corresponding sets of LR(1) items for \( S \) and \( S_0 \) which are used to construct the canonical LR parser, and explain why the conflict in the SLR(1) parse table is eliminated.