1. [20] Consider the following grammar G:

$$
\begin{aligned}
E &\rightarrow TE' \\
E' &\rightarrow +TE' \mid \epsilon \\
T &\rightarrow FT' \\
T' &\rightarrow *FT' \mid \epsilon \\
F &\rightarrow id
\end{aligned}
$$

   (a) Is G LL(1)? If yes, show the parse table. Otherwise, show why.

   (b) Is G SLR(1)? If yes, show the parse table. Otherwise, show why.

2. [20] Consider the following grammar G:

$$
\begin{aligned}
S' &\rightarrow S \\
S &\rightarrow Aa \mid bAc \mid dc \mid bda \\
A &\rightarrow d
\end{aligned}
$$

   (a) Is G SLR(1)? If yes, show the parse table. Otherwise, show why.

   (b) Is G LALR(1)? If yes, show the parse table. Otherwise, show why.

3. Consider the following grammar G:

$$
\begin{aligned}
S' &\rightarrow S \\
S &\rightarrow (S)S \mid \epsilon
\end{aligned}
$$

   (a) [10] Construct the collection of sets of LR(0) items for G

   (b) [10] Construct an NFA in which each state is an LR(0) item from (a)

   (a) [10] Show that the goto graph of the sets of LR(0) items in (a) is the same as the DFA contructed from the NFA in (b)

4. [10] The following grammar for `if-then-else` statement is proposed to remedy the dangling-else ambiguity:

$$
\begin{aligned}
\mathit{stmt} &\rightarrow \textbf{if } \mathit{expr} \textbf{ then } \mathit{stmt} \\
&\mid \mathit{matched\_stmt} \\
\mathit{matched\_stmt} &\rightarrow \textbf{if } \mathit{expr} \textbf{ then } \mathit{matched\_stmt} \textbf{ else } \mathit{stmt} \\
&\mid \textbf{other}
\end{aligned}
$$

   Show that this grammar is still ambiguous.

5. [10] Translate the expression $-(a+b)*(c+d)+(a+b+c)$ into

   (a) a syntax tree

   (b) postfix notation

   (c) tree-address code

6. Consider the follow extended assignment statement in C

   $id_1, id_2 = expr_1, expr_2$

   It has the same meaning as

   $id_1 = expr_1 ; id_2 = expr_2$

   Note that these two statements are executed simultaneously, not sequentially.

   (a) [10] Write down the semantic rules for intermediate code generation of the following grammar

   $$S \rightarrow id, id = E, E$$
   $$E \rightarrow E + E \mid E * E \mid id$$

   (b) [5] Translate a, b = b, a to 3-address code based on (a)

   (c) [5] Translate a, b = a + b + 10, a + b + 1 to 3-address code based on (a)