1. Consider the following grammar:

\[
\text{statement ::= if expr then statement else statement} \\
\text{\hspace{1cm} | if expr then statement} \\
\text{\hspace{1cm} | id = expr} \\
\text{\hspace{1cm} | id ( expr )} \\
\text{express ::= id | expr + id}
\]

Rewrite the grammar to be LL(1).

2. What is the follow set for the nonterminal expr?

3. Write a series of recursive descent routines to recognize the nonterminal statement.

4. Imagine that we are building a lexical analyzer for the three keywords int, if and integer. These three are the ONLY tokens. Show the finite state automata that would be created by a system such as Lex to recognize these tokens. Then explain the actions the system would perform if the input contains the text “integx”.

Translators Winter 98 Page 1
5. Draw a picture of the various symbol tables that would exist while parsing
the body of the procedure foo in the following; show in which symbol table
each variable name would be found. Then, describe the abstract syntax
tree that would be created for the expression e.z.

```plaintext
var a : int;
class B
begin
    var c : int;
    var z : int;
    function d ()
    var e : B;
        function foo ()
        var g : int;
        begin
            print (e.z);
    . . .
```