CS328, Winter 2003, Test 2

Time 60min. Use extra paper as needed, but make sure to identify each answer. Open notes. Five questions, each even points.

YOU MUST RETURN THIS PAGE. NAME___________________________

1 Design a DFA to recognize identifiers defined as follow:
   - identifier name starts with a letter and it can be any combination of letters and
decimal digits as long as the total number of letters is greater than the total number of
digits
   - identifier may start with white spaces and it is always followed by white spaces
   - the DFA should report whether the input is a valid identifier or error otherwise

   Example:
   x12yWS  error
   12xyzWS error
   xy12zWS ok
   a2yyyWS ok
If you have done it, skip the rest. If you think this cannot be done, explain why and
then assume that the constraint on the total number of letters vs. digits applies only
to the first 3 characters of a token - redo the problem now or argue why it still cannot
be done.

   NOTE: if you produce a DFA design, start with defining the alphabet, tokens, and
then the graph.

2 Given the production:
   S-> aSAc | Acb
   A-> bbb | empty

   implement a complete pseudocode for a recursive descent parser. Assume scanner()
function returns the next token and error() aborts processing with an error message.
Do not forget the main program. This grammar is LL(1) so no don’t modify.

3 Given
   S -> SabC | abC | aCa
   C -> ccC | c | empty | D
   D -> dd

   rewrite the grammar as LL(1) if possible or otherwise argue why it is not possible
   (and better have good arguments). Prove that it is indeed LL(1), after the
   modifications, showing only the sets that are needed and using them for your proof.

4 Suppose you have a language where a valid program is a sequence of assignments,
with each ending with a semicolon. An assignment has syntax and semantics as in
our language. Expressions can use variables and integers. Variables are not defined
just used. There are two predefined variables READ and WRITE. READ evaluates to
the standard input value, WRITE doesn’t evaluate to anything but it prints to the
output the value being assigned to it. Expressions are as follow: binary -,+,*,/,^, and
unary !. Expression can be parenthesized, which overrides any precedence.
Precedence is set as: weakest are + and -, then *, then /, then ^, then the unary.
Associativity is right to left for + and -, and left to right for all others. Write unambiguous CFG grammar.

Example statements
x:=READ+5;
WRITE:=x;
x=y+x/(x*5);

5 Suppose we want to modify our project grammar so that assignment is an operator as well, exactly like in C, including precedence and associativity. Modify the grammar as needed. Show only the modifications.

6 Extra Credit:
5pct) Using our language for the project, write a valid program that would read an input and then compute and output its factorial.
5pct) What are limitations of this program? Be very specific.