**CISC 672: ADVANCED COMPILER CONSTRUCTION**
**Spring 2005**
**"Fun with Grammars and Parsing" Homework**
**Due Date: classtime on March 24, 2005**

1. Grammars

   (a) Translate the following regular expression into a context-free grammar.

   $$((xy^*x)|(yx^*y))?$$

   (b) Construct an equivalent unambiguous grammar for the following grammar assuming operators # and $ are right associative and the same precedence, and concatenation and @ are left associative and higher precedence than # and $. * is highest precedence. The () are used to override precedence.

   $E := E\#E|E\$E|E@E|EE|E*|(E)|b$

2. Top Down Parsing

   (a) Give as many specific examples as you can find to explain why the following grammar is NOT LL(1). Apply techniques discussed in class (left factoring and left recursion removal) to convert the grammar to LL(1) without changing the language that it specifies. Is the grammar now LL(1) after applying these techniques? Justify your answer.

   $S := GT$
   $G := P|G;P$
   $P := id : R$
   $R := id|id, R$
   $T := T * D|T + D|\epsilon|D$
   $D := AB$
   $A := id.|\epsilon$
   $B := id|id(W)$
   $W := \epsilon|id$

   (b) Show the FIRST and FOLLOW sets, and LL(1) parse table for the following grammar.

   $S := (X|E]|F)$
   $X := E)|F]$
   $E := A$
   $F := A$
   $A := \epsilon$

3. Bottom Up Parsing

   (a) Consider the following grammar: (Note the ; is part of the first production.)

   $S := E;$
   $E := id|id(E)|E + id$

   (1) Build the LR(0) DFA for this grammar.
   (2) Is this an LR(0) grammar? Give evidence.
   (3) Is this an SLR(1) grammar? Give evidence.

(b) Consider the following grammar: (Note the ; is part of the first production.)

$S := X;$
$X := Ma|bMc|dc|bda$
$M := d$

(1) Build the LR(1) DFA for this grammar.
(2) Is the grammar LR(1)? Give evidence.
(3) Is the grammar LALR(1)? Give evidence.
(4) Is the grammar SLR(1)? Give evidence.

(c) Show that the grammar for which you built the LL(1) parse table above (starts with S := ( X ...) is NOT LALR(1).