CSE 401 - Homework Assignment #2

Due: Wednesday, January 30, 2008 (at the beginning of class)

1. Write an unambiguous grammar for each of the following languages. **Hint**: One way of verifying that a grammar is unambiguous is to run it through Yacc, the UNIX parsing tool, and get no conflicts.
    a. Balanced parentheses and square brackets. Example:
       ```
       ([[]](()[()][])])
       ```
    b. Statement blocks in Pascal or ML where the semicolons **separate** the statements:
       ```
       { statement ; {statement ; statement} ; statement }
       ```
    c. Statement blocks in C where the semicolons **terminate** the statements:
       ```
       { expression ; {expression ; expression; } expression; }
       ```

2. Write a grammar that accepts the straight line code language given below, but that is suitable for LL(1) parsing. That is, eliminate the ambiguity, eliminate the left recursion, and (if necessary) left-factor.
   ```
   S ::= S ; S    S ::= id := E   S ::= print( L )
   E ::= id        E ::= num       E ::= E + E
   E ::= ( S, E)   L ::= E         L ::= L , E
   ```

3a. Build the LR(0) DFA for this grammar:
   ```
   S ::= E $
   E ::= id
   E ::= ( E )
   E ::= E + id
   ```
  b. Show the parse for: ((id+id+(id)))

4. Consider the following MiniJava statement:
   ```
   while (x < 3+4*5) {
      System.out.println(x);
      y = new C().test(a,b,c);
   }
   ```
  a. Draw the **concrete** syntax tree for this statement. Use the MiniJava grammar embedded in the Parser/minijava.cup file. (You may abbreviate non-terminal names if unambiguous, e.g. Id for Identifier.)
  b. Draw the **abstract** syntax tree for this statement, labelling each AST node with the name of an AST node class from the MiniJava compiler and labelling each child edge with the name of an instance variable of the parent node's class. (Child edges that lead to non-AST nodes, such as integers or strings, can be drawn to lead to a particular integer or string value.)

Produce a hard-copy of your answers and turn them in by the start of class on the due date. Do these exercises individually, not with your project partner.