# Advanced Compilers

COMP 240 Fall 2002

## Written Assignment #2

Assigned: Tue Oct 8, 2002
Due: Thu Oct 17, 2002

Note: In the grammars below, upper case letters denote non-terminal, lower-case letters and all symbols other than "→", and "|" are terminals, and ε denotes the empty string.

**I.** [6] For the following grammar *G*.

S → a S e | B
B → b B e | C
C → c C e | d

(a) Determine *Nullable*(*X*), *Starters(X*), *Followers*(*X*), for each nonterminal *X* in *G*.
(b) Construct the *Predict* function to be used by a top-down parser for *G*.
(c) Does *G* does meet the LL(1) condition?

**II.** [6] Construct the LR(0) CFSM for the following grammar. Next, use the lookahead propagation technique to construct the LALR(1) lookahead sets. Does the grammar meet the LALR(1) condition?

S → T$
T → aDb | BD
B → Bb | ε
D → d | ε

**III.** [10] For each of the two following grammars determine whether they meet the SLR(1), LALR(1) and LR(1) conditions. Feel free to type the two grammars into Yacc or Bison.

(a)
S → id := E$
E → E + P | P
P → id | id := E | ( E )

(b)
S → id := A$
A → id := A | E
E → E + P | P
P → id | ( id , id ) | ( A )

**IV.** [6] Use grammar transformations to place the following EBNF grammar into a form meeting the LL(1) condition, and show that this condition holds. The grammar has two nonterminals, Stmt and Lval, and six terminals (shown in bold).

Stmt → Lval **:= id**
Lval → **id** | Lval **. id** | Lval **[ num ]**

**V**. [12] A variable or method declaration in a Java class may include *modifiers* to specify the visibility of the variable, to specify whether the variable is class-based or instance-based, and to specify whether the variable's value is constant. Modifiers are keywords that may be written in any order before a declaration. For example,

```
public static final <var declaration>
```

and

```
final public static <var declaration>
```

have the same meaning. Each modifier is drawn from a category, and only one modifier can appear out of each category. If a modifier category includes ε, it means a modifier from this category does not need to be present. In full Java there are something like eight of these categories, but for our purposes the categories of modifiers are limited to

visibility = {**public**, **private**, ε},

class = {**static**, ε},

modifiability = {**final**, ε}.

(a) Construct a grammar that describes the legal modifier sequences.

(b) Construct an LL(1) or LALR(1) grammar that describes the legal modifier sequences.

(c) Investigate an alternate way of recognizing legal modifier sequences that does not place the entire burden on the parser.