CS314 Spring 2014

Assignment 3

Due Friday, February 21, **before** class

1 Problem — LL(1) Grammars and Recursive Descent Parsing

<program> ::= **program** <block> **.**

<block> ::= **begin** <stmtlist> **end**

<stmtlist> ::= <stmt> <morestmts>

<morestmts> ::= **;** <stmtlist> |ϵ

<stmt> ::= <assign> |<ifstmt> |
<whilestmt> |<block>

<assign> ::= <variable> = <expr>

<ifstmt> ::= **if** <testexpr> **then** <stmt> **else** <stmt>

<whilestmt> ::= **while** <testexpr> **do** <stmt>

<testexpr> ::= <variable> <= <expr>

<expr> ::= + <expr> <expr> |
$*$ <expr> <expr> |
<variable> |
<digit>

<variable> :: = a | b | c

<digit> :: = 0 | 1 | 2

1. Show that the grammar above is LL(1). Use a formal argument based on the definition of the LL(1) property.
2. Show the LL(1) parse table.
3. Write a recursive descent parser for the above grammar in an imperative pseudo code as used in class (see lecture 7).

4. Modify your recursive descent parser such that it prints the number of assignment statements in the input program after it successfully parsed the program, and the number of variable references. For the program listed below, your parser should print "4 assignments, 15 variable references". Note that a statement such as "a = + a a" has three references, not just one.

```
program
begin
  if b <= 0 then
    while a <= 1 do
    begin
      a = + a b;
    end
  else
    begin
      a = * a b
      c = + a - + b 1 c
    end;
  c = + a b
end.
```