CSC 4101 – Programming Languages
Project 1

Introduction:

- This is an optional project. It can be used to replace the Final exam.
- To get the project grade, you must attend and successfully pass a rigorous code review held at the instructor's office.
- Failure to submit the project by the deadline will lead to no grade.

Project Description:

Write a laxer and a Recursive Descent Parser for the language described by the following specifications:

1- Programs in the language start with the word program and end with end_program
<program> → program <statements> end_program

2- The language is typeless
3- Statements end with a semicolon
4- The language supports alphanumeric identifier names that do not start with a digit
5- The language supports the operations =, +, -, *, /, %, and ()
6- The language supports if statements (no else) in the form:
<condition> → if (<logic expression>) <statements> end_if
7- The language supports binary logical operations in the form
<logic_expression> → <var> (==|!=|>|<|>=|=<) <var>

8- The language supports loops that increment a value by 1 until an end value is reached, such that
<loop> → loop(<var> = <var> : <var>)<statements> end_loop

Instructions

- Design the EBNF grammar for the language.
- Write a lexer for the language.
- Write a Recursive Descent Parser for the grammar.
- Add a GUI that allows the user to type in source code for the language and parse programs written in the language.
- The GUI should produce valid error messages.
- You are free to use any programming language you want.
- Feel free to make any design decision necessary, such as defining the token categories.
- You are allowed to use regular expressions for the lexer, but not the parser

Example programs written in this language

```
program
value = 32;
mod1 = 45;

z = mod1 / value * (value % 7) + mod1;

loop (i = 0 : value)
z = z + mod1;
end_loop

if (z >= 50)
newValue = 50 / mod1;
x = mod1;
end_if

end_program
```