Assignment #1

Date Due: January 31, 2018

Total: 100 marks

Instructions

- Submit all program code and any relevant output from program testing.
- Combine all your files into a single a compressed file. Please use a filename which includes your UPEI username and is of the form: username_ast1.zip; submit the zip file via moodle.
- Some explanations may be submitted on paper (the expanded grammar, scanner construction, etc).

Requirements

The task is to develop a translator (Java and C programs) for converting a program from source code to an intermediate format. We know that:

- The programming languages includes arithmetic expressions in infix format.
- The intermediate format is postfix.
- The program should handle a series of statements, each of which is converted to a postfix form.
- The program should be capable of handling identifiers and numbers.
- The program should include reasonable error checking.

1. (15 marks) Use the lexical analyzer as developed in Section 2.6.5 (refer to figures 2.33–2.35 and Example E2) as a starting point. Expand this scanner as described in Exercises 2.6.1, 2.6.2 and 2.6.3 on pp. 84–85.
2. (25 marks) Expand the grammar for the infix-to-postfix (Fig. 2.28 in your textbook and Example E1) translation scheme to incorporate additional productions induced by 1.

$$
\begin{array}{lcl}
prog & \rightarrow & \textbf{begin}\ list\ \textbf{end} \\
list & \rightarrow & list\ stmt \\
 & | & \varepsilon \\
 & & \\
stmt & \rightarrow & assign; \\
 & & \\
assign & \rightarrow & id = expr \\
 & | & expr \\
 & & \\
expr & \rightarrow & expr + term \\
 & | & expr - term \\
 & | & term \\
 & & \\
term & \rightarrow & term * factor \\
 & | & term / factor \\
 & | & factor \\
 & & \\
factor & \rightarrow & (expr) \\
 & | & \texttt{id} \\
 & | & \texttt{integer} \\
 & | & \texttt{real}
\end{array}
$$

Modify the syntax-directed translation scheme after eliminating left-recursion associated with the grammar so that expressions are translated into reverse Polish(postfix) notation.

Refer to Section 2.5.2 and Figure 2.23.

Submit the modified grammar productions with the translation scheme.

3. (50 marks) Develop the complete (Java and C) program for the translator. The translation scheme follows the guidelines for the translator as developed in Section 2.5 (25 marks for each programming language)

4. (20 marks) Design a number of test cases to provide reasonable confidence that your translation scheme is correct. Run your test cases and produce a file for the output of each test (record the execution in a file). Provide an overall summary of your test runs and the confidence that you have that it is working correctly.