COP4020 Fall 2016 Homework Assignment 3

1. Errors in a computer program can be classified according to when they are detected and, if they are detected at compile time, what part of the compiler detects them. Using C, give an example of each of the following.
   a) A lexical error (detected by the scanner)
   b) A syntax error (detected by the parser)
   c) A static semantic error (detected by semantic analysis)
   d) A dynamic semantic error (detected by the code generated by the compiler).

2. Standard ISO Pascal only allows pointers to reference dynamically created variables and does not allow them to reference standard static or local variables. Pointer arithmetic is not permitted. This eliminates some security risks associated with pointers caused by dangling pointers, but risks cannot be eliminated when Pascal’s dispose() statement is used (dispose() is similar to free() in C). Older Pascal standards permit pointer de-references to valid data only.
   a) Explain how Java prevents dangling references and memory leaks.
   b) How would you implement dispose() in Pascal to prevent dangling pointers and prevent memory leaks? Give two approaches, one based on a) and another alternative that uses dispose(). Find and quote online resources if necessary.

3. Convert the regular expression
   
   (+ 1 - ) digit digit*
   
   into an NFA using the NFA equivalents (base case, concatenation, alternation, Kleene closure).

4. Given the “unambiguous LL(1) grammar for simple expressions E” shown on the lecture notes on Syntax, construct the parse tree of 2 * x / (1 - y)

5. Consider the following LL(1) grammar for a simplified subset of Lisp

   \[ P \rightarrow E \] $$
   \]
   \[ E \rightarrow \text{atom} \]
   \[ \rightarrow \text{'}\ E \]
   \[ \rightarrow ( E \ Es ) \]
$Es \rightarrow E \ Es$

$\rightarrow \varepsilon$

a. Determine $\text{FIRST}(E)$
b. Construct a recursive descent parser for this grammar, using the tokens in the
   $\text{FIRST}()$ sets to determine which production applies. You can write this in C or in
   pseudo code using the class cd ~lecture notes as a guide.
c. Using this grammar, give a parse tree of $(\text{cdr } (a\ b\ c))$ $\text{cdr}$

6. After studying the “Semantics” lecture notes and “attribute grammars”, answer the
   following questions.

a. Augment the following grammar of the language of nested blocks with semantic rules
   to count the number of matching braces:

   \[
   \begin{align*}
   \langle \text{block} \rangle & \rightarrow \{ \langle \text{block} \rangle \} \\
   \langle \text{block} \rangle & \rightarrow ;
   \end{align*}
   \]

   The synthesized attribute $\text{block.num}$ should hold the nesting depth of the block.
b. Draw a decorated parse tree for the input \{ \{ ; \} \}. Show the $\text{block.num}$ attribute
   values in the tree nodes.