SFU CMPT 379 - Compilers
Summer 2021
Midterm

50 minutes. 100 points. According to my calculations, that's two points per minute.
I will be available on zoom (same link as office hours) during the exam to answer questions.
Open book, open notes. Internet may be used only to view lecture slides, format answers, or ask me questions. You are given 15 minutes after the exam period to format and upload your answers to Canvas; this time is not to be used for answering questions. By submitting answers, you are certifying that you have adhered to all exam procedures and academic honesty guidelines.

There are 6 questions. There are many subparts to questions, so this exam is probably long. Manage your time carefully.

1. (25 points; 5 each) Answer these questions in the context of this course.
   a. What is a binding? An association between a variable name, a type, and a way of accessing that variable name in memory (e.g. memory location).
   b. What is a language? A set of strings (over some alphabet).
   c. What is a lexeme? Source program text that causes the creation of a token.
   d. What is the difference between a parse tree and an abstract syntax tree? A parse tree contains all terminals and nonterminals used in the derivation, and an abstract syntax tree is an abbreviation of a parse tree that contains the essential items.
   e. What is a grammar? What is a grammar used for? A grammar is a 4-tuple (N, T, S, P) where N is a set of nonterminals, T is a set of terminals, S is the start nonterminal, and P is a set of productions of the form $a \rightarrow \alpha$ where a is a nonterminal and α is a string of terminals and nonterminals. A grammar is used to define a language.

2. (28 points; 4 each) State which phase of a typical C++ or java compiler normally performs the following tasks. If something is done at run-time, then it's the code generator's responsibility.

   a. Ensuring that an identifier (variable) is declared before it is used. Semantic analyzer
   b. Ensuring that each new object created has its own place in memory. Code generator
   c. Matching *else* clauses to the appropriate *if* statement in nested if-then-else constructs. Parser
   d. Determining if a string of characters in the input is a keyword. Lexical analyzer
   e. Ensuring that a freshly-declared variable hides other instances of variables with the same name. Semantic analyzer
   f. Ensuring that every operand is evaluated before an operator executes. Code generator
   g. Determining if an operator associates to the left or to the right. Parser

3. (7 points) Is division by zero statically or dynamically checked? Why?
   Dynamically. If the divisor is a variable, one in general cannot know the value of the variable until run time.

4. (20 points; 5 each)
   a. When is a grammar called *ambiguous*? When there is some sentence (derived string) that can be derived in two different ways (i.e. with two different parse trees).
   b. When is a grammar suitable for LL(1) parsing? When it is left-factored and has no left recursion.

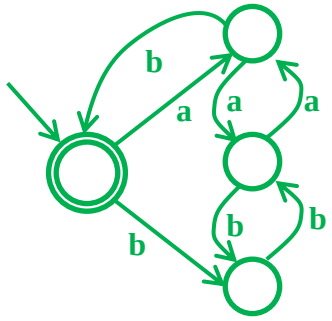
c. Which of LL(1) and LR(0) parsing is top-down, and which is bottom-up? LL(1) is top-down, and LR(0) is bottom-up.

d. What are the LR(0) items for the following grammar fragment?

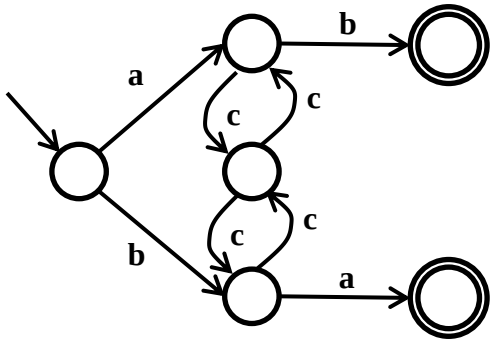
L → S L | ε
S → i C t S
S → b L e

L → · S L L → S · L L → S L · L → ·
S → · i C t S S → i · C t S S → i C · t S S → i C t · S S → i C t S ·
S → · b L e S → b · L e S → b L · e S → b L e ·

5. (10 points) Draw a DFA (not NFA) transition diagram for the regular expression
((aa|bb)*ab)*



6. (10 points) Write a single compact regular expression for the language accepted by the following NFA.



ab | ba | (a | b)(cc)$^{+}$(a|b)