Department of Computer Science **University of San Francisco**

Computer Science 414
Spring 2015

Mitderm 1 Practice Problems

1. Describe the difference between the languages described by the following 3 regular expressions:
   (a) a*b*
   (b) ab*
   (c) (ab)*

2. Give a regular expression that defines C floating point literals.

   THere are two kinds of floating-point literals in C: The kind first one consists of the following parts:

   - A nonempty sequence of decimal digits containing a decimal point character (defines significand)
   - (optional) e or E followed with optional minus or plus sign and nonempty sequence of decimal digits (defines exponent)
   - (optional) a suffix type specifier as a l, f, L or F

   The second kind consists of the following parts:

   - nonempty sequence of decimal digits (defines significant)
   - (NOT OPTIONAL!) e or E followed with optional minus or plus sign and nonempty sequence of decimal digits (defines exponent)
   - (optional) a suffix type specifier as a l, f, L or F

3. Give a Context-Free Grammar for each of the following langauges:
   (a) The set of all strings over {(,), [,]} which form balanced parenthesis. That is, (), ()(), ((()())()), [()()], and ([()[]()]) are all in the language, but)(, (() and (] are not in the language.
   (b) The set of all strings over {num, +, -, *, /} which are legal binary post-fix expressions. Thus num, num num +, num num num * -, and num num - num * are all in the language, while num *, num * num, and num num num - are not in the language.
   (c) For the grammar in part (b) above, remove left recursion and left factor as necessary to make the grammar LL(1). Then find the first and follow sets for all of the non-terminals, and give the parse table.

4. For the following CFGs, compute First and Follow sets for each non-terminal, and then create an LL(1) parse table
   (a)

Terminals	=	{for, to, id, :=, num, print}
Non-Terminals	=	$\{S, E\}$
Rules	=	(1) $S \rightarrow$ for id := E to E S
		(2) $S \rightarrow$ print E
		(3) $E \rightarrow$ id
		(4) $E \rightarrow$ num
Start Symbol	=	S

(b)

Terminals	=	{a, b, c, \$}
Non-Terminals	=	$\{S, A, B\}$
Rules	=	(1) $S \to ABc\$$
		(2) $A \to \mathrm{a}A$
		(3) $A \to \epsilon$
		(4) $B \to \mathrm{b}B$
		(5) $B \to \epsilon$
Start Symbol	=	S

5. Give a CFG for the following language. This grammar need not be LL(1)!

   A subset of C-style expressions, over the operators +, * (times, not pointer dereference), ==, =, and the identifier token id (so your grammar should have 5 terminals – 4 operators and id). Be sure to use the correct precedence and associativity for C!