Principles of Programming Languages [PLP-2014]

Exercises on Syntax-Directed Definitions - March 13, 2015

1. Given the following grammar for expressions:

E→E+T	T→T/F
E→E-T	T→F
E→T	F→(E)
T→T*F	F → id

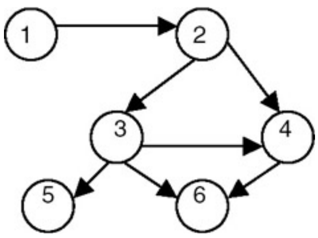
write the generated string `a*(b-c)+(b-c)/a` as a parse tree, as an abstract syntax tree, and as a DAG that is minimal.

2. Given the following attributed grammar

	Production	Semantic Rules
1)	$L \to E\ \mathbf{n}$	$L.val = E.val$
2)	$E \to E_1\ +\ T$	$E.val = E_1.val + T.val$
3)	$E \to T$	$E.val = T.val$
4)	$T \to T_1\ *\ F$	$T.val = T_1.val \times F.val$
5)	$T \to F$	$T.val = F.val$
6)	$F \to (\ E\ )$	$F.val = E.val$
7)	$F \to \mathbf{digit}$	$F.val = \mathbf{digit}.\text{lexval}$

show the annotated parse tree for expression **`(5+8*7)*4n`**.

3. Write down all the topological sorts of the following partial ordered graph:



4. This grammar generates binary numbers with a "decimal" point:

   S → L.L | L   L → L B | B   B → 0 | 1

   Design an L-attributed SDD to compute B.val, the decimal-number value of an input string. For example, the translation of string 101.101 should be the decimal number 5.625. Hint: use an inherited attribute L.side that tells which side of the decimal point a bit is on.

5. Generate the three-address code sequences for the following instruction:
   a) a[i] = b*c – b*d
   b) a = b[i] + c[j]

6. A real array A[i, j, k] has index i ranging from 1 to 4, index j ranging from 0 to 4, and index k ranging from 5 to 10. Reals take 8 bytes each. Suppose array A is stored starting at byte 0. Find the location of:
   a) A[3, 4, 5]   b) A[1, 2, 7]   c) A[4, 3, 9].

7. Consider the following Post system rules:

$$\frac{\rho \vdash e_1 : bool \quad \rho \vdash e_2 : bool}{\rho \vdash e_1 \textbf{ and } e_2 : bool} \qquad \frac{\rho \vdash e_1 : bool \quad \rho \vdash e_2 : bool}{\rho \vdash e_1 \textbf{ or } e_2 : bool} \qquad \frac{\rho(v) = t}{\rho \vdash v : t}$$

Given $\rho = \{\langle \text{a}, bool\rangle, \langle \text{b}, bool\rangle, \langle \text{c}, bool\rangle\}$, show the proof of

$$\rho \vdash \text{a } \textbf{or} \text{ b } \textbf{and} \text{ c} : \text{bool}$$

8. Define an L-attributed SDD on a top-down parsable grammar to generate the NFA associated with a regular expression, using Thompson's algorithm sketched in the next figure. Assume that there is a token **char** representing any character, and that **char**.*lexval* is the character it represents. You may also assume the existence of a function **new()** that returns a new state, that is, a state never before returned by this function. Use any convenient notation to specify the transitions of the NFA.