CSE 6341, Written Assignment 3
Due Tuesday, February 20, 11:59 pm (8 points)

Your submissions should be uploaded via Carmen. Create your answers using a text editor and upload the file (e.g., plain text, Word, PDF). Alternatively, you can write your answers by hand and take a photo (or scan), but please ensure that (1) your handwriting is *clear and legible*, and (2) your photo or scan has *high resolution*, to allow the grader to read and understand your submission.

Q1 (3 points): Consider the following context-free grammar for a C++ style language:
<program> ::= <classList>
<classList> ::= <classDecl> | <classDecl> **;** <classList>
<classDecl> ::= **class ident {** <classBody> **}** | **class ident**$_1$ **: ident**$_2$ **{** <classBody> **}**
<classBody> ::= … *[productions not relevant for this question]*

The program contains a sequence of class declarations. Each declaration contains the keyword **class**, followed by the name of the class. A class could optionally have a superclass: for example, declaration **class Y : X { … }** shows that class X is a superclass of class Y.

Define an attribute grammar to check the following condition: whenever a class declaration shows that some class Z is a superclass, that class Z must be declared earlier in the program. For example, the following program is valid
class X { … } ; class Y : X { …}
but this one is not
class Y : X { … } ; class X { …}

Your attribute grammar should accept all programs that satisfy this condition and should reject all programs that violate this condition. For each attribute you define, describe its name, type, and whether it is inherited or synthesized. Show the complete evaluation rules for all attributes. *Try to keep your solution* ***as simple as possible****. Do* ***not*** *just blindly copy the solutions from the lecture notes. Do* ***not*** *use global data structures or side effects*.

Q2 (3 points): Consider the attribute grammar for assembly code generation discussed in class. Suppose we extend the language with a do-while loop, with the following syntax:

<stmt> ::= **do** <stmt>$_2$ **while (** <cond> **)**

Show the complete attribute grammar evaluation rules for generating assembly code for do-while loops. Use notation similar to the one used in the lecture notes.

Illustrate your solution by showing the complete generated code for the following program:

j=81; do { j=j / 3; } while (j>1)

Use the code generation rules from the lecture notes together with your new rule. Since we have not defined code generation rules for conditional expressions such as **j>1**, just use **"..."** in your solution to denote the assembly code for computing the value of **j>1** and assume that this assembly code works as described in the lecture notes.

Q3 (2 points): Consider the operational semantics defined in class. Using the inference rules from the lecture notes, construct the **derivation tree** for the following triple:
$\langle x\text{-}2*y*z, \sigma\rangle \rightarrow 29$
where $\sigma = [x\mapsto 5, y\mapsto -3, z\mapsto 4]$. Assume that multiplication is left-associative. Your answer should show the entire derivation tree. Every level of the tree should correspond to one of the inference rules from the lecture notes.