CSC 530 Assignment 3 --
Formal Semantics Problem Set

ISSUED: Wednesday, 1 May 2002
DUE: Monday, 20 May 2002

1. In the correction to his 1968 paper, Knuth made an update to semantic rule 6.2 of the Turingol definition. Using an example, briefly explain why this correction was necessary.

2. In the Turingol definition, several of the attributes are global, in particular Σ, Q, δ, **label**, and **symbol**. Describe how the Turingol definition can be modified so that these five attributes are no longer global. That is, Σ, Q, δ, **label**, and **symbol** will be explicitly inherited and synthesized throughout the definition. State your answer in prose, describing the attribute flow through the rules, noting when any attribute splitting is required. Refer to rules by the numbers given in Table 1 on page 141 of the Knuth paper. Support your answer with semantic rule excerpts as necessary for clarity.

3. Answer exercises 13.4, 13,7, 13.8, and 13.9 on pages 242-243 of the Tennent book excerpt.

4. Define the syntax and formal semantics of two of the following advanced language features:
   a. function and built-in operator overloading, including resolution of overloaded function invocation
   b. inheritance, including subtype polymorphic function invocation
   c. generic modules, including information hiding

Define semantics using either attribute grammar *OR* denotational semantics (not both). If you use attribute grammar, define your syntax and semantics in the context of the SIL definition given in Lecture Notes 6. If you use denotational semantics, define syntax and semantics as extensions to the Pascal-like language presented in Tennent Chapter 13.

Accompany your definition with some prose discussion and simple examples that illuminate your definitional approach. Keep the discussion as brief and to the point as possible. In particular you need not present any rationale for why you chose one approach over another.

The syntax and semantics you define need not match any existing language, but they must be "sensible" and completely and consistently defined. Note that in Assignment 4, you will add one of these extensions to the Lisp interpreter of Assignments 2. You may therefore keep implementability in mind. However, "implementability" cannot dominate "sensibility" in that you cannot define a feature that is so wimped out as to be trivially implementable.