Homework 4: Term Project and Imperative Calculi

Due: Problems 1–3, April 20, 2000; problem 4, April 27, 2000.
In this homework, you will finish your term project and explore imperative calculi.
Read Sections 9.1–2, 9.5–6, and Chapters 10–11 of *A Theory of Objects*.

1. Do you think recursive types are important in practice (i.e., in practical object-oriented language designs)? Why or why not?

2. In your own words, briefly describe the problems discussed in section 9.5 and 9.6 of the text.

3. Does the Imp-ς calculus have minimum types? If so, prove that it does; if not, give a counterexample.

4. To finish your term project, you will add imperative features to your object-oriented calculus from the previous homework. Revise your dynamic semantics to take these features into account. Also revise your type system, if necessary.

   You should hand in the following.

   - A short English introduction to the design of your language and its dynamic semantics.
   - A version of your dynamic semantics coded in either lambda Prolog or Haskell. (Again, you do not need to do a parser, start from the abstract syntax trees.) I suggest using a big-step semantics; you do not need to have both big-step and little-step semantics.
   - Several examples of small programs in your calculus, and tests of their evaluation using your semantics.
   - A short English introduction to the design of your language's type system.
   - A type system for your language, which takes the imperative features into account. You can do this completely on paper if you wish. Your type system need only be first-order. It should include a notion of subtyping.
   - Some examples of terms in your language and derivations of their types.