

# Functional Languages

## 10.10 Explorations

- 10.28 Learn about the *typed lambda calculus*. What properties does it have that standard lambda calculus does not? What restrictions does it place on permissible expressions? Possible places to start include Cardelli and Wegner's classic survey [CW85] or the newer text by Pierce [Pie02].
- 10.29 Learn more about *fixed points*. We mentioned these when presenting the **Y** combinator in Section ©10.6.2. They also arise in the denotational definition of loop constructs, in metacircular interpreters (Example 10.20), and in the *data flow analysis* used by optimizing compilers (Section ©16.4.2). What do these subjects have in common? Are there important differences as well?
- 10.30 Explore the connection between lexical scoping in Scheme and the notion of free and bound variables in lambda calculus. How closely are these related? Why does lambda calculus require alpha conversion but Scheme does not? Is there any analogy in lambda calculus to the dynamic scoping of early dialects of Lisp?

