Scheme

- Like ML, functional focus with imperative features
  - anonymous functions, closures, no return statement, etc.
  - but every binding is mutable
- A really minimalist syntax/semantics
  - In the LISP tradition
    - LISP was earliest language that targeted symbolic computation
  - Current standard is 50 pages
- Dynamically typed while still being type-safe
  - Less compile-time checking (run-time checking instead)
  - Accepts more perfectly reasonable programs
- Some advanced features for decades
  - Programs as data, hygienic macros, continuations

Scheme Syntax/Semantics

- Scheme term is either:
  - an atom (identifier, number, symbol, string, ...)
  - a sequence of terms (t1 ... tn)
- Semantically, identifiers are resolved in an environment and other atoms are values.
- The semantics of a sequence depends on t1 :
  - certain character sequences are “special forms”
  - otherwise a sequence is a function application
- Some special forms:
  - define
  - lambda
  - if, cond, and, or
  - let, let*, letrec