CSE 341: Programming Languages

Autumn 2005

Lecture 4 — Mutation; "one-of" types; user-defined types

Where are we

- Done features: functions, tuples, lists, options, local bindings
- Done concepts: syntax vs. semantics, environments
- Today features: record types, datatypes, type synonyms, pattern-matching
- Today concepts: Mutation-free, "one-of" types, constructors/destructors, case-coverage

You want to *change* something?

There is no way to *mutate* (assign to) a binding, pair component, or list element.

How could the *lack* of a feature make programming easier?
In this case:

- Amount of sharing is indistinguishable
  - Aliasing irrelevant to correctness!
- Bindings are invariant across function application
  - Mutation breaks compositional reasoning, a (the?) intellectual tool of engineering