CSE341: Programming Languages

Lecture 18
Static vs. Dynamic Typing

Dan Grossman

Fall 2011

Static vs. dynamic typing

- A big, juicy, essential, topic about how to think about PLs
  - Conversation usually overrun with half-informed opinions ☹
  - Will consider reasonable arguments "for" and "against" last

- First need to understand:
  - What static checking (e.g., with a type system) *means*
  - What static checking *intends to do* (details depend on PL)
  - Why static checking *must be approximate*
  - The *standard features* of a type system
  - A more general view of *how eager* should error detection be

Static checking

- *Static checking* is anything done to reject a program ***after*** it (successfully) parses but ***before*** it runs

- **What static checking is performed is part of the PL definition**
  - A "helpful tool" could do more

- Most common way to define a PL's static checking is via a *type system*
  - *Approach* is to give each variable, expression, etc. a type
  - *Purposes* include preventing misuse of primitives (e.g., `4/"hi"`) and avoiding dynamic checking (dynamic means at run-time)

- Dynamically typed PLs (Racket, Ruby) do much less static checking
  - Maybe none but the line is fuzzy