Programming Languages

Dan Grossman
University of Washington

Part A Wrap-Up

&

Parts B&C Preview

Course Overview

• Wow, we have done a lot!

Part A:

1. Basics, functions, recursion, scope, variables, tuples, lists, …
2. Datatypes, pattern-matching, tail recursion
3. First-class functions, closures [and course motivation!]
4. Type inference, modules, equivalence

Overall: A *precisely specified* introduction to statically typed functional programming built up piece-by-piece

– Tremendous synergy among these topics
– We programmed in the [very] small, but the ideas scale!
– Will highly leverage this foundation in Parts B & C

Part B

5. Quick "re-do" in a dynamically typed language;
     Delaying evaluation

6. Implementing languages with interpreters;
     Static vs. dynamic typing

"Just" two homeworks, but the second one is particularly challenging and rewarding
  – Part C is 2 homeworks and 1 exam

Part C

7. Dynamically-typed Object-Oriented Programming

8. OOP vs. Functional decomposition
   Advanced OOP topics (e.g., mixins, double dispatch)
   Generics vs. Subtyping

"Finishes the story" even if you "already know OOP"
  – Some with OOP background find 7 "less interesting" but stay tuned for 8
  – Some find Part C "anti-OOP", which is mostly ☺ not true
    • About contrasting with what many "already think/know"
    • And yes, some focus on where commitment to "pure OOP" *may* be unwise even in "an OOP language"