CS 441
Programming Languages

Andrew Appel

Princeton University, Fall 1999

Many of these slides are adapted from slides by Robert Harper, Carnegie Mellon University, 1999.

Welcome!

- CS 441: Programming Languages
  - intro to the **fundamental results and techniques** in programming languages
  - emphasis on **scientific principles**, de-emphasis on taxonomizing the world
  - based on **current research** in PL's

Overview

- Type systems and type safety
  - What is a "safe" language?
  - What are their properties?
- Abstract machines
  - What is a virtual machine?
  - Why are virtual machines fashionable?
- Type inference, constraints.
  - How can types help, not get in your way?

Overview

- Polymorphism and data abstraction.
  - What is an "abstract data type"?
  - Why are ADT's important?
- Subtyping.
  - When is one type a subtype of another?
  - How does subtyping support reuse?

Overview

- Functional programming
  - Uses of higher-order functions
  - Programming with values, not objects
  - Abstraction in programs
- Objects, Classes, Inheritance.
  - How to build flexible, extensible software?
  - How are subtyping and inheritance related?
- Concurrent Programming
  - How can threads communicate?
  - Taming complexity of concurrent programming

Unifying Framework

- Static semantics:
  what is a well-formed program?
  - **Type systems.**
- Dynamic semantics:
  how is a program executed?
  - **Transition systems.**

Unifying Framework

- Type systems and transition systems provide a "Grand Unified Theory" of PL's.
  - unifying conceptual framework for discussing language "features"
  - mathematical basis for making precise statements about program behavior
  - crucial for modern implementation techniques

What is this course like?

- Stresses **mathematical skills**, **implementation skills**, and **common sense.**
  - Prove theorems.
  - Build systems.
  - Evaluate claims.
- Emphasis on **rigor**, **clarity**, and **elegance**.
  - "If it isn't beautiful, it isn't right."

What is this course like?

- You will have to **work hard** to keep up.
  - Ideas are abstract and largely unfamiliar. They take time to "sink in".
  - There is no one text covering the material in this course. Studying and experimentation is **essential** to keep up.
  - Assignments will involve both **programming problems** and **written work**.

What is this course like?

- Most of all, it's **fun**!
  - Elegant interplay of theory and practice.
  - Lots of interesting ideas to discuss.
  - Enormous potential for further development of ideas.
  - Rapidly gaining importance in the "real world".

What this course is not!

- This is **not** a compilers course!
  - Take COS 320.
- Not about **syntax** of languages.
  - Theory and practice of syntax is in 320.
- This is **not** a way to master the "Top 10" PL's.
  - We care about **enduring principles**, not **passing fads**.

People

- Andrew Appel, professor.
  - 409 Computer Science Building, 8-4627, appel@cs.
  - Office hours: by appointment.
  - Send e-mail if you want to be sure of finding me.
- Eun-Young Lee, teaching assistant.
  - 414 Computer Science Building, 8-5388, elee@cs.
  - Office hours: Mon. 3 - 4:30, Thurs. 1 - 2:30.

Course Information

- Web site
  `http://www.cs.princeton.edu/courses/cs441`
  - notes, slides, handouts, code, solutions
  - check the web frequently for updates

Textbooks

- Main Texts:
  Harper, *Programming Languages: Theory & Practice* (manuscript) 1999.
  Reppy, *Concurrent Programming in ML*, Cambridge, 1999.
- Supplemental Texts:
  Ullman, *Elements of ML Programming*, Prentice Hall, 1998.
  Arnold & Gosling, *The Java Programming Language*, Addison-Wesley, 1996.

Programming Language

- Programming assignments will be in **Standard ML** and **Java**.
  You can run these on Unix or Windows
  Both already installed on CS-dept Unix machines
  You can get ML and Java for free to install on your own Unix or Windows machine
  See Web page for more information.

Homework

- One problem set every week.
  - Some pencil-and-paper exercises,
  - Some small (but deep) programs.
  - **Individual** work only, unless specified otherwise.
- **Clarity** and **elegance** count.
  - It's not sufficient to "work" on test data!

Exams, Grades

- Midterm
  - take-home, open book, due Oct 28 (20%)
- Final
  - take-home, open book, due Jan ?? (30%)
- Homework (50%)

Today's lecture:

ML Programming

(on the blackboard, no slides)

Running ML on cs-dept machines:

`/usr/local/sml/bin/sml`

(see Web page for more info)