CS 565: Programming Languages

Fall 2011
M,W,F: 11:30 - 12:20
Lawson B134
Administrivia

- **Course Web Page:**

- **Office Hours:**
  - Tu, Th: 2 -- 3 PM
  - By appointment

- **Main text:**
  - *Software Foundations: hypertext version available from the course web page*
Course Work

- Lectures
- Homeworks
  - Exercises each week
  - Collaborating on homework encouraged
  - All homework exercises will involve using the Coq proof assistant
- Semester project
- Cumulative final which will also serve as the qualifying exam
Prerequisites

- **Programming experience/maturity**
  - exposure to various language constructs
    - Java, ML, Lisp, Prolog, C
  - Undergraduate compilers and/or PL class
    - CS 352 and/or CS 456 or equivalent

- **Mathematical maturity**
  - familiarity with first-order logic, set theory, graph theory, induction

- **Most important:**
  - Intellectual curiosity and creativity
Resources

- Web page for text:

- Supplementary material:
  - [http://www.cis.upenn.edu/~bcpierce/tapl](http://www.cis.upenn.edu/~bcpierce/tapl)

- Coq resources:
  - [http://coq.inria.fr](http://coq.inria.fr)

- Proceedings of conferences
  - POPL, PLDI, ICFP, ...
Background

- Our main goal is to find ways to describe the behavior of programs precisely and concisely

Motivation

- Significant industry and government interest
  - Web, Java
  - Security issues
  - Complexity of modern-day applications
Motivation (cont)

- Prove specific facts about programs
  - Verify correctness
    - Important in mission-critical systems
  - Safety properties
  - Need an unambiguous vocabulary

- Understand specific language features
  - Better language design
  - Guide improvements in implementations
Goals

- A more sophisticated appreciation of programs, their structure, and the field as a whole
  - Viewing programs as rich, formal, mathematical objects, not mere syntax
  - Define and prove rigorous claims about a program’s meaning and behavior
  - Develop sound intuitions to better judge language properties

- Develop tools to be better programmers, software designers, computer scientists, and thinkers
Non-goals

- An introduction to advanced programming techniques

- No detailed discussion of machine implementations
  - The course will not be motivated from the perspective of a compiler writer
  - But, impact of design decisions on implementation tractability will be considered when appropriate

- A survey of different languages
Topics

- Part I (Tools): Proof assistants and functional programming
- Part II (Foundations): Program logic and reasoning principles
- Part III (Semantics and specifications): operational, axiomatic
- Part IV (Type Systems): $\lambda$-calculus, records, references, polymorphism, subtyping, inheritance, ...
Project

☐ Identify a problem
  ■ language, types, security, databases, security, data structures

☐ Formalize a system that captures salient properties of this domain

☐ Prove interesting properties about this system

☐ Formalization and proofs done in Coq
Homework

☐ Install Coq
  ■ version 8.2pl

☐ Install either
  ■ Proof General
    ☐ http://proofgeneral.inf.ed.ac.uk
  ■ Coq ide
    ☐ available as part of the Coq distribution

☐ Answer questions in hw1.v