CS611

Advanced Programming Languages
Radu Rugina

Lecture 01: Introduction
30 Aug 02

Outline

- Course organization
  - General course information
  - Homework & project information
- Course overview
  - What 611 is about
  - What 611 is not about
  - Contents overview
- A simple example

General Information

When Where	MWF 10:10 - 11:00AM Olin Hall 218
Instructor Teaching Assistant Admin Assistant	Radu Rugina (Upson 4141) Stephen Chong (Upson 5157) Juanita Heyerman
Course staff email	cs611@cs.cornell.edu
Newsgroup	cornell.class.cs611
Web page	courses.cs.cornell.edu/cs611

Workload

- Six homeworks (about half with programming requirements, in ML)
- Scribe 2-3 lectures (in pairs)
- One prelim: tentatively 10/31, 7:00-9:30 PM
- Final exam: 12/20, 9:00-11:30 AM

Grading

- Homeworks: 40%
- Preliminary exam: 20%
- Final exam: 30%
- Class participation: 10%

Policy for Homeworks/Exams

- NOT done in groups
  - do your own work
- Late homeworks increasingly penalized
  - Penalty linearly increasing
  - x days late = (x * 10)% penalty

Course Contents

- What the course IS about:
  formal semantics of programming languages
  - Formal frameworks for describing and reasoning about the computation in a program
  - Tools for formally analyzing and proving properties about the programs
  - Formalism ⇒ lots of math!
- What the course IS NOT about:
  - Survey of modern programming languages or of advanced language constructs
  - Study of implementation techniques (see cs 412)

Overview

- Semantics of programming languages

1. Semantics of a simple language (~40%)
2. Semantics of other language constructs (~40%)
3: Applications to analysis of programs (~20%)

Part I: Basic PL Semantics

- Three main ways to describe semantics:
  - Operational semantics
  - Denotational semantics
  - Axiomatic Semantics
- Discuss relation between different semantics
- Use IMP = simple imperative language
  - Only integers and booleans
  - Only assignments, if statements, and while statements

Part II: Language Features

- Functions
  - Lambda calculus, higher order functions
  - REC = IMP + functions, recursion
- Continuations, scope, references, etc.
- Types
  - Typed lambda calculus
  - Recursive types, subtyping
  - Type inference

Part III: Analysis of Programs

- Standard analyses
  - Dataflow analysis
  - Constraint based analysis
  - Abstract interpretation
- Other approaches
  - Model Checking
  - Theorem proving

Textbooks

- Textbooks:
  - Glynn Winskel: The formal semantics of programming languages
  - Carl Gunter: Semantics of programming languages
  - John Mitchell: Foundation of programming languages
  - Benjamin Pierce: Types and programming languages
  - Flemming Nielson, Riis Nielson, Chris Hankin: Principles of program analysis
- Books are on reserve in Engineering Library