Programming Languages CS442

Overview and Organization

David Toman

School of Computer Science
University of Waterloo

Winter 2006

Course Outline

1. Imperative Programming Languages

   a.k.a. from **while** to "Modula-2/ADA"
   in three easy steps

2. Introduction to Denotational Semantics

   a.k.a. what are these *loops* really mean?

3. Functional Programming Languages
   - untyped λ-calculus (and computation)
   - types in λ-calculus
   - type inference, "practical" languages

4. Logic Programming
   - resolution proofs and unification
   - PROLOG

What is it NOT about

1. “a language a week” (or so):

   $\Rightarrow$ focus on understanding underlying principles

   . . . ok, I’ll talk about SML and Prolog

2. implementation techniques:

   $\Rightarrow$ take *Compilers CS444/644*

What is it NOT about

1. “a language a week” (or so):

   $\Rightarrow$ focus on understanding underlying principles

   . . . ok, I’ll talk about SML and Prolog

2. implementation techniques:

   $\Rightarrow$ take *Compilers CS444/644*

What is it NOT about

1. “a language a week” (or so):

   $\Rightarrow$ focus on understanding underlying principles

   . . . ok, I’ll talk about SML and Prolog

2. implementation techniques:

   $\Rightarrow$ take *Compilers CS444/644*

Organization

Instructor: David Toman
e-mail: `david@uwaterloo.ca`
phone: x4447 (use e-mail—*much* more reliable)

Classes: TT 8:30-9:50 MC 4060

Office: DC 3344 (Tue 3-5pm)

Class Info: `http://db.uwaterloo.ca/~david/cs442`
`news:uw.cs.cs442`

TA's: David Yeung

$\Rightarrow$ office hours, etc., TBA on the web site

Books, Lecture Notes, etc.

Required Textbook:

Structure of Typed Programming Languages.
David A. Schmidt, MIT Press 1995.

Other Books:

Types and Programming Languages.
Benjamin Pierce, MIT Press 2002.

Denotational Semantics: A Methodology for Language Development. David A. Schmidt, Allyn and Bacon 1986.
$\Rightarrow$ [out of print; pertient parts avilable online]

"Semantics of Programming Languages.
Carl A. Gunter, MIT Press 1992.

Lecture Notes:

Additional *lecture notes* and *copies of transparencies* are/will be available from the class WWW page.

Assessment

- lectures (no influence on grade),
- homework assignments: 25% (15% for CS648)
- midterm exam: 32% (30% for CS642)
- final exam: 43% (40% for CS642)
  mark $\geq$ 50% on the final is needed to pass.
- a project: 15% (CS642 only)

Fine print: *the usual university policies on academic honesty, fair use of computing facilities, etc., apply by default.*

Assignments

1. a report that analyzes an existing programming language with respect to various features/constructions discussed in the lectures.
   - you can analyze any language you want
     $\Rightarrow$ I won't be able to help with too esoteric choices
   - deliverables: a PDF file submitted electronically.

2. an implementation of a mini-ML functional language
   - again, you can use any language available on CSCF machines
     $\Rightarrow$ I strongly suggest using SML/NJ