Preliminary version of 6 July

Course Information

Lectures

TBA

Instructor

Darko Stefanovic, office hours TBA

Teaching assistant

TBA

Course topics and format

The course is an informal introduction to the theory used to describe and define programming languages, and to guide their implementation. Our approach is type-based, in the spirit of our textbook, Pierce's *Types and Programming Languages* (TAPL). As a prelude, the course offers an introduction to modern programming techniques and programming language features, focusing on purely functional programming,

The course is intended for first-year graduate students, but advanced undergraduates are welcome as well. No specific prerequisite courses are needed, but programming experience and mathematical maturity are necessary. Facility with formal system manipulation is essential.

The course will provide students with necessary background for CS550 (offered in the spring).

The course consists of lectures, written assignments, programming assignments, two mid-term examinations, and a final examination.

Assignments

There will be two *in-class* midterm exams (the first will be given right before we start on TAPL, the second after we have completed discussion of the simply-typed lambda calculus, i.e., after Chapter 11). There will be a final exam covering the entire course. Several short written homework assignments will be given to consolidate lecture material; they may take the form of short algebraic proofs of program fragment equivalence, or consideration of small language extensions. Several programming assignments will be given: in the early part of the course the tasks will be drawn from the general domains of mathematics, science, and engineering, to practice programming skills; in the latter part of the course the tasks will correspond to the implementation of programming language theory.

Textbooks

Required reading

Benjamin C. Pierce, *Types and Programming Languages*, MIT Press, 2002, ISBN-10: 0262162091.

Optional reading

Graham Hutton, *Programming in Haskell*, Cambridge University Press, 2007, ISBN-10: 0521871727.

Richard Bird, *Introduction to Functional Programming*, Prentice Hall, 2nd edition, 1998, ISBN-10: 0134843460

Robert Harper, *Practical Foundations for Programming Languages* (working draft on-line)

Michael L. Scott, *Programming Language Pragmatics*, Morgan Kaufmann, 2nd edition, 2005, ISBN-10: 0126339511

Grading

You are expected to attend class regularly, *read the assigned reading before class*, and participate in class discussion. The grade will be determined as follows:

Homeworks 50%

Exams 50% (15% each midterm exam, 20% final)

Homework and programming assignment hand-in policy

This course covers a lot of material and being late with assignments will hamper your ability to learn the next section of the course. Therefore, late assignments will be penalized $3n^3$%, where n is the number of days late.

Lecture Plan

1. organizational; functional programming and Haskell introduction
2. prelude types and classes
3. functions and list comprehensions; unit testing; literate programming; interactive programs
4. recursive and higher-order functions
5. declaring types and classes
6. lists in depth: map, filter, and their algebraic laws
7. lists in depth: foldr, scanr, and their algebraic laws

8. trees with folds, binary heap trees, rose trees
9. efficiency: accumulating parameters, tupling, fusion and deforestation
10. modules and abstract data types
11. lazy evaluation and infinite data structures; approximation ordering; cyclic structures; streams
12. monads
13. syntax
14. operational semantics
15. lambda calculus syntax and reduction
16. programming in the lambda calculus
17. combinators and combinator reduction
18. types
19. simply typed lambda calculus
20. simple extensions (ascription; let-bindings; records)
21. simple extensions (variants; recursion)
22. references
23. exceptions
24. subtyping
25. recursive types
26. type reconstruction
27. unification
28. universal polymorphism

Mailing list

A mailing list will be used for class discussion. It may also be used for administrative announcements. See `http://www.cs.unm.edu/cgi-bin/mailman/listinfo/cs591swf`.

UNM statement of compliance with ADA

Qualified students with disabilities needing appropriate academic adjustments should contact the instructor as soon as possible to ensure their needs are met in a timely manner. Handouts are available in alternative accessible formats upon request.