Programming Language Foundations
22C:123
Spring 2004

Professor Ken Slonneger
201G MLH, 335-0745
email: slonnegr@cs.uiowa.edu
web: http://www.cs.uiowa.edu/~slonnegr/plf
2:30 - 3:30 MWF (or by appointment)

Teaching Assistant Brandon Holmes
B20 J MLH, 335-3650
email: bholmes@cs.uiowa.edu
1:30 - 2:30 M, 11 - 12 TT (or by appointment)

Course Objectives

To provide an understanding of formal models of programming language syntax and semantics, in order to provide a deeper understanding of the processes of programming (the construction of correct programs and the formal verification of programs) and programming language definition, design, and implementation.

Textbook (available on Web)

Slonneger and Kurtz, *Formal Syntax and Semantics of Programming Languages: A Laboratory Based Approach*.

Grading			**Tentative Dates**
	Assignments	25%	
	Exam 1	25%	Week of February 23 - 27
	Exam 2	25%	Week of April 5 - 9
	Exam 3	25%	Monday, May 10, 4:30 pm

Prerequisites

The specific prerequisites for 22C 123 are stated in terms of both University of Iowa courses and important concepts from those courses. Individuals who lack the required background should be prepared to do additional work.

1) 22C 34: Discrete mathematics, including set theory, relations and functions, boolean algebra and logic, mathematical induction, and elementary graph theory.

2) 22C 44: Extensive programming experience in high level languages, including programs with recursion, linked lists, dynamic variable allocation, and string manipulation.

3) 22C 54: Familiarity with programming language concepts, including BNF for syntactic description, informal semantic models to describe run-time concepts (binding and the run-time execution stack), language features for typing, data abstraction, and modularity found in languages such as C, C++, Java, and Ada, and familiarity with the various language paradigms, such as functional programming, logic programming, and object-oriented programming.

References:

Dershem & Jipping, *Programming Languages: Structures and Models*

Fischer & Grodzinsky, *Anatomy of Programming Languages*

Ghezzi & Jazayeri, *Programming Language Concepts*

Louden, *Programming Languages: Principles and Practice*

Sebesta, *Concepts for Programming Languages*

Sethi, *Programming Languages: Concepts and Constructs*

Stansifer, *Study of Programming Languages*

Watt, *Programming Language Concepts and Paradigms*

Policies

1. Although the assignments count only 25% toward the final grade, they play an essential role in preparing students for the exams. The syntactic and semantic systems that represent the major topics in the course can only be mastered by practice. The solutions to the assignments are sometimes long and tedious to construct, and students often question the value of such exercises. However, such problems represent the core material of the exams, and the only way to prepare for the exams adequately is to work the problems by yourself.
2. Assignments will primarily be derivations, proofs, and short answer questions plus several Prolog projects.
3. If you have questions about assignment specifications, ask for a clarification in class.
4. Assignments that are submitted will be graded carefully and returned as soon as possible. The primary purpose of assignment grading is to provide feedback on your work prior to exams. Assignments are expected to be submitted on time.
5. Assignments are meant to be *individual projects.*
6. Each student enrolled in 22C 123 will be issued an individual account number for the department workstations. Information on the use of Unix and a text editor is available upon request—however students are responsible for learning how to use the necessary computer facilities.
7. I need to hear from anyone who has a disability, which may require some modification of seating, testing or other class requirements so that appropriate arrangements may be made. Please contact me during my office hours.

Tentative Outline (Order may vary)

	Classes
1. **Specifying Syntax**	3
a) Introduction	
b) Grammars and BNF	
c) The Programming Language Wren	
d) Variants of BNF	
e) Abstract Syntax	
2. **Introduction to Laboratory Activities**	4
a) Introduction to Prolog	
b) Scanning	
c) Logic Grammars	
d) Parsing Wren	
3. **Lambda Calculus**	4
a) Concepts and Examples	
b) Lambda Reduction	
c) Laboratory: A Lambda Calculus Evaluator	
4. **Traditional Operational Semantics**	6
a) Concepts and Examples	
b) SECD: An Abstract Machine	
c) Structural Operational Semantics: Introduction	
d) Structural Operational Semantics: Expressions	
e) Structural Operational Semantics: Commands	
e) Laboratory: Implementing Structural Operational Semantics	
5. **Denotational Semantics**	7
a) Concepts and Examples	
b) A Calculator Language	
c) The Denotational Semantics of Wren	
d) Laboratory: Implementing Denotational Semantics	
e) Denotational Semantics with Environments	
f) Context-sensitive Syntax with Denotational Semantics	

6. **Domain Theory and Fixed Point Semantics**	6
a) Concepts and Examples	
b) Domain Theory	
c) Fixed-point Semantics	
7. **Attribute Grammars**	3
a) Concepts and Examples	
b) An Attribute Grammar for Wren	
c) Laboratory: Context Sensitive Checking for Wren	
8. **Translational Semantics**	3
a) Concepts and Examples	
b) Attribute Grammar Code Generation	
c) Laboratory: Implementing Code Generation	
9. **Axiomatic Semantics**	6
a) Concepts and Examples using Wren	
b) Axiomatic Semantics of Pelican	
c) Proving Termination	
10. **Algebraic Semantics**	0
a) Concepts and Examples	
b) Mathematical Foundations	
c) Application	
Exams	3
Total	45