

# 91.301 Organization of Programming Languages, Fall 2005

## Syllabus

### Contact Information

Prof. Holly Yanco  
Office: Olsen 206  
Lab: Olsen 304  
E-mail: [holly@cs.uml.edu](mailto:holly@cs.uml.edu) (best way to reach me)  
Phone: 978-934-3642

### Class Meetings

Tuesday and Thursday, 11:30-12:45, Olsen 415

### Office Hours

Tuesdays 10:15 to 11:15 in Olsen 206  
Thursdays 1:00 to 3:00 in Olsen 206  
and by appointment.

### Course Description

We will study programming languages using Scheme. You'll learn about salient semantic features of various programming language paradigms including the imperative, functional, logical, and object-oriented approaches. Key concepts include: building abstractions, computational processes, higher-order procedures, compound data, data abstractions, controlling interactions, generic operations, self-describing data, inheritance and message passing, streams and infinite data structures, meta-linguistic abstraction, interpretation of programming languages, machine model, compilation, and embedded languages.

### Textbook

*Structure and Interpretation of Computer Programs*, Second Edition. Harold Abelson and Gerald Jay Sussman with Julie Sussman. MIT Press, 1996

The full text of the book is available on the web at  
<http://mitpress.mit.edu/sicp/full-text/book/book.html>

### Course Website

<http://www.cs.uml.edu/~holly/91.301>

## Software

For the course, will we use Dr. Scheme, which is available as a free download (info about downloading will be on the course web site).

## Exam Dates

Exam 1:           **Thursday, 20 October 2005**, in class

Exam 2:           **Tuesday, 22 November 2005**, in class (NOTE: This is the Tuesday before Thanksgiving. Make your travel plans appropriately so that you can take the exam at its scheduled time.)

Final Exam:      To be determined by the Registrar

## Grading

Homework	30%
Exam 1	20%
Exam 2	20%
Final Exam	30%

## Collaboration Policy

You must do the homework assignments individually. You may discuss the questions with your classmates away from a computer, but you must sit at a computer and program by yourself. To learn, you'll need to actually program in Scheme, not watch another person do it.

Turning in identical (or nearly identical) code violates the collaboration policy. You may not input code into a single file, then turn it in for more than one person. Coding yourself is the only way to learn.

## Homework Policy

Assignments must be turned in on the date they are due in order to receive full credit. Assignments may be passed in up until the next scheduled class meeting to receive 50% credit. After the next class meeting, no credit will be received for assignments, but you may turn them in to have your code read and commented upon.

## Class Schedule

<b>Date</b>	<b>Topic</b>	<b>Reading</b>	<b>Assignment</b>
T 9/6	Course overview Introduction to Scheme	1.1	
Th 9/8	More Scheme intro Substitution model	1.1	1 out
T 9/13	Orders of growth Recursion and iteration	1.2	
Th 9/15	Higher-order procedures	1.3	1 due, 2 out
T 9/20	Compound data Data abstraction	2.1	
Th 9/22	Aggregate data: lists	2.2	2 due, 3 out
T 9/27	Aggregate data: trees	2.2	
Th 9/29	Henderson picture language	2.2.4	3 due, 4 out
T 10/4	Symbolic data	2.3	
Th 10/6	Data structures	2.3, 2.4	4 due, 5 out
T 10/11	Multiple representations of data	2.4	
Th 10/13	Generic Operators	2.5	5 due
T 10/18	State	3.1	
Th 10/20	Exam 1, includes material from lectures through 10/13		
T 10/25	Environment model	3.2	6 out
Th 10/27	Object-oriented programming	Handout	
T 11/1	Mutable data	3.3	6 due, 7 out
Th 11/3	More mutation	3.3	
T 11/8	No Class: Friday Schedule		
Th 11/10	Streams	3.5	7 due, 8 out
T 11/15	Streams	3.5	
Th 11/17	Metacircular evaluator	4.1	8 due
T 11/22	Exam 2, includes material from lectures through 11/16		
Th 11/24	No Class: Thanksgiving		
T 11/29	Metacircular evaluator	4.1	9 out
Th 12/1	More on MC eval: lazy evaluation	4.2	
T 12/6	More on MC eval: amb evaluation	4.3	9 due, 10 out
Th 12/8	Memory management and garbage collection	5.3	
T 12/13	OPL Jeopardy		10 due
TBD	Final Exam		