Programming Languages

Rensselaer CSCI-4430/6969

April 24, 2007

This course is being offered in the Spring 2007 semester. It meets Mondays and Thursdays 2-3:50pm in Darrin 318, and on the Web in Rensselaer LMS. The lectures and class discussions are an important part of the course but so is the Web site. Check it frequently for the latest information about the course and for material not covered in the lectures or textbook.

General Information

Instructor and teaching assistants

Instructor:

Dave Musser, 276-8660, `musser[at]cs.rpi.edu`
Office Hours: Lally 313, Tuesday & Wednesday 2:30-4 pm or by appointment

Teaching Assistants:

Ying Yang, `yangy2[at]rpi.edu`
Office Hours: Lally 09, Tuesday 10:00 am - 12:00 pm

Jon Purnell, `purnej[at]rpi.edu`
Office Hours: first floor, 2009 15th Street (between Sage and Peoples Avenues), Monday 4:00 - 6:00 pm

Anurat Chapanond, `chapaa[at]rpi.edu`
Office Hours: Amos Eaton 217, Thursday 11:00 am - 12:00 pm

When you have questions outside of class, either visit one of us or send email. Further instructions as to which persons to direct email to will be given but may vary during the semester, or for different purposes, so it's a good idea to check here first for the latest advice.

Brief overview

This course is a study of important concepts of modern programming languages. Topics include

- Programming Language Syntax and Semantics
- Abstraction: Data Abstraction and Control Abstraction
- Programming Paradigms: Declarative, Declarative Concurrent, Object-Oriented, and Logic Programming; possibly others

Emphasis is on evaluation of a programming language or programming paradigm based on its underlying semantic concepts. As a unifying basis for demonstrating, experimenting with, and comparing different concepts, we will use the Oz language and its supporting programming environment, Mozart. Comparisons will also be made with languages that exemplify a particular paradigm, such as C++, Erlang, Haskell, Java, Scheme, ML, Prolog. Note: This is *not* an "Oz course." The main subject matter is, again, programming language *concepts*. Oz is a useful tool in this context because it exhibits in a single language most of the important concepts of today's widely used programming languages. You may or may not also find that Oz is useful in itself for serious future programming projects, but if you do, consider it a bonus — it's not the main goal of this course.
More overview is presented in Lecture 01.

Prerequisites

CSCI-2400 Models of Computation or equivalent.

> *From the Rensselaer Catalog:* **CSCI-2400 Models of Computation** This course introduces conceptual tools for reasoning about computational processes and the languages with which they are prescribed. It bears directly upon language translation, program verification, and computability. Topics to be covered include formal languages, finite automata, pushdown automata, nondeterminism, regular expressions, context-free grammars; parsing, compiler design basics; computability, Turing machines, Church's thesis, unsolvability and intractability. Prerequisites: CSCI-2300 and MATH-2800. Fall and spring terms annually. 4 credit hours

Homework and exams

There will be four to six homework assignments. Late homework will not be accepted. There will be two exams, one at midterm and a final exam; each will cover all the material up to that point but with greater emphasis on topics discussed since the previous exam. Exams will be designed to test both understanding of concepts and problem solving skills.

The course grade average will be determined as follows:

Homework	25%
Midterm Exam	35%
Final Exam	40%

Grade averages required for each letter grade are as follows:

A	90%
B	80%
C	70%
D	60%
F	$< 60\%$

The D letter grade is not available to graduate students (Grad School rule). One additional rule: you cannot pass the course unless you have a passing average on the exams.

Additional details The homework assignments are where you learn technical skills and gain understanding of important concepts working at your own pace; it is therefore important that you do them on your own. Instructions for submitting the homework assignments will be given.

There is no late submission for homework assignments. They will not be accepted after the deadline, and a grade of F will be assigned. Extensions may be arranged only in the event of severe illness, dire emergencies, and religious observances. Please present appropriate documentation to me regarding the nature and duration of the illness or emergency ASAP thereafter; documentation is also needed for any religious observances not listed in the Rensselaer calendar.

The exams will be given in class. You are expected to arrive at the beginning of the class period, and no extra time will be given for late arrivals.

Try to make it to the exams! In case of any emergency you don't have to worry, but you have to inform me *in advance* and I need a *written and signed explanation.* If I accept your excuse you have to take an oral exam (which, due to the nature of oral exams, is usually more difficult). If you miss an exam without an excuse there will be no oral exam and the exam will be graded F with 0 points.

Extra requirements for some students If you are a graduate student taking the course to satisfy the Programming Languages Component of the Ph.D. Qualifying Exam you'll need an A average on the exams in addition to an A average for the course grade. (This is a separate issue from taking the course for graduate credit, which is discussed next.)

All students taking the course for graduate credit (registered for CSCI-6969) will be assigned extra homework problems, usually ones delving a little deeper into theoretical issues, such as proofs of functional correctness or deriving or proving properties of programming language semantics. Grades on these problems will be included in computation of the homework average (which will remain as 25% of the course grade).

In addition, CSCI-6969 students should volunteer to give a presentation in class, either

- about a programming language topic that is not definitely scheduled for a lecture, such as one of the topics marked "tentative" in the Schedule. Another programming

language related topic can be proposed instead, subject to my approval. These presentations may be from 15 to 35 minutes (discuss the proposed length with me in advance);

- or about a programming-language-related *tool*. Example topics: a debugger, a profiler, a static analyzer, a comparison of different compilers in terms of error reporting, or in terms of code optimization, etc. Such tools exist for Oz, but tools for other languages can be chosen, particularly if you already have experience with them. These presentations may be from 10 to 15 minutes, and should include a live demonstration of the tool (not just slides about it).

In either case, the presentation should be prepared and presented by two students working as a team. Please find a partner and discuss with me your presentation proposal as early as possible but no later than March 29. (Tool presentations can be scheduled earlier than that.) [**Added 3/29:** The deadline for proposing a presentation is extended to April 5.]

These presentations will not be graded. (In the past, however, such presentations have helped me say favorable things in letters of recommendation.)

Collaboration and other academic integrity issues

Except for assignments in which it is specifically permitted to work in pairs, homework must be done individually: copying or allowing another student to copy one's work is not permitted. Nor is it permissible to use material from other sources without proper attribution. *You are encouraged to use the Internet and its resources but you have to give the original author(s).* Any acts of plagiarism or other anti-intellectual behavior will be dealt with severely: the assignment or exam will receive a grade of zero, and two or more incidents will result in a failing grade for the course.

Using Rensselaer LMS

The course Web site is maintained in Rensselaer LMS[1] (formerly WebCT).

Schedule Follow the Schedule link to see the schedule of lecture topics, reading assignments, and homework and exam dates. More detail and suggestions regarding reading selections will be given in the lecture notes.

Homework Homework solutions should be submitted electronically by uploading files in LMS. Once you are in the "Programming Languages" site in LMS, click on the link for "Homework." Detailed instructions for submission of each assignment will be given in each assignment but watch for any updates in the Homework section in LMS.

[1] http://rpilms.rpi.edu

Grades Clicking on the "My Grades" link should show a page with all of your grades. Clicking on each field title produces a table of statistics for all students' performance on that assignment or exam.

Schedule

This schedule is subject to change—check the course Web site frequently for updates.

The reading assignment for each date is to be done *before* class on that date.

VRH = The Van Roy and Haridi textbook

The first few lecture titles are linked to the lectures notes; for more recent lecture notes do an `svn update`. (Lecture notes will usually be made available in the repository on the morning of the lecture).

DATE	LEC	TOPIC	TEXT	READINGS	HW
1/18	1	Organization, Overview, Getting Started			
1/22	2	Programming Concepts: Compound Values, Partial Values, Lists, Pattern Matching	VRH	Preface [Browse as you like], 1.1-1.9, Appendix A.1 for Mozart/Oz, Appendix B.1	TO DO: install Mozar HW 1 out (in lecture r
1/25	3	Programming Concepts: Correctness, Complexity, Higher-Order Programming, Towards the Model	VRH	1.10-1.18, 2.1	TO DO: install Subversion
1/29	4	Syntax Specification; Kernel Language Syntax	VRH	2.2 & 2.3 [careful], 2.4 [browse]	
2/1	5	Semantics, Kernel Language, Computational Model	VRH	2.4 [careful], 2.5 [browse]	
2/5	6	Abstract Machine	VRH	2.4 [again]	HW 1 due
2/8	7	Abstract Machine: Procedures, Lexical Scoping	VRH	2.5 [careful]	HW 2 out
2/12	8	Recursive Procedures and Last Call Optimization	VRH	2.5 [again]	
2/15	9	From Kernel Language to Practical Language; Exception Handling	VRH	2.6 [careful], 2.7 [careful]	
2/20	10	(Tues) Iterative Computations	VRH	3-3.2, 3.4-3.4.2.6	HW 2 due
2/22	11	Higher-order Programming	VRH	3.4.8 (Parsing), 3.6	
2/26		**Midterm Exam**			
3/1	12	Abstract Data Types	VRH	3.7	
3/5		SPRING BREAK			
3/8		SPRING BREAK			

DATE	LEC	TOPIC	TEXT	READINGS	HW
3/12	13	Declarative Concurrency: Threads, Data Flow, Streams	VRH	4.1-4.3	
3/15	14	Lazy Evaluation	VRH	4.4-4.5.3	HW 3 out
3/19	15	Demand-Driven Evaluation: RC-Circuit Example, auxiliary file: SimGraph.oz	VRH	4.5	
3/22	16	Message Passing/Agents	VRH	5-5.3	
3/26	17	Agents With State; Erlang	VRH	5.4, 5.7	
3/29		No Class			
4/2	18	Explicit State	VRH	6-6.3	HW 3 due
4/5	20	Reasoning with State; Object Oriented Programming	VRH	6.6, 7-7.2	
4/9	21	Object Oriented Programming	VRH	7.3-7.5, 7.7	HW 4 out
4/12	22	Active Objects	VRH	7.8	
4/16	23	Relational Programming	VRH	9-9.3	
4/19	24	Natural Language Parsing; Prolog	VRH	9.4, 9.7	
4/23	25	Lambda Calculus (Medha Atre); Intel C++ Compiler Optimization (Yu Sheng & Jiao Tao); Foreign Language Support in Matlab (Christos Boutsidis)			
4/26	26	Foreign Language Support in R (Akintayo Holder); Distributed Programming: The Oz Model and the Google Model (MapReduce)	VRH	11-11.6	HW 4 due
4/30	27	GUI Programming (Edward Levee & Andile Metfula); Review	VRH	10	
5/9		**Final Exam** (Wed 11:30-2:30)			

Resources

Textbook

The following textbook is required:

> Peter Van Roy and Seif Haridi, *Concepts, Techniques, and Models of Computer Programming*,[2] MIT Press, Cambridge, MA, 2004, ISBN 0-262-22069-5.

Languages

We'll study concepts that appear in many languages, and discuss some of the details of how those concepts are manifested in one or more of the languages C++, Haskell, Java, ML, Prolog, and Scheme. But the only language on which assignments will be directly based is Oz (exception: students taking the course for graduate credit might work directly with another language as part of their extra assignments).

Oz and Mozart

First, if you don't already have Emacs installed on your laptop, install it according to the instructions below.

Then visit The Mozart Programming System[3] and download and install the version on your laptop corresponding to the OS you are running (Windows, Linux, FreeBSD, or OS X). See Appendix A.1 (through A.1.1 for now) for a brief introduction; more detailed system documentation is on the Mozart Web site. Some of the more advanced features will be discussed as needed in the course.

A few examples in the textbook require the following file (from the book's Supplements Web site):

> booksuppl.oz

General software tools

Emacs

GNU Emacs, "the extensible, customizable, self-documenting real-time display editor," is available on most campus computers. The Mozart programming environment runs on top of Emacs, so it is essential for this course. If you are running Linux, you already

[2] http://www.info.ucl.ac.be/people/PVR/book.html
[3] http://www.mozart-oz.org

have it; if you are running Windows and don't already have Emacs installed, visit the GNU Emacs[4] Web page.

If you are new to Emacs, try the built-in tutorial (type `Esc-x help t`). Its "info system" (type `Esc-x info`) provides an online reference manual. To gain additional proficiency, consider also signing up for the short course on Emacs offered by the Campus Computing Center in February(?).

Cygwin Tools

The Cygwin Tools are a set of freeware compilers and other programming tools (such as debuggers), and utility programs (ranging from the BASH shell to X Windows) for Windows 95/98/NT/2000/XP. They provide many of the same useful tools you would find on a typical Unix platform.

If you have a Windows box and want to install the Cygwin Tools on it, download the installation file for your OS version from the Cygwin Tools[5] Web page.

Subversion

From the Subversion[6] project home page: "The goal of the Subversion project is to build a version control system that is a compelling replacement for CVS in the open source community." Subversion is fully documented in the Subversion Book,[7] which is available in various electronic formats. The main part of the book needed for use in this course is "Basic Work Cycle: Chapter 3. Guided Tour."

Subversion is installed on CSLAB FreeBSD and Solaris machines. You should also install it on your personal laptop or PC.

[**Added 1/25:** The following problem has come up: With new installations of the Subversion client on Windows laptops, there seems to be a bug in the interaction between the client and the CS Lab Subversion server, which results in the message `500 Internal Server Error`. CS Lab staff is looking into this but in the meantime you should be able to get around the problem by installing this older version: http://subversion.tigris.org/files/documents/15/23230/svn-1.2.0-setup.exe You can install this without first un-installing an existing version. (Remember, this is only for Windows platforms.)]

[**Added 2/1:** With svn-1.2.0, some people have encountered this error:

```
$ svn update svn:  This client is too old to work with working copy '.';
please get a newer Subversion client
```

[4] http://www.gnu.org/software/emacs/emacs.html
[5] http://www.cygwin.com
[6] http://subversion.tigris.org
[7] http://svnbook.red-bean.com

What apparently happened was that they had managed to check out the repository with svn version 1.4.something, then they had trouble with that and installed 1.2.0, but that doesn't work with the already checked out files. The workaround is simply to check out a new copy of the materials directory and do all future updates in it.]

On any file system in which you keep files related to the course, you should do an initial `svn checkout` and frequent `svn update`s of the checked out directories.

If you don't already have a local directory for course files, make one and `cd` into it. Then do

```
svn checkout https://svn.cs.rpi.edu/svn/csci4430s07/materials materials
```

You will be asked to authenticate. **CONFUSION FACTOR: The popup window asks for your *CS Lab* username and password, which is what is normally used with CS department subversion access, but what you should actually provide in this case is your *RCS* username and password.**

[**Added 1/30:** As mentioned in class last week, if you use have the same username for CS Lab and RCS and you use the same password for both, authentication fails. You must change the password for one or the other, and then use the RCS password.]

This should result in a subdirectory `materials`, with several subdirectories and files within them, being placed in your local course directory.

Thereafter, you can at any time get the latest file updates simply by entering your local `materials` directory and doing

```
svn update
```

(If you do this while `cd`'d into one of the subdirectories of `materials`, you'll only get updates for that subdirectory and any of its subdirectories.)

You can also view the `materials` files using a Web browser. Just click on the URL https://svn.cs.rpi.edu/svn/csci4430s07/materials here or enter it in the address bar. (You'll be asked to authenticate.)

[**Added 2/5:** Everyone in the class now has a private area of the course repository in which you can commit files. First, cd to where you want to have a local copy (*not* within your local copy of the `materials` area) and do a checkout

`svn checkout https://svn.cs.rpi.edu/svn/csci4430s07/private/`*username* `myarea`

where *username* is your RCS username. This will create a subdirectory called `myarea` (choose another name if you prefer). Next, cd into this directory and create a little text file, `test.txt`, containing, say, the line `This is a test.`. Then do

```
svn add test.txt
svn commit -m "Initial test of committing to private area."
```

You should see messages like

```
Sending test.txt
Transmitting file data .
Committed revision 54.
```

(except the revision number will be higher).

Next, for my benefit, please copy any file containing a photo of yourself into this directory, `svn add` it (Subversion will ignore it if you don't `svn add` it), and do another `svn commit`. Then do the same with one or more text files in which you tell me a little about yourself. (If you already have a CV you could use that, but I'll welcome any information or format you want to use.)

Your `private/`*username* directory in the repository is not entirely private; the TAs and I have read/write access to it. But each student has access (both read and write) only to his or her own `private/`*username* directory.]

FreeBSD and Solaris machines

The Mozart Programming System is available on the CS Department's FreeBSD and Solaris lab machines. Remote access to these machines is available by logging into `freebsd.remote.cs.rpi.edu` or `solaris.remote.cs.rpi.edu`. You need to use a remote login program that supports the Secure Shell protocol (SSH) for logging in to lab equipment. If you don't already have one, see CS Lab Staff's Computing FAQ[8] page. Once you are logged in, type `oz &` at the command prompt, which will start Emacs/Mozart in a window on your computer. This assumes your computer is set up to run X Windows; if you are running the Windows OS and that's not the case, install Cygwin/X (see the Cygwin Tools section).

Exams

Midterm - Monday, February 26

Midterm Exam Solution Key

Exam scores will be available on LMS before Friday, March 9. The exams will be returned in class on March 15.

Rescaling: raw scores on the exam (0-72 points) will be rescaled to a 100 point scale. For example,

Raw		Rescaled
72	→	100
58	→	90
46	→	80
35	→	70
26	→	60

[8] http://www.cs.rpi.edu/~lindss2/faq.html

The rescaled score is what will be used in computing the course grade.

(This is not the same as a "curve" of the grades, since it's something I decide on after making up the exam based on my judgement about its difficulty, before the exam is given.)

The following instructions were given for exam: The Midterm Exam will be given Monday, February 26, in the regular class time and room. It will be closed book: no notes, printed material, computers, calculators, or communicating devices may be used. In addition to the lecture notes and readings in the text, review also the in-class labs and homework assignments, including the solution keys provided on the Web.

Here are an old Midterm Exam and solution key, from a previous edition of the course with approximately the same syllabus.

Old Midterm Exam

Old Midterm Exam Solution Key

Final

The Final Exam will be given Wednesday, 5/9, 11:30am-2:30pm. It will be closed book: no notes, printed material, computers, calculators, or communicating devices may be used. In addition to the lecture notes and readings in the text, review also the in-class labs and homework assignments, including the solution keys provided on the Web.

Most, and perhaps all, of the questions on the final will be directly on the material covered since the midterm exam. But keep in mind that a good understanding of the concepts covered in the new material depends heavily on a good understanding of the concepts covered in the first half of the course.

Here are an old Final Exam and solution key, from a previous edition of the course with approximately the same syllabus.

Old Final Exam

Old Final Exam Solution Key