Spring10

Online Book	Owlspace Course Site	Office Hours	SVN Guide	Scheme HW Guide	Java HW Guide	HW Checklist	HW Grading

COMP 211: Principles of Program Design

Instructors:	Prof. Robert "Corky" Cartwright	**Staff:**	Shams Imam
	Prof. Vivek Sarkar		Dragos Sbirlea
	Dr. Zung Nguyen		Alina Simion
			Mina Yao
			Robert Brockman
			Kiran Nair
Lectures:	Duncan Hall (DH) 1064	**Time:**	MWF 10:00-11:50am
Labs:	Ryon 102	**Times:**	Monday 2:00-3:20 pm, Monday 3:30-4:50 pm, Tuesday 2:30-3:50 pm.

Introduction

This course is an introduction to the fundamental principles of programming. The focus is on systematic methods for developing robust solutions to computational problems. Students are expected to have experience writing interesting programs in some credible programming language (e.g., Python, Java, Scheme, C#, C++, Visual Basic .NET, PRL, Scheme, Lisp, etc.) but no specific programming expertise is assumed. The course is targeted at potential Computer Science majors but mathematically sophisticated non-majors are welcome. We expect students to be comfortable with high-school mathematics (primarily algebra, mathematical proofs, and induction) and the mathematical rigor and vocabulary of freshman calculus. Success in the course requires a deep interest in the foundations of computer science and software engineering, self-discipline, and a willingness to work with other people on programming projects. Topics covered include functional programming, algebraic data definitions, design recipes for writing functions, procedural abstraction, reduction rules, program refactoring and optimization, object-oriented programming emphasizing dynamic dispatch, OO design patterns, fundamental data structures and algorithms from an OO perspective, simple Grapical User Interfaces (GUIs), and an exposure to the challenges of concurrent computation.

Students will learn the practical skills required to write, test, maintain, and modify programs. Labs and assignments use the Scheme and Java programming languages.

Text For Scheme: How to Design Programs by Felleisen et al. QA76.6 .H697 2001 (Available online; no purchase is necessary.)

DrScheme: Please download and use the DrScheme system available from the PLT Scheme web site. To avoid compatibility problems, please make sure you use a Version numbered 4.2.x. As of 2-1-10, the latest version is 4.2.4, which fixes some bugs in 4.2.3.

Notes for Java: Object-oriented Design

References for Java

- Principles of Object-Oriented Programming by Zung Nguyen and Stephen Wong. An online self-contained introduction to OOP in Java roughly corresponding to the former Comp 212 course. It is reasonably complete, but still under construction.
- Index to online Java Tutorials by Sun Microsystems The Sun tutorials refer to the full Java language not the Elementary and Intermediate language levels supported by DrJava. Nevertheless, they cover many important language details in depth, such as the complete collection of primitive operators on primitive data types.
- Java Basics by Fred Swartz is a clearly written traditional introduction to Java that focuses on Java mechanics rather than OO Design. It can be helpful in learning the {\em mechanics} of writing full Java code. Please ignore what he says about program design.
- Java Notes by Fred Swartz is a reasonably comprehensive Java reference that is a good supplement to the official Sun documents.

DrJava: Please download and use the DrJava pedagogic programming environment available from drjava.org. You must install either the Java 5 or Java 6 JDK on your machine for DrJava to work. If you machine is running some flavor of Windows or Linux, go to the \[Sun Download Site for the Java SE 6.0 (http://java.sun.com/javase/downloads/index.jsp\]). Make sure that you download the **JDK** not the **JRE**. If you have a Mac, a Java JDK is available from Apple. In fact, it is part of the standard Mac OS X software package. DrJava will run on either a Java 5.0 or Java 6.0 JDK.

Entrance Survey

Please fill out this survey.

Pick a Lab Section

Every student must attend an assigned lab section each week. Lab sections meet M 2-3:25pm, M 3:30-4:55pm, and T 2:30-3:55 in Ryon 102 (the CLEAR lab room on the ground floor of the Ryon Building). If you have not already contacted Dr. Nguyen (dxnguyen@rice.edu), immediately send him an email message stating your preferred lab sections (first and second choices).

Computing Environment

All Comp 211 programming assignments will be run and graded on the new CLEAR educational computing facility. For instructions on how to use CLEAR, see the CLEAR web page.

Errata

The Confluence wiki for Comp 211 is new and presumably full of broken links and typos. If you notice an error in the wiki, please send email to the Comp 211 staff.

Course Schedule

Note that future date schedules are only guidelines. Future homeworks and slides may contain materials from previous Comp 210 and Comp 212 classes. New material will be provided before the corresponding class. There will only be two exams in the course: one given on functional programming during week 7 and one on object-oriented programming given during in the last week of the course. Both are take-home exams. There is no final examination.

	Day	Date (2009)	Topic	Reading	Lectures	Problems	Due	Lab	Supplements
1	Mon	Jan 11	Introduction		L 1	HW 0	W Jan 13	Lab 0	Pair Programming
2	Wed	Jan 13	Scheme primitives; function and data definitions	Skim Chs. 1-10	L 2	HW 1	F Jan 22		
3	Fri	Jan 15	Inductive data, conditionals, and the design recipe	Review Chs. 1-10	L 3				
-	Mon	Jan 18	School Holiday						
4	Wed	Jan 20	Data-directed design I	Read Chs. 11-13	L 4			Lab 1	
5	Fri	Jan 22	Data-directed design II	Read Chs. 14-15	L 5	HW 2	Fri Jan 29		
6	Mon	Jan 25	Mutually Referential Data Definitions	Skim Chs. 16-17	L 6			Lab 2	
7	Wed	Jan 27	Local definitions and Lexical Scope	Read Ch. 18	L 7				
8	Fri	Jan 29	Functions as Values	Read Chs. 21-22	L 8	HW 3	Fri Feb 5		
9	Mon	Feb 01	Functional Abstraction and Polymorphism	Read Chs. 19-20	L 9			Lab 3	
10	Wed	Feb 03	Lambda the Ultimate	Read Ch. 24	L 10				
11	Fri	Feb 05	Generative Recursion	Read Chs. 25-28	L 11	HW 4	Fri Feb 15		
12	Mon	Feb 08	Generative Recursion Illustrated	Study Chs. 25-28	L 12			Lab 4	
13	Wed	Feb 10	Complexity and Accumulators	Read Chs. 29.1-2 Skim Chs. 30-32	L 13				
14	Fri	Feb 12	Accumulators and Tail Calls	Read Chs. 30-32	L 14	HW 5	Fri Feb 19		
15	Mon	Feb 15	Clever Programming With Functions	Review prior readings	L 15			Lab 5	210 Exam 1 210 Exam 2
16	Wed	Feb 17	Vectors and Iteration	Review prior readings	L 16				
17	Fri	Feb 19	Exam Review	Review prior readings	L 17	HW 6 (optional) Exam 1	Fri Feb 26		
18	Mon	Feb 22	On to Java	OO Design Notes Ch 1.1-1.4	L 18			Lab 6	parse/unparse
19	Wed	Feb 24	Java Design Recipe	OO Design Notes Ch 1.1-1.4	L 19				
20	Fri	Feb 26	Defining Inductive Data in Java	OO Design Notes Ch 1.5	L 20	HW 7		IntList.dj1 IntListTest.dj1	
-	M-F	Mar 01-05	Spring Break						

21	Mon	Mar 08	Static Class Members and the Singleton Pattern	OO Design Notes Ch 1.6	L 21			Lab 7	ObjectList.dj1 ObjectListTest.dj1
22	Wed	Mar 10	Polymorphism and Interfaces	OO Design Notes Ch 1.8	L 22				ComparableList.dj1 ComparableListTest.dj1
23	Fri	Mar 12	Handling Exceptions and Errors	OO Design Notes Ch 1.9-1.10, 1.12	L 23	HW 8	Fri Mar 13		
24	Mon	Mar 15	The Strategy and Visitor Patterns	OO Design Notes Ch 1.9, 1.11	L 24			Lab 8	IntList.dj1 IntListVisitor.dj1 IntListTest.dj1
25	Wed	Mar 17	Visitors, Visitors, Vistors ...	OO Design Notes Ch 1.11	L 25				
26	Fri	Mar 19	Full Java, Arrays, Mutation	OO Design Notes Ch 1.13	L 26	HW 9	Wed Mar 31		IntList.java IntListTest.java
27	Mon	Mar 22	Visibility, Type-Checking, and Generics	OO Design Notes Ch. 1.10, 1.13	L 27			Lab 9 List.java ListTest.java	
28	Wed	Mar 24	Generics with Discretion		L 28				
29	Fri	Mar 26	Mutation and Bi-Directional Linked Lists	OO Design Notes Ch 1.13	L 29				
30	Mon	Mar 29	Graphical User Interfaces	OO Design Notes Ch 3	L 30			Lab 10	
31	Wed	Mar 31	Anonymous Inner Classes and Task Decomposition	OO Design Notes Ch 2.1	L 31	HW 10			BigBiList.java BigBiListTest.java
-	Fri	Apr 2	School Holiday						
32	Mon	Apr 5	Mutable Trees	OO Design Notes Ch 2.1	L 32			Lab 11	TreeMap.java TreeMapTest.java OOTreeMap.java OOTreeMapTest.java
33	Wed	Apr 7	Review: Confronting the Reality of Full Java		L 33				
34	Fri	Apr 9	QuickSort Revisited		L 34	HW 11			FunctionalQuicksort.java CallableQuicksort.java CallableQuicksortAlternateOrder.java ParallelQuicksort.java QuicksortWithComparable.java QuicksortTest.java
35	Mon	Apr 12	Graphical User Interfaces II	OO Design Notes	L 35			Lab 12	
36	Wed	Apr 14	OO Sorting Algorithms		L 36				Design Patterns for Sorting Design Patterns for Sorting SIGCSE paper.pdf Sorter Demo code.zip
37	Fri	Apr 16	Fast Searching with Balanced Trees		L 37	HW 12			Red-Black Trees Animated PPT OOPSLA paper on balanced trees Executable demo Demo source code
38	Mon	Apr 19	Fast Searching and Memoization		L 38			Lab 13	MyHashMap.java MyHashMapTest.java BetterMath.java
39	Wed	Apr 21	Parallel Programming Tradeoffs		L 39				
40	Fri	Apr 23	Exam II Review		L 40			Lab 14	

Grading, Honor Code Policy, Processes and Procedures

Grading will be based on your performance on homeworks (worth 50%) and exams (20% for first exam, and 30% for the second exam).

Take-home exams, which are pledged under the honor code, test your individual understanding and knowledge of the material. Collaboration on exams is strictly forbidden.

Mailing Lists:

- comp211-discussion-l@mailman.rice.edu
  (subscribe here (https:mailmanriceedumailmanlistinfocomp211discussionl)):
  - This is where important announcements related to the class will be posted.
  - Students are required to sign up to this list.
  - You may use this list for open discussions relating to the course.
    Postings are expected to abide by standard Netiquette.
- cs-events-l@mailman.rice.edu:

- Announcements relating to talks and other interesting events hosted by the CS departments.
- Subscription to this list is optional but highly recommended

Questions

If you have a question about homework, you can raise the question with a TA in lab or on the discussion mailing list. If, after doing so, you don't feel that your concerns have been addressed, you may wish to contact Prof. Cartwright or Prof. Sarkar directly.

Homeworks:

Homeworks help you verify your understanding of the material and prepare you for the exams. You are encouraged to discuss the homework problems with the instructors and staff. Help from other students, including Comp 211 graduates, is also encouraged (but should be cited), although that does *not* include giving or receiving complete answers. All homework partners are responsible for knowing all the submitted material. If you fail to understand the homework solutions, you won't succeed on the exams.

Homeworks will generally be handed out on Fridays, and will be due before class the following Friday.

You are expected to work in groups of two.

You may change partners during the semester.

Partners should work together on all aspects of the homework -- all students are expected to contribute equally. You and your partner should hand in exactly one solution.

Late homework will not be accepted with one exception. Every student is allotted 7 slip days. Each whole day or fraction of a day that an assignment is late counts as a slip day. Each student in the pair submitting a late assignment must spend the requisite number of slip days. Since assignments get progressively harder during the semester, we strongly encournage you to hoard your slip days for use near the end of the term.

We recommend that you review the homework guide as you develop your solutions. Review the submission checklist when you turn in your homework. Your work will be graded as documented on the grading page.

Reading: For each lecture, there is associated reading. Students are required to complete the reading before the class associated with this reading.

h2 Other Resources

- Practical matters:
  - DrScheme Programming Environment
- Special interest groups:
  - CSters
  - Computer Science Club
  - ... (please send in suggestions!)

Additional References

Here is a nice article about the basic approach taken in this course.

More on CS

The New Turing Omnibus, A. K. Dewdney	QA76 .D448 1993
Algorithmics: The Spirit of Computing, David Harel	QA76.9 .A43 H37 2004
Computers Ltd.: What They Really Can't Do, David Harel	QA76.5 .H3575 2000
Great Ideas in Computer Science, Alan W. Biermann	QA76 .B495 1997
Computer Science: An Overview, J. Glenn Brookshear	QA76 .B743 1997
Gödel, Escher, Bach: An Eternal Golden Braid, Douglas Hofstadter	QA9.8 .H63 1980
Metamagical Themas, Douglas Hofstadter	Q335 .H63 1985

More on Functional Programming

Google's MapReduce	(Online)
The Little Schemer, Friedman & Felleisen	QA76.73 .S34 F75 1996
The Seasoned Schemer, Friedman & Felleisen	QA76.73 .S34 F77 1996
Developing Applications with Objective Caml, Emmanuel Chailloux, Pascal Manoury, and Bruno Pagano	
The Haskell School of Expression: Learning Functional Programming Through Multimedia, Paul Hudak	QA76.62 H83 2000

More on Object-Oriented Design in Java

Principles of Object-Oriented Programming	The Connexions Curriculum for the former Comp 212 course
The Java Programming Language, Arnold & Gosling	Gosling was the principal designer of Java 1.0
Thinking In Java, Bruce Eckel	The 3rd edition is available free.
Design Patterns: Elements of Reusable Object-Oriented Software, Gamma, Helm, Johnson & Vlissides	The "Bible" of OO software design.
Head First Java, Bert Bates & Kathy Sierra	A fun read while you get introduced to Java and learn how to think like a Java Programmer.
Head First Design Patterns, Freeman & Freeman	An accessible introduction to Design Patterns.

More on Data Structures and Algorithms

Introduction to Algorithms: Book written by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. The authoritative reference on algorithms.

Accommodations for Students with Special Needs

Students with disabilities are encouraged to contact me during the first two weeks of class regarding any special needs. Students with disabilities should also contact Disabled Student Services in the Ley Student Center and the Rice Disability Support Services.