

# CS 2210 Syllabus

Fall 2004

Class web page: <http://www.cs.pitt.edu/~mock/cs2210/>

Class mailing list: [cs2210@cs.pitt.edu](mailto:cs2210@cs.pitt.edu)

---

## Instructor & T.A.:

	Name	Office	Email	Office Hours
Professor	Markus U. Mock	6407 SENSQ	<a href="mailto:mock@cs.pitt.edu">mock@cs.pitt.edu</a>	MW 4:00 – 6:00 and by arrangement
TA	Rupa Natarajan	6406 SENSQ	<a href="mailto:rnataraj@cs.pitt.edu">rnataraj@cs.pitt.edu</a>	TBD

## Time and Place:

Lectures	MW	2:30-3:45	5129 SENSQ
----------	----	-----------	------------

## Course description:

The implementation of high level programming languages. We will cover the structure of compilers, lexical, syntactic and semantic analysis. After covering the basics, the main focus will be data flow analysis and program optimization. We will learn about code generation for modern architectures, including some backend optimizations such as register allocation and code scheduling. Run-time issues, such as memory allocation are another topic you will learn about in this class.

## Course prerequisites:

CS1621 Structure of Programming Languages, or an equivalent course, or the consent of instructor. This course requires some programming experience and some basic knowledge of assembly language. Also some familiarity with basic concepts of lexical and syntax analysis are assumed, for example, the equivalence of finite automata and lexical expressions. Therefore, lexical and syntax analysis will be covered speedily – if your background in these topics is weak, you should expect to put in some extra time at the start of the course to review this material (covered mainly in the “Dragon Book”).

If you are unsure whether you meet these requirements, talk to the instructor.

### Textbooks:

Required	Steven S. Muchnick: "Advanced Compiler Design and Implementation", 1 <sup>st</sup> edition, 3 <sup>rd</sup> printing, Morgan Kaufmann Publishers 1997.
Recommended	A. Aho, R. Sethi, and J. D. Ullman: "Compilers: Principles, Techniques, and Tools", 2nd edition, Addison-Wesley, 1986, aka. the Dragon Book.
Reference	Robert Morgan: "Building an Optimizing Compiler", Digital Press, 1998.

The reference and recommended textbooks are on reserve in the Library. The text that covers the first few weeks of course material is the Dragon Book. We will cover this material relatively quickly, at normal pace for those familiar with the material from an undergraduate course (finite and pushdown automata or undergraduate compilers / programming languages). If you are unfamiliar with this material, this is not a problem, but expect to put in some extra time to come up to speed on it. After that, all students will be on the same page.

### Home Page:

The class home page for the course is <http://www.cs.pitt.edu/~mock/cs2210>. All course handouts, answers to frequently asked questions, lecture notes, and updates on assignments will be posted. Please check the home page and mailing list regularly for important course information. Students are responsible for changes to assignments posted on the home page.

### Written Assignments:

As in the previous year, there will be a number of written assignments. However, this year there will be an *important change*: (a) there will be a lot fewer of them, and (b) once you have completed about half of them you will have achieved "completion status", i.e., turning in additional homework won't improve your grade, but you may do so to practice and have your work checked for correctness.

### Programming Project:

The course project consists of five programming assignments. Four of the assignments form the components of a compiler. The project is to be done individually. Discussion with other members in the class and the TA is encouraged but each student must do the

design of the programs and the coding individually. The programs are to be written in Java. We will be using some code that has been written for the compiler project and some commercial software. **Start the programming assignments early!** Completing the course project is a large, complex, and rewarding task, which is made much easier by giving adequate forethought to design. The course schedule allows ample time to complete the assignments if started on time -- take advantage of it. The project components are assigned in roughly increasing order of size and difficulty; proportionately more time is allotted for the later assignments. Later assignments will be weighted more heavily in the final grade. Programs will be evaluated for correctness, organization, and documentation.

Documentation and structuring should be incorporated into programs from the beginning. Neither the instructor nor the teaching assistant will help with incomprehensible programs.

### **Late Policy:**

No late assignments will be accepted.

### **Examinations:**

There will be one midterm examination and a final. Both exams will be closed book exams. The date for the midterm is Monday, October 11. The date for the final exam is Wednesday, December 15. There will be no make-up or early exams given so please plan your schedule accordingly.

#### **A NOTE ABOUT PRELIMS:**

The prelim exam in programming languages will be held jointly with the final exam in this class. It will consist of the final exam questions plus some additional questions and additional time. Both the final and the prelim exam will be on the topics of the whole course, and in case of the prelim include background material on programming languages that is not specifically covered in class but on the prelim reading list.

## Grading:

Your grade will depend on your performance in the course---there is no predetermined curve. It is impossible to pass the course without doing the programming assignments. The relative weight of the components of your grade will be approximately:

Written assignments & participation	10 %
Course project (5 parts)	35 %
Project 1	3 %
Project 2	6 %
Project 3	6 %
Project 4	10 %
Project 5	10 %
Midterm	25 %
Final	30 %

**It is expected that all students understand University policies on academic honesty. Cheating on assignments or exams is very serious and will not be tolerated** in this class. Evidence of cheating will result in serious actions.

## Readings:

The material presented in class will correspond roughly but not exactly to the material covered in the readings. The assigned readings will be updated as the term progresses.

## Handouts and Lecture Notes:

In addition to being passed out in class, handouts will be available on-line on the class home page. Lecture notes will also be available on the web page by noon on the day of the class. If you want hard copies of the notes for class, you must print the copies yourself.

## Important Dates - 2004

Monday, August 30 - First day of class

Monday, October 11 - Midterm exam (tentatively)

Friday, October 29 - Last day to withdraw from course

Monday, December 13 - Last day of class

Wednesday, December 15- Final exam

---

### **Course Contents**

1. Introduction
2. The COOL programming language (used in the course project)
3. Lexical Analysis
4. Syntax Analysis
5. Syntax-directed translation
6. Type checking
7. Storage Allocation
8. Run-time execution support
9. Intermediate representations
10. Control flow analysis
11. Data flow analysis
12. Data flow lattices
13. Static single assignment form (SSA), dominators, dominance frontier
14. Dependence and alias analysis
15. Loop optimizations
16. Procedure optimization, inlining, tail-call elimination
17. Register allocation
18. Code scheduling
19. Interprocedural analysis
20. Optimizations for the memory hierarchy
21. Automatic memory management and garbage collection