

# ECE495S Introduction to Compilers & Translation Engineering

Tentative Syllabus

Fall 2008 Mo/We/Fr 12:30–1:20 PM, EE115

**Instructor** Prof. R. Eigenmann  
**Tel** 49-41741  
**Email** eigenman@purdue.edu (email may be answered in the next lecture)  
**Office** EE334C  
**Office Hours** Tu/Th 9:00–10:00, (from 8AM on email request – put “office hours” in subject line)  
**Secretary:** Jill Comer, EE339, 49-43649 (will do grade bookkeeping, contact Jill if instructor is not present for office hours)  
**Course TA** Sajjad Hossain, sajjad@purdue.edu  
Office Hours: Mo/We 2-4PM, ENAD 302F  
**Course Web Page:** <http://www.ece.purdue.edu/~eigenman/ECE495S>

**Prerequisites:** Proficiency in C language. C++ or Java experience is of advantage. ECE363 (Microprocessor Systems and Interfacing), ECE368 (Data Structures).

**Text:** Fischer and LeBlanc, *Crafting a Compiler with C*, Benjamin/Cummings, 1991, ISBN 0-8053-2166-7

**Additional References:** (1) Cooper and Torczon, *Engineering a Compiler*, Morgan Kaufmann. (2) Muchnick, *Advanced Compiler Design&Implementation*, Morgan Kaufmann, ISBN 1-55860-320-4. (3) Aho, Sethi and Ullman, *Compilers: Principles, Techniques and Tools*, 1986.

**Description:** The design and construction of compilers and other translators. Topics include compilation goals, organization of a translator, grammars and languages, symbol tables, lexical analysis, syntax analysis (parsing), error handling, intermediate and final code generation, assemblers, interpreters, and an introduction to optimization. Emphasis is on engineering, from scratch, a compiler or interpreter for a small programming language, such as a subset of C. The course includes a substantial project component, in which the students will stepwise implement such a compiler.

**Course Outcome:** After completing this course, the student will have demonstrated:

1. an understanding of the terminology, representation and use of formal languages and grammars (1, 2, a)
2. an understanding of the terminology and techniques of lexical analysis, parsing, semantic processing, code generation, and optimization (1, 3, a, e)
3. an ability to design and implement a compiler, translator or interpreter for a small language based on their knowledge of (1) and (2) (1, 3, 4, a, b, c, e, k)

In order to receive a passing grade in the course, students must show proficiency in *all* topics: (1) terminology, (2) representations, and (3) use/application of formal languages; compiler techniques (5) lexical analysis, (6) parsing, (7) semantic processing, (8) code generation, and (9) optimization; and project skills (10) compiler design from specifications and (11) problem solving.

**Course Grading:**

We will use +/- grading (A+, A, A-, B+, etc)

Tests	50%	(10% each of three midterm exams, 20% final exam)
Homework	10%	
Class participation	5%	
Projects	35%	

Regrading policy: any information that you feel will affect your grade, including grade disputes and emergencies that prevent you from attending class, must be sent by email to the instructor. In general, the instructor will not respond to these notes or change your intermediate grades. However, the information will be factored into your final class grade. Note that a regrade request may increase *or* decrease your grade. Regrade requests must be received within a week from the assignment of the grade. Requests received after the last week of classes will not affect your grade.

It is your responsibility to check with the TA or secretary before the end of the last week of classes that the grades on file agree with your records. If you don't check this information, you may receive a grade other than the one you earned.

**Student pictures:** As part of your first homework assignment you must see the TA to take a digital photo. This is for the purpose of the TAs and instructor getting to know you and properly assigning credits and grades. You will only get credit for class participation if the instructor is in possession of your picture.

**Academic Honesty:** You are expected to do all programming and homework assignments on your own or within the designated group, respectively. You may not copy files or solutions from other students/groups or from previous years' courses. Your friends may not do part of all of assignments for you.

You may discuss concepts and ideas with other students. You may answer general questions about programming language rules, help someone interpret an error message, or locate a bug. In general, if the instructor or TA would provide a particular form of assistance, you may provide it too. When in doubt, check with the instructor or TA.

You must protect your work so that others cannot copy from it. You must get informed about possible ways others may break into your computer account and you must take all reasonable measures to prevent such misuse. If your work is being copied from, you may get involved in lengthy misconduct investigations and there is a chance you are found guilty of contributing to cheating. Such investigations may delay the assignment of your final grade and your graduation.

Punishment for academic misconduct can be severe, including receiving an F in the course or even being expelled from school. All cases of cheating will be reported to the Dean of Students. Note, that the Dean of Student will make an independent determination of the students' guilt. This procedure may continue even if the instructor assigns a final grade for ECE468

Be sure to understand the Guidelines on Academic Integrity by the Dean of Students.

**Course schedule:**

45 lectures (eff. 15 weeks) plus final exam

		week		
Monday	Aug 25	Sem starts	M W F	1
	Sept 1		. W F	2
	8		M W F	3
	15		M W F	4
	22		M W F	5
	29		M W F	6
	Oct 6		M W F	7
	13		. W F	8
	20		M W F	9
	27		M W F	10
	Nov 3		M W F	11
	10		M W F	12
	17		M W F	13
	24		M . .	14
	Dec 1		M W F	15
	8		M W F	16
	15	final exams week		Final exam date TBA

**Tentative Course Outline:**

	Topic	#weeks	Reading
1.	Structure of a compiler	1	Chapters 1,2
2.	Scanning, Scanner Generators	2	Chapter 3
3.	Grammar, Parsing, Parser Generators	2	Chapters 4,5,6
4.	Semantic processing	1	Chapter 7
5.	Symbol Tables and Declarations	1	Chapters 8,10
7.	Processing Expressions, & control structures	2	Chapters 11,12
8.	Procedures and Functions	2	Chapters 13
9.	Code Optimizations	2	Chapter 15, 16, handouts
10.	Program Analysis	1	handouts
	Exams	2	

**Reading:** Reading the textbook sections corresponding to the presented lecture material is an essential part of the course.

**Homework:** Homework will be assigned in the first lecture (excluding exams) each week and is due before the first lecture of the following week. The TA will collect homework outside the classroom *before* the class starts. There will be a deduction for homework turned in late.

**Exams:** There will be three exams during regular class hours. The final exam schedule will be announced on the course web page when known. All exams are comprehensive. For midterm exams, the emphasis is on the recently learned material. All exams are “open textbook and notes.” However it is strongly recommended that you page through the textbook and notes as little as possible during the exam. Searching in your notes can slow you down significantly. Use the textbook only in “emergencies”. Exams will include a question about the syllabus.

Any conflict with the exam schedule need to be brought to the instructor's attention in the first lecture. For emergencies that prevent attendance of a scheduled exam, you must present documentation confirming exceptional reasons beyond your control. You must also contact the instructor, TA or secretary in person as soon as you become aware of the cause. Don't rely on email, as it may arrive late or not at all. The following are examples of insufficient reasons: non-emergency doctor visits, travel delays of less than a day, emergencies of persons other than immediate family members, job interviews scheduled after the first lecture.

**Projects:** In a group of two students, you will implement a compiler for a simple language. Starting with the code given in the textbook for the Micro language, you will extend the language to a "useful" mini language and add compiler passes for code generation and simple optimizations. You can choose the implementation language for your project. All project steps and grading procedures will be specified on the course web page. All project questions should be directed at the TA.

**Class Participation and Attendance:** Class attendance is necessary to follow the course. Taking class notes is essential to comprehend the material. The class participation grade will be based on interactions during the class. Good participation means you answer at least one question each lecture. This grade often makes the difference between a good and a very good grade.

If you cannot attend class for a good reason, it is your responsibility to make up the missed material. Please do not ask the instructor "I could not attend the last lecture, what did I miss?" Make sure you assign a class mate to take notes for you *before* the class you miss.