

CS423 Compiler Construction

Instructor: Dr. Stephen Blythe
Classes meet: MW 3–4:15PM, in EB1024
Office: EB3042
Office Hours: MW 10–11:15AM, TR 2–4PM
& by appointment
Office Phone: (618) 650-3344
E-mail: sblythe@siue.edu
Class WWW page: <http://www.siue.edu/~sblythe/cs423.html>

Objectives:

This course will be an introduction to the theory behind and construction of compilers. By the time you have completed this course, you should be able to:

- understand the basic components of a compiler
- understand the theory behind how compilers work
- build your own compiler

Prerequisite:

The official course prerequisite is a 'C' or better in CS330 (Programming Languages). Although not explicitly a prerequisite, you will find some concepts discussed in MATH224 (Discrete Mathematics) useful; we will review the basics of any prerequisite material as needed, and then explore those concepts more fully.

Textbook:

Engineering a Compiler

Keith D. Cooper & Linda Torczon, Morgan Kaufmann, 2004

Grading:

Grading will be broken down as follows:

Component	Percentage
Homeworks (2-3)	20%
Programming Projects (4±1)	30%
Midterm Exam	25%
Final Exam	25%

There will be no curve, and grades will be assigned as follows:

A = 90% – 100%

B = 80% – 89%

C = 70% – 79%

D = 60% – 69%

F = 0% – 59%

Although class participation is not mandatory, it will be taken into account when final grades are assigned. Thus, I *strongly* encourage you to ask questions in class.

Grading Questions:

I will address all questions about grading *if they are brought to my attention in a timely manner*. Please see me *within one week* of the return date of the exam, project, or homework in question.

Course Policies:

1. Note that the final exam is scheduled for Monday, May 1, 2006 at 10AM (running until 11:40AM). Be sure to make any end of semester travel plans accordingly, as this date/time cannot be changed! If you know now that you cannot make this date/time see me as soon as possible - requests for a “make-up” final exam are guaranteed *not* be honored if they are made after February 3, 2006.
2. Tests will be *closed* book and *closed* notes.
3. Make-up tests will *not* be given. If you know you will miss a test, you must notify me in advance so that alternative arrangements can be made.
4. All programming projects and homework assignments are due at the time and date indicated on the assignment. There will be the following early/late submission bonus/penalties:
 - Before 48 hours of due date: +5 points (bonus)
 - On Time: Full Credit (no bonus/penalty)
 - Up to 1 class late: -10 points
 - Up to 2 classes late: -25 points
 - Over 2 classes late: no credit (i.e. a zero is recorded)

Note that these bonus/penalty points do *not* apply to exams, and this policy (or portions of it) may be overridden by an explicit statement found on the handout corresponding to the assignment.

5. If you need an extension to complete an assignment, **you must request this extension before the original assignment due date**. There is no guarantee that such an extension will be granted, and my decision on such extensions will be final.
6. You will be responsible for *all* material covered in class, unless otherwise explicitly stated, whether or not it is covered in the required text.

Collaboration and Cheating:

- Collaboration is strictly prohibited during tests. Anyone caught cheating during a test will receive an **F** as their final grade for the *entire course*.
- I encourage collaboration on homeworks, however. Please feel free to work in groups to solve the homework problems. If you do work in groups on homework problems, make sure that each person in the group hands in a separate *hand written* homework assignment; one submission per group is *not* acceptable, and neither is a photocopied or computer printed group based submission.
- It is more difficult to define “cheating” on projects, which are to be worked on *individually*. It is all right to discuss the project with other people and together formulate ideas about how to attack it. It is **not** all right to copy *any* part of a program from anybody or to allow *any* part of your programs to be copied. The first violation of this policy will result in a grade of **0** for that project for *all* involved parties. Any second time offenses will result in an **F** as a final grade in the class.

Topics Covered:

1. Introduction & Review –*compilation, linkage, lexical analysis, parsing*
READING: chapter 1
2. Lexical Analysis –*finite state automata, the lex/flex/flex++ programs*
READING: chapter 2
3. Syntax Analysis, Translation, and Type Checking– *grammars, LL parsing, LR parsing, syntax trees and translation, type checking, the yacc/bison programs*
READING: chapters 3, 4
4. Code Generation –*intermediate code, target machines, DAGs*
READING: chapters 5, 6, 7
5. Code Analysis & Optimization –*basic blocks, loops, data flow analysis*
READING: chapters 8,9,10
6. Register Allocation, Instruction Selection, Instruction Scheduling –*graph coloring, approximations, tree-pattern matching, peepholing, list scheduling*
READING: chapters 8,9,10