CSCE 531 Compiler Construction Spring 2016

Time:	TuTh 11:40am - 12:55pm
Place:	WMBB Nursing Rm 409
Instructor:	Stephen Fenner
Office:	Swearingen 3A65
Office Hours:	TuTh 1:40pm - 2:40pm and 4:05pm - 4:35pm, or by appointment
Phone:	777-2596 (only during office hours; email is preferred otherwise)
E-mail:	FENNER-dot-SA in the domain gmail-dot-com
Teaching Assistant:	Xiaopeng Li
TA's Office:	Swearingen 3D41
TA's Office Hours:	TWRF 9:00am - 11:30am or by appointment
TA's Phone:	803-735-6806
TA's E-mail:	xl4 in the domain email-dot-sc-dot-edu
Course Homepage:	http://www.cse.sc.edu/~fenner/csce531/index.html
Required Text:	Aho, Lam, Sethi, & Ullman, *Compilers: Principles, Techniques, and Tools (2nd ed.)*, Addison-Wesley, 2007

Prerequisites

CSCE 240 or the equivalent (which itself requires CSCE 215 or the equivalent). Particularly, this course assumes that you have previouse experience writing substantial programs in C or C++, spanning multiple source files, on a Unix or GNU/Linux platform or the like, and that you know how to use the command-line `make` utility to coordinate software development. **This course involves extensive programming in C on Linux. This is not a course to take to learn how to program in one of these languages. This is not a course to take if you do not already know GNU/Linux fundamentals. You are assumed to know these things before you begin.**

Objectives

This course has two major objectives. First, you will learn how modern compilers are designed and implemented (and also a bit about the difficulties of designing languages). The second objective of the course is for you to improve your software engineering skills by

- working in a small team;
- working with a system involving thousands of lines of code;
- working with a system written partially by others;
- working with a system for which testing is nontrivial; and,
- learning to use program generation tools (Lex/flex and Yacc/bison).

IMPORTANT: This course (especially the project, see below) is very time-consuming and often frustrating. Ask anyone who has taken this course what a workload it is. However, if you persevere, you will learn a great deal. Please keep this in mind when scheduling your time for coursework. If you are not sure that you can devote an ample amount of time and effort to this course, I recommend that you not take it at this time.

Lecture Topics

Compiler design is an amalgam of many varied ideas and techniques, mixing theory and practice. We will concentrate mostly on front-end issues: general background (Chapter 1 of the text), lexical analysis (2.6, 3.1--3.7 in the text), syntax analysis (2.4, 4.1--4.7), syntax-directed translation (2.3, 5.1--5.5), and type checking (6.3, 6.5). We will touch upon code generation and optimization, but will not go into much detail with either. If time permits, we will go into more detail with one of the topics above, or explore possible extensions of existing techniques.

Equipment

The "official" computer development work will be on the Linux/x86 machines in the lab (l-1d39-01.cse.sc.edu through l-1d39-20.cse.sc.edu). Make sure you have an account on these machines. All your code will be tested on one of these machines.

The Project

The major part (35%) of your course grade will be based upon a large, team-based programming project---writing a compiler for a sizable subset of C++ (target machine is GNU/Linux/x86). The project will consist of the following parts, with their approximate due dates:

I	Week 6	declarations
II	Week 8	expressions and functions
III	Week 11	control constructs, pointers, and array accesses
IV	Week 13	records (extra credit only)

Each part will be graded with equal weight, but they are not necessarily of equal difficulty. Part II is likely the most difficult, Part III the easier because it is in smaller, independent chunks. Handouts describing each part in more detail will be posted on the web off of the course homepage.

You will work on the project in a team of two or three people (three is recommended). Teams will be chosen before the project starts. Teams cannot mix graduate and undergraduate students.

Programming project parts may be handed in late at a penalty of 20% per weekday up to one week late. A weekday is a 24-hour period starting at 12:00 midnight and going through 11:59 pm. Weekends (Saturday and Sunday) and vacation/snow days are free and not included in the count. A program submitted any time through 11:59 pm on a given day will be counted as having been submitted that day. For example, for a project part due on Wednesday, submitting on Friday is two days late, and submitting any time Saturday through the following Monday counts as three days late. The following Wednesday counts as five days late, and no submissions will be accepted after that.

Each project part has three possible levels of difficulty. Those students taking the course for **graduate credit** will be expected to do the highest level to receive full credit. Those taking the course for **undergraduate credit** must do the second highest level for full credit, and may optionally do the highest level for extra credit. Functionality at the lowest level is necessary for further parts of the project.

Grading of programming assignments will be automated. This means that you must adhere to the specified interfaces. We will make the scripts we will use for grading available to you for self-testing.

WARNING: Although grading is mostly or entirely automatic, your code must be adequately documented and structured. If we choose to examine your source code directly but cannot easily read and understand it, you will lose points.

Other Homework, Quizzes, and Exams

You might be given up to two pass/fail oral quizzes individually during the semester. These quizzes will be given outside of normal class time, and will be 10--15 minutes in duration. These quizzes will cover one or more of your team's prior submissions of programming assignments. You will be asked to explain and defend your code: design decisions, algorithm choices, implementation details, documentation, etc. Your grade (pass or fail) will be based upon on how well you can explain your team's programs. Your oral exam performance, if any, will be considered part of your project grade. (Failing an oral quiz may have serious consequences (see Plagiarism and Cheating Policy, below).)

There will be at least two or three other individual (non-team) homework assignments given during the semester. A homework might be a short to medium length programming assignment and/or a written assignment. The homework (in total) will be worth 15% of your course grade. These assignments must be turned in on time. No late nonproject homework assignments will be accepted.

In general, all exercises that I assign throughout the course, except drawings, are to be submitted by CSE Dropbox. Drawings may, but are not required to, be submitted by dropbox as well. Alternatively, drawings may be submitted in class.

The rest (50%) of your course grade will be divided between an open-book open-notes **midterm exam** given **Thursday, February 25 in class** and a similar **final exam** given **Thursday, April 28 from 9:00am to 11:30am**. For both exams you are allowed any printed and hand-written material, but you are *NOT* allowed electronic devices of any kind, except for legitimate use by students with disabilities registered through the university's Office of Student Disability Services.

NOTE to APOGEE students: you must be physically present to take both exams.

In summary, your course grade will be determined in the following way:

Programming Project	35%
Homework	15%
Midterm Exam (Thurs February 25 in class)	15%
Final Exam (Thurs April 28, 9:00am - 11:30am)	35%

Extra Credit Policy

The policy about distributing extra credit has changed from previous semesters. From now on, extra credit earned on the team project counts toward the team project only, up to 100% (35% of your grade). So earning extra credit in one installment can offset a poor showing in another, but only up to 100% of the project. Extra credit earned on individual homework assignments counts for both the project and the homework, but not exams. Extra credit on exams (midterm and final) counts toward everything and is unlimited.

Here is how your grade is computed: Suppose you earn p out of 100% on the project, you earn h out of 100% on the individual homework, and you earn e out of 100% on the combined exams. Any or all of these three quantities may exceed 1 = 100% because of extra credit earned. Then your total score for the course is

$$T = \min(\min(0.35p, 0.35) + 0.15h, 0.5) + 0.5e$$

T may still surpass 100%, but only because of extra credit earned on exams.

Plagiarism Policy

You are expected to read and understand the student honor code in the Carolina Community Student Handbook and Policy Guide. You are also expected to know the policies specific to this course, given below.

The following applies to individuals and to teams.

Copying someone else's work without proper citation, with or without the other person's knowledge, constitutes academic fraud. It is a serious intellectual crime, as it betrays the trust of the instructor and other students. The same is true for letting someone else copy your work. Neither will be tolerated. The punishment is *automatic failure of the course* for all parties involved (copier and copiee), entry onto the student's permanent record, as well as any other disciplinary action taken by the university. Discussion and collaboration among students *at the conceptual level only* (i.e., general ideas, approaches, and techniques to solve a problem) is tolerated, even encouraged, but the answer you give and the program code you write must be completely your own or your team's. You may not copy anyone else's work, even if you modify it afterwards. Likewise, do not under any circumstances give a copy of your homework to, or make your homework available for copying by, anyone except me or your other team members, otherwise you will be held equally responsible for any plagiarism that results. This includes acts of negligence such as failing to read-protect your files, even if you don't explicitly invite someone else to copy them. Failure of an oral exam (described above) may constitute sufficient, but not necessary, evidence of wrongdoing. Other forms of cheating (on exams for example) will also be punished by automatic failure from the course. If you have doubts or questions about what any of this means, please see me any time.

This page was last modified Tuesday January 12, 2016 at 08:41:45 EST.