USC Viterbi School of Engineering

Compiler Development

ITP 439x (3 Units)
Spring 2017

Catalogue Description	Practical applications of techniques used to develop a compiler. Topics include scanners/parsers, intermediate representations, optimization, and the back-end.
Objective	This course is designed to provide students with a fundamental understanding of how compilers are created. As the focus of most present-day compiler development is on intermediate representations, optimization, and the back-end, this course will tilt more towards these concepts, though it will touch on the major front-end concepts. Furthermore, although compilers are the primary topic of this course, it is also the intent to provide students with the extremely valuable experience of working with an existing large-scale code base (specifically LLVM). This includes the requirement to utilize some test-driven development practices.
Language	Programming assignments will be written in C++, and extensive prior experience with the language is expected.
Prerequisites	ITP 435x *or* Instructor approval
Instructor	Sanjay Madhav
Email	*Email:* madhav@usc.edu
Office Hours	M/W: 2-4:30PM in OHE 530H
Lecture	M/W: 5-6:20PM in KAP 148
Course Structure	This course features programming assignments that involve creating a working compiler for a subset of C called University Simple C (USC), which was created specifically for this course. The assignments are broken down as follows: 1. Recursive descent parsing 2. Semantics and Symbol Table 3. Generation of LLVM IR 4. Implementation of Single Static Assignment 5. Basic LLVM optimization passes 6. Register allocation In addition to these programming assignments, there are five written homework assignments. These homework assignments will serve as a prologue to the programming assignments. Thus, it is critical students finish these homework assignments prior to beginning the corresponding programming assignment. There is a midterm and a cumulative final exam.
Textbooks	*Engineering a Compiler (Second Edition)*. Cooper, Keith and Linda Torczon. ISBN-10: 012088478X.

Grading

The course is graded with the following weights:

Programming Assignments (6% each)	36%
Homework (3% each)	15%
Midterm Exam	24%
Final Exam	25%
TOTAL POSSIBLE	100%

Grading Scale

Letter grades will be assigned according to the following scale:

93%+	A
90-92%	A-
87-89%	B+
83-86%	B
80-82%	B-
77-79%	C+
73-76%	C
70-72%	C-
69	D+
67-68	D
66	D-
65 and below	F

Half percentage points will be rounded up to the next whole percentage. So for instance, 89.5% is an A-, but 89.4% is a B+.

There is no curving. Students will receive the grade they earn. Extra credit is generally not offered.

Policies

Make-up policy for exams: To make up for a missed exam, the student must provide a satisfactory reason (as determined by the instructor) along with proper documentation. Make-up exams are only allowed under extraordinary and emergency circumstances.

Late Homework: The homework will be due at the start of class on the designated due date, and will not be accepted late, barring a documented emergency.

Late Programming Assignments: Students will be provided three "slip" days for the entire semester. These slip days can be used one at a time or all at once. Upon consumption of these slip days, students will be assessed a 25% penalty per day late, for up to three additional days. After three additional days (past the slip days), the program will no longer be accepted. In the case that all slip days have been used, extensions will only be provided in the event of a documented emergency.

Software

The programming assignments will use the free and open source LLVM framework. The assignments support Windows, Mac OS X, or Linux, though Mac, if possible, is the recommended development environment. Windows requires Cygwin to build the projects, and debugging on Windows/Linux must be done through GDB. Mac, however, supports debugging directly via Xcode.

Statement on Academic Conduct and Support Systems	**Academic Conduct** Plagiarism – presenting someone else's ideas as your own, either verbatim or recast in your own words – is a serious academic offense with serious consequences. Please familiarize yourself with the discussion of plagiarism in *SCampus* in Section 11, *Behavior Violating University Standards* https://scampus.usc.edu/1100-behavior-violating-university-standards-and-appropriate-sanctions/. Other forms of academic dishonesty are equally unacceptable. See additional information in *SCampus* and university policies on scientific misconduct, http://policy.usc.edu/scientific-misconduct/. Discrimination, sexual assault, and harassment are not tolerated by the university. You are encouraged to report any incidents to the *Office of Equity and Diversity* http://equity.usc.edu/ or to the *Department of Public Safety* http://capsnet.usc.edu/department/department-public-safety/online-forms/contact-us. This is important for the safety whole USC community. Another member of the university community – such as a friend, classmate, advisor, or faculty member – can help initiate the report, or can initiate the report on behalf of another person. *The Center for Women and Men* http://www.usc.edu/student-affairs/cwm/ provides 24/7 confidential support, and the sexual assault resource center webpage sarc.usc.edu describes reporting options and other resources. **Support Systems** A number of USC's schools provide support for students who need help with scholarly writing. Check with your advisor or program staff to find out more. Students whose primary language is not English should check with the *American Language Institute* http://dornsife.usc.edu/ali, which sponsors courses and workshops specifically for international graduate students. *The Office of Disability Services and Programs* http://sait.usc.edu/academicsupport/centerprograms/dsp/home_index.html provides certification for students with disabilities and helps arrange the relevant accommodations. If an officially declared emergency makes travel to campus infeasible, *USC Emergency Information http://emergency.usc.edu/* will provide safety and other updates, including ways in which instruction will be continued by means of blackboard, teleconferencing, and other technology.
A Further Note on Plagiarism	In this class, all homework submissions will be compared with current, previous, and future students' submissions using MOSS, which is a code plagiarism identification program. If your code significantly matches another student's submission, you will be reported to SJACS with the recommended penalty of an F in the course. It is okay to discuss solutions to specific problems with other students, but it is not okay to look through another student's code. It does not matter if this code is online or from a student you know, it is cheating. Do not share your code with anyone else in this or a future section of the course, as allowing someone else to copy your code carries the same penalty as you copying the code yourself.

Course Outline			
W	**Date**	**Topic(s)**	**Reading/Due Dates**
1	1/9	Intro; Compiler Basics; Scanning	Ch. 1; Ch. 2 (§2.1-2.3)
	1/11	More Scanning; Top-down parsing	Ch. 2 (§2.1-2.3); Ch. 3 (§3.1-3.3)
2	1/16	**No class – MLK Day**	
	1/18	More Top-down parsing	Ch. 3 (§3.1-3.3); **HW1 Due in class**
3	1/23	Bottom-up parsing	Ch. 3 (§3.4);
	1/25	Semantic Analysis; Symbol Tables	Ch. 4 (§4.1-4.2); Ch. 5 (§5.5); **PA1 Due 1/27 @ 11:59PM**
4	1/30	Intermediate Representations	Ch. 5 (§5.1-5.3)
	2/1	LLVM IR	"LLVM Language Reference"; **HW2 Due in class**
5	2/6	Generating Expressions and Control Flow	Ch. 7 (§7.1-7.4; 7.8)
	2/8	Arrays, Strings, Records/Structures	Ch. 7 (§7.5-7.7); **PA2 Due 2/10 @ 11:59PM**
6	2/13	Procedures and Calling Conventions	Ch. 6 (§6.1-6.3); Ch. 7 (§7.9)
	2/15	Midterm review	
7	2/20	**No class – President's Day**	
	2/22	**Midterm exam**	
8	2/27	Object Oriented Languages	Ch. 6 (§6.4-6.6); **HW3 Due in class**
	3/1	Static single-assignment, Part I	"Simple and Efficient SSA"; **PA3 Due 3/3 @ 11:59PM**
9	3/6	Basics of Optimization	Ch. 8 (§8.1-8.4)
	3/8	Basics of Data-flow Analysis; Dominators	Ch. 9 (§9.1-9.2.1)
		Spring Break	
10	3/20	Available Expressions; Live-variable analysis	Ch. 9 (§9.2.2);
	3/22	Code Motion; LLVM Opt Passes	Ch. 10 (§10.3); "Writing LLVM Opt Passes"; **PA4 Due 3/24 @ 11:59PM**
11	3/27	Static single-assignment, Part II	Ch. 9 (§9.3.4-9.3.6)
	3/29	The Backend; Instruction Selection	Ch. 11; **HW4 Due in class**
12	4/3	Global Register Allocation	Ch. 13 (§13.1-13.4);
	4/5	More Global Register Allocation	**PA5 Due 4/7 @ 11:59PM**
13	4/10	Instruction Scheduling	Ch. 12 (§12.1-12.3);
	4/12	More Instruction Scheduling	Ch. 12 (§12.1-12.3);
14	4/17	Optimizing Compiler Case Studies	**HW5 Due in class**
	4/19	Interprocedural Analysis + Aliasing	
15	4/24	TBD	
	4/26	Conclusion; Final review	**PA6 Due 4/28 @ 11:59PM**
		Final Exam – Wednesday, May 3 @ 4:30-6:30PM	