University of South Carolina
Department of Computer Science and Engineering
College of Engineering and Computing

Compiler Construction
CSCE 531 Sections 001 and J60 Location: INNOVA 1400 (Section 001) and Asynchronous Delivery (Section J60) – Spring 2022

Instructor	Dr. Marco Valtorta
Course Websites	https://cse.sc.edu/~mgv/csce531sp22/index.html (main), https://blackboard.sc.edu (lectures), https://dropbox.cse.sc.edu (assignments)
Phone	(office) 803-777-4641; (mobile) 803-446-3225
Email	mgv@cse.sc.edu
Office location	Innovation Center (INNOVA) 2269
Meeting Times	TTh 1450-1605, INNOVA 1400.
	Office Hours: Monday 1430-1730 in INNOVA 2269. Please email mgv@cse.sc.edu in advance; if necessary, a phone or virtual face-to-face conversation will be arranged at a different time.
Teaching Assistant	Canyu Zhang, CANYU@email.sc.edu. Office hours; Tuesday 1000-1130, INNOVA 2242

NOTE: I check email very frequently and usually respond within a few hours. Email is the preferred and strongly recommended means of communication with the instructor.

Academic Bulletin Description:

CSCE 531 Compiler Construction. (Credits:3) (Prerequisites: CSCE 240) Techniques for design and implementation of compilers, including lexical analysis, parsing, syntax-directed translation, and symbol table management.

Carolina Core Learning Outcome: None.

Course Learning Outcomes

Upon successful completion of this course, students should be able to:

- Formally define the grammar and semantics of a language.
- Given an appropriate context-free grammar, design either a bottom-up or a top-down parser for the grammar.
- Use Lex- and Yacc-like tools (Alex and Happy)
- Specify an interpreter for a language.
- Specify a type checker for a language.
- Generate intermediate code for a language.
- Generate machine code for a language.
- Given the semantic definitions for an appropriate language, implement the semantic routines for a top-down compiler.
- Perform register allocation and liveness analysis.

Graduate students will also be able to demonstrate the ability to evaluate an important paper from the peer-reviewed literature on compilation and assess its contribution by writing a report and/or preparing a presentation and/or writing a software program.

Course Overview

This course will be delivered in person (Section 001) and asynchronously (section J60) through Blackboard Collaborate Ultra.

• Student-to-Instructor (S2I) Interaction: Students will listen/view lectures online and interact with the professor via Blackboard Collaborate Ultra or email. The professor will hold online office hours in his office (INNOVA 2269), post announcements on the course websites, and provide individual feedback to students through the CSE dropbox website and email.

• Students-to-Student (S2S) Interaction: Students will engage in discussions through email and the discussion forum on Blackboard (which will be set up if there is interest).

• Student-to-Content (S2C) Interaction: Students will engage with course content by completing assignments and reports and (possibly) participating in video conference meetings. The instructor will reply to all feedback in a reasonable amount of time; the same is expected of the students. Specifically,

• Communication: Responses to email communication and questions will be provided within 48 hours.

• Assignment and Test Grading Grades for assignments will be returned within one week of due date.

• All assignments are due before the beginning of class. A 10% deduction will be given to late submissions that are turned in before the beginning of the next class. No credit will be given for assignments that are turned in after the beginning of the next class.

Required Textbooks:

- Aarne Ranta. [R] *Implementing Programming Languages: An Introduction to Compilers and Interpreters*. College Publications, 2012. ISBN: 978-1-84890-064-6. (required text). Supplementary materials from the author are available. The most recent errata list is here.
- Torben Aegidius Mogensen. [M] *Introduction to Compiler Design, second edition*. Springer, 2017. ISBN: 978-3-319-66965-6. (required text). Supplementary materials from the author are available.
- We will use parts of this free online textbook by Thorben Aegidius Mogensen of the Department of Computer Science at the University of Copenhagen [M10] (local copy).

Recommended Textbooks:

- Watt, David A. and Deryck F. Brown. [W] *Programming Language Processors in Java*. Prentice-Hall, 2000 (not required; we will use parts of it). Supplementary materials from the author, including an errata list, are available.
- Graham Hutton [H]. *Programming in Haskell*. Cambridge University Press, Second Edition, 2017 (not required; we will use parts of it). Supplementary materials from the author, including an errata list, are available.
- Andrew Appel. [AJ]. *Modern Compiler Implementation in Java*, 2nd edition. Cambridge University Press, 2002. ISBN: 9780521820608. The "Tiger Books" also include an ML and a C edition.
- Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, & Tools*, 2nd edition. Addison-Wesley, 2007. ISBN: 0-321-48681-1. The "Dragon Book."

Technology and Required Course Materials:

- Students must view PowerPoint and pdf lectures/presentations.
- Students must read chapters in the required textbook.
- Students must have access to a computer with Internet access to check the course website maintained by the instructor (https://cse.sc.edu/~mgv/csce531sp21), the departmental dropbox (https://dropbox.cse.sc.edu/login), and Blackboard.
- Students must have the ability to create, save and upload programs to the departmental dropbox.
- Students must have access to a computer with the Haskell Platform (including Alex and Happy) installed, such as the departmental Linux computers. It is recommended that students install this software on their own computers.

Course Purpose and Overall Structure of the Course:

This course is an introduction to compilation. It has five parts. In the first part, we review programming language concepts and describe compilers and interpreters as tools for software development. In the second part of the course, we address syntax issues: how to specify the lexical and syntactic structure of a language using regular grammars and context-free grammars, and how to design and code lexers and parsers. In the third part, we address contextual issues: scope ant type checking and how to design and code type checkers and interpreters; the issue of symbol tables is addressed in this part. In the fourth part, we address code generation, which we divide into two parts: intermediate ("three-address") code generation and machine code generation. In the fifth part, we address register allocation, which is an important issue in many actual target computers, and (if time allows) how to deal with function calls.

This course is foundational and emphasizes theory and methods. Nevertheless, the course has a project component. The programming language to be used for the project is Haskell, and a goal of the course is to get the students to learn enough Haskell to implements parts of a simple compiler, with the aid of tools such as Alex, Happy, and BNFC, from a simple imperative language to Java Virtual Machine code. It is not expected that the students will build a complete working compiler. The details of the project will be communicated later.

Technical Support:

Blackboard Help (http://ondemand.blackboard.com/students.htm)

If you have problems with your computer or Blackboard, please contact the IT Technology Support Service Desk at 803.777.1800 (open Monday – Friday from 8:00 AM – 6:00 PM), or use one of the online support options described at https://www.sc.edu/about/offices_and_divisions/division_of_information_technology/about_us/news/2020/tech_support.php. If you have problems with the departmental dropbox, please contact the instructor. The departmental and college computers are always available online.

Late Work/Make-Up Policy:

Late work is accepted with a 10% penalty until the beginning of the class after the due date. The clock on your computer may be different than that clock on the departmental dropbox. If

the clock is different by one minute, you might be subject to the late homework penalty. Plan accordingly. I recommend that you submit your assignments well before the deadline.

Extra Credit:

Extra credit assignments will not be assigned.

Attendance Policy:

There is no penalty for missing classes. Students are very strongly encouraged to attend every class and are responsible for making up the material covered in missed classes. **I estimate that one missed class will result in at least two hours of extra work outside of class.**

Ability to Work at Your Own Pace:

The course builds upon the material in an incremental fashion. It is difficult to catch up if several classes are missed.

The course syllabus includes an accessibility statement that encourages students with disabilities to register with the Office of Student Disability services; should a student with a registered disability enroll in the course, the professor will work with the Office to make any additional accommodations appropriate to that student's needs.

Course Communications:

You are required to use your *UofSC email account* throughout this course. I will be communicating with you regarding grades and assignments. I will reply to emails within 24 hours and will provide feedback on assignments within 48 hours. When sending an email, please include a *detailed subject line*. Additionally, make sure you reference the course (CSCE 531) and section (001 or J60) and sign the email with your name. Begin these emails with a proper salutation (e.g., Dear Dr. Valtorta, Hello Dr. Valtorta, and Good evening Dr. Valtorta). Starting an email without a salutation or a simple "Hey" is not professional or appropriate. Of course, if the emails evolve into a thread with rapid exchanges, e.g., when discussing a programming issue, you may omit the salutation.

Grades Will Be Calculated as Follows.

The course grade will be based on homework, project work, a midterm exam, and a final exam. The grading policy for undergraduate students is as follows:

- Midterm Exam: 20%
- Homework assignments (including project work): 45%
- Final Exam: 35%

The final exam for graduate students will include an extra question and be graded more strictly. Graduate students are required to evaluate an important paper from the peer-reviewed literature on compilers and assess its contribution by writing a report and/or preparing a presentation and/or writing a software program. Graduate students are **required** to do the extra work, and a student may lose up to one letter grade for missing or unsatisfactory extra work. Your final grade is based on the total points you have earned over the course. Individual homework assignments are not curved, and all points for all assignments are weighted equally. The numeric scores are translated to letter grades as follows:
[90-100] = A, [87-90) = B+, [80-87) = B, [77-90) = C+ ,[70-77) = C, [67-70) = D+ [60-67) = D, [0-60) = F
However, a combined score of 60% on the midterm and the final is required to obtain a C in the course; a student who would otherwise obtain a C or better in the course may obtain at most a D+ if his or her combined score on the midterm and final is less than 60%; this percentage is computed as a weighted average of the percentages in midterm and final, ~~with the final weighing twice the midterm~~ with the final and midterm weighted according to their percentage contribution to the final score, i.e., in 20/35 ratio. Simple grading rubrics will be posted on the instructor's course website under "points per assignment." You are encouraged to review the rubric before starting an assignment.

Warning.: Please **do not plagiarize**. I normally do not use an automated system to detect plagiarism, but this may change at any time. See the section on Academic Honesty below for more.

Disability and Other Student Support Services:

Students with disabilities should contact the Student Disability Resource Center (SDRC). The contact information is below:

1705 College Street
Close-Hipp, Suite 102
Columbia, SC 29208
Phone: 803-777-6142

Fax: 803-777-6741
Email: sadrc@mailbox.sc.edu
Web:
https://sc.edu/about/offices_and_divisions/student_disability_resource_center/

These services can aid with accessibility and other issues to help those with disabilities be more successful in the course. Additionally, students with disabilities should review the information on the SDRC website and proactively communicate with the professor before or during the first week of class.

The following other academic support services and resources may help you be more successful in the course as well.

Library Services (http://www.sc.edu/study/libraries_and_collections)
Writing Center (http://artsandsciences.sc.edu/write/)
Student Technology Resources
(http://www.sc.edu/about/offices_and_divisions/division_of_information_technology/)

Academic Honesty:

Every student has a role in maintaining the academic reputation of the university. It is imperative that you refrain from engaging in plagiarism, cheating, falsifying your work and/or assisting other students in violating the Honor Code.

Plagiarism/Cheating, as defined in the code of student Academic Responsibility, will **result in a grade penalty (up to course failure)** in this course in addition to any penalty/penalties exacted by the appropriate Academic Dean and the University Honor Council to whom all offenses will be reported. The Student Conduct and Academic Integrity website (https://sc.edu/about/offices_and_divisions/student_conduct_and_academic_integrity/index.php) contains information and links to relevant policies and procedures; particularly important is policy STAF 6.25 (Academic Responsibility—The Honor Code), which, among other things, describes what constitutes plagiarism and cheating, and describes the possible sanctions. You are responsible for reading and abiding by these policies and procedures.

You must save files on a USB drive or other secure area. Many of you are familiar with github from other courses or have used other version control or code management systems (e.g., bitbucket). The programming assignments in this course are rather short and do not build on each other, so you are not required to use such a system, but doing so may be useful for creating an electronic portfolio as you start your careers. Do not save your work on a public computer or library computer as others will have access to your work. You should have a back-up of all your files in another location. Submitting someone else's work is cheating and against the Carolina Code. Cheating will result in penalties**.** All parties will also be referred to the Office of Academic Integrity for additional retribution.

It is very good to study in groups. In fact, there is evidence that group studying is a predictor of success, at least in early college mathematics courses. Some of you may enjoy studying in groups! You are therefore encouraged to discuss the material you study, but you must do your homework individually, unless an assignment is explicitly designated as a team assignment. The **minimum** grade penalty for a violation will be a zero on the work involved. In addition, an honor code violation will be subject to the sanctions described in the USC Community Handbook and Policy Guide. The following paragraph, written by Professor Duncan Buell, clarifies the distinction between "learning from a discussion" and "turning in someone else's work": If, after having participated in a group activity, you can walk away, put the books down, have lunch, and then come back afterwards to re-create from your own head the material and techniques you discussed as a group, then you can legitimately say that you have learned from the group but the work you turn in is your own.

CSCE 531 --- SPRING 2021
TIME ALLOCATION FRAMEWORK

WEEK	TOPIC	SOURCE
1 (1/11,13)	Introduction: Syntax, Semantics, and Language Processors	Ch.1 [R]; Chs.1-2 [W]; Sect. 16.7[H]
2 (1/18,20)	Introduction: Syntax, Semantics, and Language Processors	Ch.1 [R]; Ch.2 [W]; Ch.13 [M10]
3 (1/25,27)	Lexical Analysis	Ch.1 [M]; Chs. 2-3 [R]
4 (2/1.3)	Lexical Analysis	Ch.1 [M]; Chs. 2-3 [R]
5 (2/8,10)	Syntactic Analysis	Ch.2 [M]; Chs. 2-3 [R]
6 (2/15,17)	Syntactic Analysis	Ch.2 [M]; Chs. 2-3[R]
7(2/22,24)	Alex, Happy, and BNFC	Ch. 2 [R]; Manuals
8 (3/1,3)	Review (if time permits) and Midterm	
3/8,10	Spring Break	
9(3/15,17)	Scopes and Symbol Tables	Ch.3 [M]
10 (3/22,24)	Interpretation	Ch.4 [M], Ch.5 [R]; Ch.8 [W]
11 (3/29,31)	Type Checking	Ch.5 [M], Ch.4 [R]
12 (4/5,7)	Intermediate-Code Generation	Ch.6 [M]
13 (4/12,14)	Machine Code Generation	Ch.7 [M]
14 (4/19,21)	Register Allocation	Ch.8 [M]
April 28	Final Exam: April 28 at 1600	

The time allocation framework is subject to review and revision during the course.
Homework assignments will be provided during the course.
The Academic Calendar for spring 2022 is at https://www.sc.edu/about/offices_and_divisions/registrar/academic_calendars/2021-22_calendar.php. Spring break is from March 6-13 (Sunday-Sunday). The last day to withdraw without a grade of WF being recorded is March 28 (Monday). Reading day is April 26 (Tuesday). The final exam schedule is at https://www.sc.edu/about/offices_and_divisions/registrar/final_exams/final-exams-spring-2022.php.