Preliminary version of 15 August 2021

Special Information for Fall 2021

It is currently expected that the course will be taught on campus. We will follow all mandates as they develop. In any event, I intend the course to be accessible to all students, even if they are not able to come to campus for whatever reason.

Course Information

General

This course counts for 3 credits. Undergraduate students should register for CS 454. Graduate students should register for CS 554. Students outside CS should consult the instructor beforehand.

Specific outcomes of instruction

At the completion of this course students will be able to:

1. Understand the tasks and internal organization of a compiler.
2. Understand one or more scanning and parsing algorithms.
3. Understand the problems of register allocation and instruction scheduling, and one or more algorithms for these problems.
4. Understand program representation using control flow graphs.
5. Understand the program analysis problem and the dataflow framework for program analysis.

Lectures

Please consult the main UNM schedule for up-to-date information.

Instructor

Darko Stefanovic, office hours TBA

Teaching assistant

TBA

Course topics

The course covers introductory topics in compiler construction, including computer organization and architecture, operating system support, code and data layout, memory management, generation of executable code, intermediate representations, simple code optimizations, as well as the traditional topics of syntax analysis. Students will implement several components of a compiler, ranging from parsing to native code generation, with an emphasis on tools for automated translation. The course will focus on the implementation of imperative languages; special techniques for functional, object-oriented, and logic languages will be left out. The course will treat simple program analysis and code optimization techniques.

Prerequisites

Students should be familiar with computer organization (at the level of UNM CS 341) and with several high-level programming languages, so that they can appreciate the purpose and the tasks of a compiler. Students should be experienced programmers able to develop fairly large programming projects quickly, and they should be comfortable working on a team.

Course format

The course will consist of lectures, written homework assignments, and projects.

Assignments

Midterm exam, 1–3 short written homework assignments to consolidate lecture material, 3–4 programming projects.

Written homework assignments

Homework assignments will be carried out individually. Detailed submission instructions will be given with each assignment.

Projects

Each project will be an implementation of an algorithm or phase in a compiler, or an algorithm or tool used to automatically generate a phase in a compiler. Detailed input/output specifications will be provided. Students will be free to choose any implementation language(s), subject to the constraint that a standalone executable file (runnable on CS machines) must be generated and submitted as part of the solution. (Similarly, the instructor will be free to discuss implementation strategies using any implementation language(s) by way of example.)

Projects will be carried out in teams. Detailed submission instructions will be given with each project.

Textbooks

Required reading

Torben Æ. Mogensen, *Introduction to Compiler Design*, 2nd Edition, 2017, Springer, `https://doi.org/10.1007/978-3-319-66966-3`.

General reading on compilers

Keith D. Cooper and Linda Torczon, *Engineering a Compiler*, Morgan Kaufmann, 2nd Edition, 2011, ISBN-10: 9780120884780.

Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman, *Compilers: Principles, Techniques, and Tools*, Addison Wesley, 2nd Edition, 2006.

Dick Grune, Henri E. Bal, Ceriel J.H. Jacobs and Koen G. Langendoen: *Modern Compiler Design*, Springer, 2nd Edition, 2012, ISBN-13: 978-1461446989.

Grading

You are expected to attend class regularly, read the assigned reading before class, and participate in class discussion. The grade will be determined as follows: Homeworks 20%; Programming projects 60%; Midterm exam 20%. Final letter grades will be assigned as follows. For CS454:

A+	100
A	92–99
A-	91
B+	90
B	82–89
B-	81
C+	80
C	72–79
C-	71
D+	70
D	62–69
D-	61
F	0–60

For CS554:

A+	100
A	92–99
A-	91
B+	90
B	82–89
B-	81
C+	80
C	72–79
F	0–71

Homework and programming assignment hand-in policy

Written homework assignments are due on the date assigned, no extensions will be granted, and no credit will be given for late homework. Late programming project submissions will be penalized $2n^2\%$, where n is the number of days late.

Lecture Plan (tentative)

- Week 1: Course organization. Translation and interpretation. Code generation. Register allocation.
- Week 2: Lexical analysis. Regular expressions. Finite automata. Theory, pragmatic issues, tools.
- Week 3-4: Syntax analysis. Parsing algorithms, top-down, bottom-up, pragmatics, tools.
- Week 5: Names, scope, and binding. Types. Semantic elaboration. Attribute grammars.
- Week 6: Intermediate representations.
- Week 7: Representing data types. Representing control flow.
- Week 8: Representing the procedure abstraction.
- Week 9: Code generation. Instruction scheduling. Register allocation.
- Week 10-14: Code optimization. Program analysis.

Communication

At the beginning of the course we will announce the tools to be used for communication and collaboration.

UNM statement of compliance with ADA

Every instructor should include an official statement in their course syllabus. The suggested syllabus statement should include the following text:

"In accordance with University Policy 2310 and the Americans with Disabilities Act (ADA), academic accommodations may be made for any student who notifies the instructor

of the need for an accommodation. It is imperative that you take the initiative to bring such needs to the instructor's attention, as I am not legally permitted to inquire. Students who may require assistance in emergency evacuations should contact the instructor as to the most appropriate procedures to follow. Contact Accessibility Resource Center at 277-3506 for additional information.

If you need an accommodation based on how course requirement[s] interact with the impact of a disability, you should contact me to arrange an appointment as soon as possible. At the appointment we can discuss the course format and requirements, anticipate the need for adjustments and explore potential accommodations. I rely on the Disability Services Office for assistance in developing strategies and verifying accommodation needs. If you have not previously contacted them I encourage you to do so."