*Version of 8 August 2018*

# Course Information

A project-centered class covering various topics in programming language implementation.

## Course structure for Fall 2018

Graduate students: register for CRN 63257. Undergraduate students: the course is also listed as CS 491; register for CRN 63272.

Course covers intermediate representations, program transformations, and code generation in the compilation of functional programming languages. At the end of the course, students will understand how higher-order functions can be efficiently implemented and how programming languages can be defined, implemented, and validated via abstract machines.

Format: Readings; code review; discussion; occasional lectures; discussion; presentations of classical papers in the literature.

## Prerequisites

CS341 (or equivalent) and any one of CS357, CS558, CS556, or equivalent experience, mainly programming in a functional language; willingness to work in a team.

Assumed background:

- functional programming in general, and Scheme, ML, or Haskell in particular
- understanding recursive data types, recursive functions to compute over them, and structural induction to prove things about them
- some familiarity with computer organization and architecture, operating systems, machine language and assembly language programming, and the C programming language

## Lectures

Tuesdays and Thursdays 2-3:15 in FEC3100.

## Instructor

Darko Stefanovic, office FEC2020, phone 2776561, email darko — office hours Mondays 11:00-12:00, Tuesday 3:15-4:00, Thursdays 3:15-4:00

### Teaching assistant

None

### Grading

You are expected to attend class regularly, read the assigned reading before class, give occasional oral presentations on research papers and team projects, and participate in class discussion. Your grade will be determined as follows:

- Programming projects, including reports: 50%
- Oral presentations: 30%
- Discussion and participation: 20%

### Programming assignment hand-in policy

Programming assignments are to be submitted on-line. Detailed instructions will be provided with each assignment.

### Textbooks

All reading materials will be provided electronically, free of charge.

## List of topics

- Introduction to compilation
- The structure of compilers
- Front-end design
- Back-end design
- Common issues in the compilation of functional languages
- Types and type checking
- Abstract machines for lambda calculi
- Representing and analyzing control flow
- Call-by-value, call-by-name, and call-by-need
- Higher-order functions and their implementation

- Parametric polymorphism and its implementation
- Program analyses and optimizations
- Validation of programs and compilers

## 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 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."