Preliminary version of 13 August 2024

Course Information

Lectures

Lecture day/time: Tuesdays and Thursdays, 12:30–1:45
Lecture location: CENT-1028

Instructor

Darko Stefanovic
Email: `darko@cs.unm.edu`
Office hours: Tuesdays and Thursdays, 2:30-3:30
Office hours location: Farris Engineering Center 2020

Teaching assistant

None

Course topics and format

The course explores the theory used to describe and define programming languages and to guide their implementation. Our approach is type-based, in the spirit of our textbook, Pierce's *Types and Programming Languages* (TAPL). As a prelude, the course offers a very brief overview of functional programming techniques and of programming language features found in the purely functional programming language Haskell.

The course is intended for first-year graduate students, but advanced undergraduates are welcome as well. Students outside CS should consult the instructor beforehand.

No specific courses are prerequisites, but programming experience and mathematical maturity are necessary. Experience with functional programming (at the level of UNM CS357) and discrete mathematics is recommended.

The course will provide students with the background they need for CS550.

The course consists of lectures, homework assignments, two *in-class* mid-term examinations, and a final examination covering the entire course. Homework assignments will primarily consist of programming executable implementations of the theoretical concepts studied, to aid in the understanding of the concepts.

Course objectives

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

1. *Construct* computer programs in a functional programming language to *solve* various application problems.
2. *Apply* compiler algorithms to symbolic input data processing in various application domains.
3. *Evaluate* a static and a dynamic semantics of a programming language with respect to type soundness properties.
4. *Design* and *program* a definitional interpreter for a programming language.

Textbook

Benjamin C. Pierce, *Types and Programming Languages*, MIT Press, 2002, ISBN-10: 0262162091.

Other useful books

Graham Hutton, *Programming in Haskell*, 2nd Ed., Cambridge University Press, 2016, ISBN-13: 978-1316626221.

Grading

You are expected to attend class regularly, read the assigned reading before class, complete homework assignments, and participate in class discussion. Your grade will be determined as follows:

- First midterm exam (8 October, in class, 75 minutes) 30%
- Second midterm exam (5 November, in class, 75 minutes) 30%
- Final exam (during the scheduled finals' week slot, 120 minutes) 40%

The purpose of exams is to test one's knowledge of the material at a specific moment in time. Therefore, unofficial requests for accommodations or delays will not be granted except in circumstances that make sitting the exam in person on the exam date impossible (e.g., prearranged conference travel or a serious medical emergency). Requests for a delay for medical reasons must be accompanied by a valid medical note presented at the time of the initial request. In such cases, students will be required to sit a catchup exam as soon as possible after the original exam date. All exams must be taken in person and cannot be taken remotely.

No requests regarding grading, such as grade mode changes, will be considered after the final class period. There will be no extra credit assignments or do-overs for exams.

Communication

Canvas will be used for administrative announcements. Lecture notes and homework assignments will be uploaded to the Canvas page for the class.

List of topics

- Topics in functional programming
  - functional programming and Haskell introduction
  - prelude types and classes
  - functions and list comprehensions; unit testing; literate programming; interactive programs
  - recursive and higher-order functions
  - declaring types and classes
  - lists in depth: map, filter, and their algebraic laws
  - lists in depth: foldr, scanr, and their algebraic laws
  - trees with folds, binary heap trees, rose trees
  - efficiency: accumulating parameters, tupling, fusion and deforestation
  - modules and abstract data types
  - lazy evaluation and infinite data structures; approximation ordering; cyclic structures; streams
  - monads
  - metaprogramming in a functional programming language
- Topics in programming languages
  - syntax
  - operational semantics
  - lambda calculus syntax and reduction
  - programming in the lambda calculus
  - combinators and combinator reduction
  - types
  - simply typed lambda calculus
  - simple extensions (ascription; let-bindings; records)
  - simple extensions (variants; recursion)
  - references
  - exceptions
  - subtyping
  - recursive types
  - type reconstruction
  - unification
  - universal polymorphism
  - program transformations

Credit-hour statement

This is a three credit-hour course. Class meets for two 75-minute sessions of direct instruction for fifteen weeks during the Fall 2024 semester. Students are expected to complete a minimum of six hours of out-of-class work (or homework, study, assignment completion, and class preparation) each week.

Academic integrity statement

Each student is expected to maintain the highest standards of honesty and integrity in academic and professional matters. The University reserves the right to take disciplinary action, up to and including dismissal, against any student who is found guilty of academic dishonesty or otherwise fails to meet these standards. Any student judged to have engaged in academic dishonesty in course work may receive a reduced or failing grade for the work in question and/or for the entire course.
Academic dishonesty includes, but is not limited to, dishonesty in quizzes, tests, or assignments; claiming credit for work not done or done by others; hindering the academic work of other students; misrepresenting academic or professional qualifications within or without the University; and nondisclosure or misrepresentation in filling out applications or other University records.

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."