CSCI 499: Advanced Programming Paradigms

Units:

Term—Day—Time:

<u>**IMPORTANT**</u>:
The general expectation for a standard format course offered in a standard 15-week term is that the number of 50-minute contact hours per week should equal the number of semester units indicated and that one semester unit entails 1 hour of class time and 2 hours of outside work (3 hours total) per week. Standard fall and spring sessions (001) require a final summative experience during the University scheduled final exam day and time.

Please refer to the *Contact Hours Reference* to see guidelines for courses that do not follow a standard format and/or a standard term.

Location: Physical address and/or course-related URLs, etc.

Instructor: Mukund Raghothaman
Office: SAL 308
Contact Info:
Email: raghotha@usc.edu
Webpage: https://r-mukund.github.io/

Course Description

This course will introduce the student to a range of advanced programming paradigms. We will assume an elementary knowledge of programming, such as that covered in CSCI 103 and CSCI 104, and study powerful ways of structuring code with higher-order functions, of providing strong guarantees with static type systems, and ways to liberate the programmer from low-level resource management. By blurring the distinction between programs and data, and widening the gap between a program and its execution, our goal is to blow the student's mind about what it means to program a machine, and to reinforce their developing sense of computational thinking. The first half of the course can alternatively be seen as an introduction to functional programming with Ocaml, while the second half of the course can be regarded as an introduction to logic programming.

Learning Objectives

We assume that the incoming student has already taken at least two courses with a heavy emphasis on practical programming. Specifically, we expect that the student is comfortable programming in an imperative language such as C, C++, or Java, is developing a mental model of computation, and has been exposed to simple analysis of asymptotic (“*big-oh*”) complexity. At the end of this course, the student will be able to:

1. Describe the model of computation and associated conceptual ideas---recursion, closures, higher-order functions, static types, relations, and fixpoints---of functional and logic programming languages.
2. Write moderately complex programs in the Ocaml programming language.
3. Define (recursive) data types of their own using ADTs, and express computations using pattern matching.
4. Use higher-order functions to perform iterative computations on lists.
5. Parse simple proto-languages using tools such as `ocamllex` and `ocamlyacc`.
6. Be able to infer and check the well-typedness of simple expressions.
7. Query relational data using languages such as SQL and Prolog.

Prerequisites

1. Required: CSCI 103, CSCI 104, CSCI 170 (or equivalent)
2. Strongly recommended: CSCI 201

Readings

The first half of the course will follow the Real World Ocaml textbook. This is the only required textbook for this course. We will assign additional supplementary readings as appropriate.

1. Yaron Minsky, Anil Madhavapeddy, and Jason Hickey. *Real World Ocaml*. 2nd edition.
   We will be using drafts of the second edition of the book which is currently in preparation and freely available at http://dev.realworldocaml.org/.
2. Harold Abelson, Gerald Jay Sussman, and Julie Sussman. *Structure and Interpretation of Computer Programs*. 2nd edition. The MIT Press, 1996.
   The book is freely available at https://mitpress.mit.edu/sites/default/files/sicp/full-text/book/book.html.

3. Leon Stirling and Ehud Shapiro. *The Art of Prolog*. 2nd edition. The MIT Press, 1994.

Related Courses

This course mirrors advanced courses on programming methodology offered at many universities. It is most closely modeled on CS 3110 from Cornell and CSE 341 from the University of Washington. Other similarly placed courses include CS 61A from Berkeley, CS 173 from Brown, and Compsci 220 from UMass Amherst.

Description and Assessment of Assignments

The course will consist of four homework assignments, an in-class midterm and a final exam. Each homework assignment will consist of a mix of short practical programming problems and theoretical questions, as appropriate. The final grade will be based on a weighted combination of the assignments and exams as follows:

1. Homework assignments (4 × 15%): 60%
2. Midterm exam: 20%
3. Final exam: 20%

Course Schedule

Unit 1: Functional Programming in Ocaml

Week 1	• The REPL as a calculator 1. Calculations with arithmetic, Booleans, strings and lists 2. Conditionals, variable bindings and shadowing
Week 2	• An introduction to types 1. Structuring data with pairs, tuples, variants, and records 2. Pattern matching 3. Recursive definitions and algebraic data types
Week 3	• Abstracting computations with functions 1. The arrow type of functions 2. Recursive definitions 3. Execution model: Evaluation order and scoping rules 4. Higher order functions: Arguments, return values, and anonymous functions
Week 4	• Processing recursive data 1. Processing lists with map and fold 2. Translating iteration to recursion • Dynamic guarantees with assertions, contracts, and property-based testing
Week 5	• Mutable state • Polymorphism, modules, and programming in the large
Week 6	• An introduction to the lambda calculus* 1. Church encodings of data as functions 2. The Curry-Howard isomorphism

Unit 2: Implementing a Language Interpreter

Week 7	• Syntax 1 1. Describing syntax with regular languages and context-free grammars 2. Lexical analysis with finite state automata (DFAs and NFAs)
Week 8	• Syntax 2 1. The CYK algorithm for parsing context free grammars 2. Bottom-up parsing using LALR grammars
Week 9	• Analyzing types 1. Type checking 2. Type inference
Week 10	• The runtime 1. Closures and tail-call optimization 2. Translating recursion to iteration with explicit stacks 3. Elementary garbage collection

Unit 3: Programming with Relations

Week 11	• Spreadsheets 1. The computational model: Cells, values, formulas, and dependence graphs 2. Pivot tables, array formulas, and lookups 3. Turing-completeness of spreadsheets as a programming medium
Week 12	• The relational data model 1. Relational algebra: SPJ queries and set operations 2. Non-recursive queries with SQL 3. Querying graph data with Cypher
Week 13	• An introduction to recursive query languages 1. Rule-based queries using Datalog 2. Bottom-up query evaluation
Week 14	• Logic programming with Prolog • Top-down query evaluation by backtracking
Week 15	• The problem of negation 1. The closed world assumption 2. Stratified queries 3. Negation as failure • Conclusion and review 1. Why study programming languages? 2. Reflections on the future of programming

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 Part B, Section 11, "Behavior Violating University Standards" policy.usc.edu/scampus-part-b. Other forms of academic dishonesty are equally unacceptable. See additional information in SCampus and university policies on scientific misconduct, policy.usc.edu/scientific-misconduct.

Support Systems:

Counseling and Mental Health - (213) 740-9355 – 24/7 on call
studenthealth.usc.edu/counseling
Free and confidential mental health treatment for students, including short-term psychotherapy, group counseling, stress fitness workshops, and crisis intervention.

National Suicide Prevention Lifeline - 1 (800) 273-8255 – 24/7 on call
suicidepreventionlifeline.org
Free and confidential emotional support to people in suicidal crisis or emotional distress 24 hours a day, 7 days a week.

Relationship and Sexual Violence Prevention and Services (RSVP) - (213) 740-9355(WELL), press "0" after hours – 24/7 on call
studenthealth.usc.edu/sexual-assault
Free and confidential therapy services, workshops, and training for situations related to gender-based harm.

Office of Equity and Diversity (OED)- (213) 740-5086 | Title IX – (213) 821-8298
equity.usc.edu, titleix.usc.edu
Information about how to get help or help someone affected by harassment or discrimination, rights of protected classes, reporting options, and additional resources for students, faculty, staff, visitors, and applicants. The university prohibits discrimination or harassment based on the following *protected characteristics*: race, color, national origin, ancestry, religion, sex, gender, gender identity, gender expression, sexual orientation, age, physical disability, medical condition, mental disability, marital status, pregnancy, veteran status, genetic information, and any other characteristic which may be specified in applicable laws and governmental regulations. The university also prohibits sexual assault, non-consensual sexual contact, sexual misconduct, intimate partner violence, stalking, malicious dissuasion, retaliation, and violation of interim measures.

Reporting Incidents of Bias or Harassment - (213) 740-5086 or (213) 821-8298
usc-advocate.symplicity.com/care_report
Avenue to report incidents of bias, hate crimes, and microaggressions to the Office of Equity and Diversity |Title IX for appropriate investigation, supportive measures, and response.

The Office of Disability Services and Programs - (213) 740-0776
dsp.usc.edu
Support and accommodations for students with disabilities. Services include assistance in providing readers/notetakers/interpreters, special accommodations for test taking needs, assistance with architectural barriers, assistive technology, and support for individual needs.

USC Support and Advocacy - (213) 821-4710
uscsa.usc.edu

Assists students and families in resolving complex personal, financial, and academic issues adversely affecting their success as a student.

Diversity at USC - (213) 740-2101
diversity.usc.edu
Information on events, programs and training, the Provost's Diversity and Inclusion Council, Diversity Liaisons for each academic school, chronology, participation, and various resources for students.

USC Emergency - UPC: (213) 740-4321, HSC: (323) 442-1000 – 24/7 on call
dps.usc.edu, emergency.usc.edu
Emergency assistance and avenue to report a crime. Latest updates regarding safety, including ways in which instruction will be continued if an officially declared emergency makes travel to campus infeasible.

USC Department of Public Safety - UPC: (213) 740-6000, HSC: (323) 442-120 – 24/7 on call
dps.usc.edu
Non-emergency assistance or information.