Course Syllabus

Jump to Today ✎ Edit

Course

Number: 785
Section: 002
Title: Special Topics: Advanced Programing Languages

Course (learning) outcomes

How to view programs and whole languages as formal, mathematical objects
How to make and prove rigorous claims about them
Detailed study of a range of basic language features
Deep intuitions about key language properties such as type safety
Powerful tools for language design, description, and analysis

Structure

The course will consist of:

• Lectures, which include unannounced quizzes

• Assignments (OCaml programming)

• Project

• Midterm

• Final

Instructor

Iulian Neamtiu, Professor, https://web.njit.edu/~ineamtiu/ (https://web.njit.edu/~ineamtiu/)
ineamtiu@njit.edu (mailto:ineamtiu@njit.edu)
Office hours (GITC 4417)
Tuesdays, 11:30am--12:50pm
Thursdays, 11:30am--12:50pm

Materials

Textbook: **Types and Programming Languages by Benjamin Pierce (https://www.cis.upenn.edu/~bcpierce/tapl/)**

For the OCaml part, we'll use **Introduction to Objective Caml by Jason Hickey (http://courses.cms.caltech.edu/cs134/cs134b/book.pdf)**

Exams

Structure: long-form questions. Exams are closed-book, closed-notes.

Midterm: in-class, Tuesday March 5.

Final: TBD
non-cumulative (i.e., only material from after the midterm is on the final).

Make-up policy: no make-up. If you miss the midterm, your final will be comprehensive. No make-up for the final.

Required CS background

While prerequisite-free, I expect students to be:
- comfortable with proofs by induction
- willing to learn OCaml
- curious about the machinery/math behind basic concepts in programming languages
- comfortable choosing and running an open source tool, e.g., from GitHub.

Students who are competent programmers and familiar with topics such as compilers and program analysis will find the material and assignments more accessible. For the assignments and project, please note that there will be no programming help.

Lectures

Will be mostly on the board, rather than slides-based. Lectures include unanncouced quizzes.

Class participation and quizzes

Participation and quizzes are 15% of your grade. *Mere class attendance does not count*: aside from attending, to receive credit you need to either:

- actively engage in class, e.g., by answering and asking questions, or
- submit insightful questions; these questions will be featured in the discussion.
Participation credit will be awarded in three increments, basically after every 8 lectures.

Project

Running an open source program analysis tool of your choice (still subject to professor approval). The project will be due in three stages, and a 3-page write-up will be the final project report. The project is individual. Project deadlines:

Tool&topic sent to professor for approval:
First try: Feb. 16 at 5pm
Second try, if first topic was declined: Feb. 23 at 5pm

- 4% for intermediate stage A (1-page write-up) due March 22
- 4% for intermediate stage B (2-page write-up) due April 5
- 8% for the final submission (3-page write-up) due April 26. Write-up: 10-point font, single space.
- 4% for the in-class presentation (April 25 lecture)

The topic/tool cannot be changed once confirmed. Not having a confirmed project topic by the second try deadline is an automatic 0 (zero) on the project.

Assignments

Small OCaml programming tasks. The assignments are individual.

Grading policy

Grading: raw score x = a weighted average of:

Class participation and quizzes: 15%
Assignments: 10%
Project: 20%
Midterm: 25%
Final: 30%

Assuming **x** is your raw score, your grade will be:

x < 63 : F
63 ≤ ***x*** < 73 : C
73 ≤ ***x*** < 77 : C+
77 ≤ ***x*** < 83 : B
83 ≤ ***x*** < 87 : B+
87 ≤ ***x*** : A

I do not curve.

Statement on academic integrity:

- ***"Academic Integrity is the cornerstone of higher education and is central to the ideals of this course and the university. Cheating is strictly prohibited and devalues the degree that you are working on. As a member of the NJIT community, it is your responsibility to protect your educational investment by knowing and following the academic code of integrity policy that is found at: http://www5.njit.edu/policies/sites/policies/files/academic-integrity-code.pdf (http://www5.njit.edu/policies/sites/policies/files/academic-integrity-code.pdf) .***

  Please note that it is my professional obligation and responsibility to report any academic misconduct to the Dean of Students Office. ***Any student found in violation of the code by cheating, plagiarizing or using any online software inappropriately will result in disciplinary action. This may include a failing grade of F, and/or suspension or dismissal from the university.*** *If you have any questions about the code of Academic Integrity, please contact the Dean of Students Office at dos@njit.edu (mailto:dos@njit.edu) "*

Absence/missed work excusals

https://www.njit.edu/dos/student-excusals (https://www.njit.edu/dos/student-excusals)

Do not contact the professor. Instead, please contact the DOS who will verify the circumstances and inform the professor on your behalf.

"Students who miss class due to bereavement, medical concerns, military activity, legal obligations, or university-sponsored events must provide the Office of the Dean of Students (DOS) with official and verifiable documentation related to the absences within 14 days and complete the online **Student Absence Excuse Request Form. (https://forms.gle/8ExUeswm24M4Tkaa9)**

Students can also stop by the Office of the Dean of Students located at 255 Campus Center or email dos@njit.edu (mailto:dos@njit.edu) .

Once the absence has been verified, the DOS will communicate on your behalf to your professor(s).

[...]

The DOS will not seek absence excusals for pre-planned vacations, trips, weddings, graduations, or non-NJIT activities."

Project Suggestions

This project is specific to 785. You cannot claim, in whole or part, any research that you do with your advisor, as being part of this project. However, applying program analysis, preferably type systems, to solve a problem in your research area is in scope, as long as it is reviewed and approved by me.

My preference would be for a research project in this space, i.e., writing a new program analysis and then implementing that into a new or existing tool, to solve a PL research problem.

Two ambitious examples would be to improve aspects of

- OCaml, essentially a narrow versions of what these papers do: https://ocaml.org/papers, (https://ocaml.org/papers,) or
- Rust, https://www.rust-lang.org/ (https://www.rust-lang.org/) see this for a collection of papers on Rust https://alastairreid.github.io/RelatedWork/notes/rust-language/ (https://alastairreid.github.io/RelatedWork/notes/rust-language/)

Note: these topics might be more difficult than what you can find on your own.

- TBD

Program Analysis Tools

(https://github.com/ashishb/android-security-awesome) TBD

Tools that are already taken or do not qualify for the project

- TBD

Course Summary: