

COMP 410: Logic Programming

Fall 2020

Instructor: Kyle Dewey (kyle.dewey@csun.edu)

Course Web Page: <https://kyledewey.github.io/comp410-fall20/>

Office: JD 4419 (I will not physically be there); Zoom link for office hours posted on Canvas and available via email.

Special COVID-19 Message

The course is being run as a synchronous online course. We will never meet in person, though we will meet virtually via Zoom regularly each week at the assigned time (see SOLAR for meeting time details). Assuming you're enrolled in the course or on the waitlist, I have emailed the Zoom link to you. If somehow you do not have the Zoom link, email me at kyle.dewey@csun.edu. For exams, if you cannot be on Zoom during the scheduled exam time, let me know ahead of time for alternative arrangements.

The synchronous lectures will be recorded and made available through Canvas, in case students want to view them asynchronously. These lectures will only be accessible to CSUN students either enrolled or waitlisted for the course. However, **by enrolling in, or waitlisting this course, you consent to having any voice or webcam recorded.** That said, **I will never require you to use your webcam or speak**; only what you voluntarily send will be recorded. Questions can either be asked verbally or textually through the Zoom chat. I will verbally repeat any questions in the chat before answering them, but I will not identify who said the question. Even so, it's possible that your name will end up in the meeting recording, identifying you as a participant (Zoom sometimes unavoidably shows participant names in the recording).

Course Description (from the catalog)

Programming techniques in the logic programming language Prolog. Prenex conjunctive normal form and grammatical algorithms. Tableaux, sequenzen, resolution and other semi-decision procedures. Closures of relations, fixed point theory, control mechanisms, relationship to functional programming.

Learning Objectives

Successful students will be able to:

- Recognize problems which are well-suited to the logic programming paradigm
- Write Prolog and Mercury programs which manipulate lists, solve NP-complete problem instances, automatically generate software tests, and interpret other Prolog programs
- Understand the connection between formal logic and logic programming, and the theoretical underpinnings of logic programming

Course Background, Emphasis, and Design

Logic programming, while often understated, is a major programming paradigm. In my opinion, logic programming is well-suited to problems with one or more of the following properties:

- There are many distinct answers of interest
- Solving the problem requires trying different approaches and seeing which works
- Answers can be described easily, but it's difficult to formulate how to arrive at an answer

This course emphasizes the use of logic programming to solve problems, along with programming techniques which are unique to the logic programming paradigm. To this end, there will be a multitude of programming assignments, most of which require you to write code. This code will be written in either Prolog or Mercury, both of which are reputable logic programming languages.

Will you ever use Prolog or Mercury ever again? Honestly, probably not. However, my goal isn't to teach these languages, but rather the ideas behind these languages. Logic programming techniques can be implemented in more mainstream, non-logical languages; it's merely inconvenient opposed to impossible (similarly, object-oriented programming can be done in C, even though C predates object-oriented programming).

Textbook and Other Required Class Materials

No textbook is required. However, there are several sources which may be helpful:

- Learn Prolog Now! (<http://www.learnprolognow.org/lpnpag.php?pageid=online>); free online textbook that goes over the basics of Prolog
- The Art of Prolog (<https://mitpress.mit.edu/books/art-prolog>); very complete book on Prolog and logic programming which covers both basic and advanced topics; I have a copy if you'd like to peruse it
- The Craft of Prolog (<https://mitpress.mit.edu/books/craft-prolog>); discusses advanced topics in Prolog, including logic programming design patterns; I have a copy if you'd like to peruse it
- The Practice of Prolog (<https://mitpress.mit.edu/books/practice-prolog>); discusses real-world Prolog applications; I have a copy if you'd like to peruse it
- Logic, Programming, and Prolog (<https://www.ida.liu.se/~ulfni53/lpp/bok/bok.pdf>); free online textbook that focuses on the theoretical underpinnings of Prolog

Additionally, a computer, be it a laptop or otherwise, is required.

Grading

Your grade is based on the following components:

Assignments	30%
Midterm Exam 1	15%
Midterm Exam 2	25%
Final Exam	30%

Not all assignments will be weighted evenly, nor will you always be given the same amount of time for assignments. Exactly which assignments are assigned depends on how the class progresses. In general, assignments will be submitted through Canvas (<https://canvas.csun.edu/>). In the event that there is a problem with Canvas, you may email your assignment to me (kyle.dewey@csun.edu), though this should be considered a last resort.

Plus/minus grading is used, according to the scale below:

If your score is >=...	...you will receive...
92.5	A
89.5	A-
86.5	B+
82.5	B
79.5	B-
76.5	C+
72.5	C
69.5	C-
66.5	D+
62.5	D
59.5	D-
0	F

If you are not present for the final exam and you have not previously made alternative arrangements with me for the final exam, a grade of WU (unauthorized withdrawal) will be assigned.

Collaboration for Assignments

All students are required to submit their own individual work. For assignments (and **only** assignments), students may discuss among each other, as long as they don't digitally share code. That is, you **cannot** simply email your code to someone else. However, you **may** discuss your actual code with someone else, including merely viewing code. The only stipulation is that **if you do discuss with someone else, say so in your submission**. This is not for punitive reasons; this is only so I get a sense of who is working with who. My intention with this policy is to enable collaborative learning, as opposed to simply sharing a solution.

Plagiarism and Academic Honesty

While collaboration is allowed on assignments, you are responsible for all of your own work. You may **not** take code from online sources and submit it as your own. No discussion whatsoever is allowed during exams, except with the instructor. Any violations can result in a failing grade for the assignment, or potentially failing the course for egregious cases. A report will also be made to the Dean of Academic Affairs. Students who repeatedly violate this policy across multiple courses may be suspended or even expelled.

Attendance

In the first week of class, I will take attendance. If you miss both sessions in the first week and have not made alternative arrangements with me, you must drop the class, as per University policy (<http://catalog.csun.edu/policies/attendance-class-attendance/>). After the first week I will not take attendance, though you are strongly encouraged to attend.

Communication

You're encouraged to use Canvas discussions to ask questions, as long as the questions don't involve your specific solution. This way, anyone in the class can answer the question, and everyone can benefit from the answers. For anything else, email me directly at kyle.dewey@csun.edu.

Late Policy / Exam Scheduling

Late assignments will be accepted without penalty if prior arrangements have been made or there is some sort of legitimate emergency (at my discretion). If you must be absent from an exam, contact me ASAP to see if alternative accommodations can be made.

If an assignment is otherwise submitted late, it will be penalized according to the following scale:

If your assignment is late by <= this many days...	...it will be deducted by...
1	10%
2	30%
3	60%
4+	100%

To be clear, assignments which are submitted four or more days beyond the deadline will not receive credit. The reason for such a harsh late policy is that we will generally discuss solutions in class shortly after the deadline, and this late policy discourages people from simply pulling a solution from an in-class discussion.

Final Project Instead of Final Exam

You may opt to do a final project instead of taking the final exam. For this option, I would need a project proposal by 10/7, and ideally we should discuss before then. The project needs to be a substantial amount of work. Teams are permissible. Talk to me and see the course website for more details.

---Schedule of Topics on Next Page---

Class Schedule and List of Topics (Subject to Change)

Week	Tuesday	Thursday
1	8/25: Introduction, motivation, functional programming refresher	8/27: functional programming refresher, BNF review, abstract syntax trees, SAT, semantic tableau
2	9/1: SAT, semantic tableau	9/3: SAT, semantic tableau, Introduction to Prolog
3	9/8: Nondeterminism, backtracking, recursion	9/10: Recursion, structures
4	9/15: Recursion, structures, unification	9/17: Structures, unification, lists
5	9/22: Lists, recursion with lists	9/24: Recursion with lists, review
6	9/29: Midterm Exam 1	10/1: Midterm exam retrospective, generating tests
7	10/6: Generating tests	10/8: Generating tests
8	10/13: Generating tests, optimizing Prolog code	10/15: Prolog-based interpreters
9	10/20: Prolog-based interpreters, metainterpreters	10/22: Metainterpreters
10	10/27: Metainterpreters, Prolog control structures	10/29: Metainterpreters, Prolog control structures
11	11/3: Peano arithmetic, constraint logic programming	11/5: constraint logic programming, modes, review

Week	Tuesday	Thursday
12	11/10: Midterm Exam 2	11/12: Midterm exam retrospective, introduction to Mercury
13	11/17: Types and data structures in Mercury	11/19: Types and data structures in Mercury, higher-order procedures
14	11/24: Higher-order procedures	11/26 : Thanksgiving (no class)
15	12/1: Higher-order procedures, Logic programming in Python	12/3: Logic programming in Python, Monadic iterators (logic programming in other languages)
16	12/8: Monadic iterators (logic programming in other languages), final exam review	12/10 : No class; lectures over

Final exam: Tuesday, December 15 from 3 - 5 PM; Zoom link will be on Canvas.