

Programming Language Concepts

Welcome to PLC! I'm glad you are here. In this class we confront the fundamental question of how programming languages are created, and in the process we will implement one of our own. The key to everything will turn out to be the idea of program transformations, which will allow us to build tools and language features that were previously impossible. And we'll do all this in the context of Scheme, a language that looks and acts very differently from any you've seen thus far.

It's a perfect course for anyone who's ever felt dissatisfied with the programming they've been doing thus far and wanted to explore new directions. Hopefully this course will bend your brain—but in a pleasing way. If you get lost—I'm happy to help. If you can persevere the ideas in this course can change the way you think about code!



(fig 1) Young programmer confronts a parenthesis in its full majesty for the first time.

Key Learning Outcomes

Functional Programming

The functional approach of Scheme is different from the Object-Oriented approach you've learned elsewhere. Closures (a particular kind of function) will be our core abstraction. Functions will take functions as parameters, construct new functions and return them, or even transform them.

A Language to Specify Languages

You will learn BNF, a language for describing the syntax of programming languages. You'll write code that takes in one language and outputs another. You'll produce code that transforms to simplify language features, code that allows for type checking, and much more . . .

Implementing your first programming language

Over the course of 5 weeks you will implement your own version of Scheme, in Scheme. You'll start with parsing into an Abstract Syntax Tree. You'll implement variables, closures, recursion, etc.

Macros and Continuations

We'll utilize two particularly powerful and strange features of Scheme. Macros will allow us to expand the Scheme language itself adding whatever features we desire. Continuations will allow us to radically transform the flow of code execution and "travel back" to a preserved execution state.



(fig 2) Languages describing programs, programs transforming languages...We are fully lost in the recursive labyrinth here. Don't try to solve it as you would in Java...just play around with it. What is strange will become everyday.

Homework and Grades

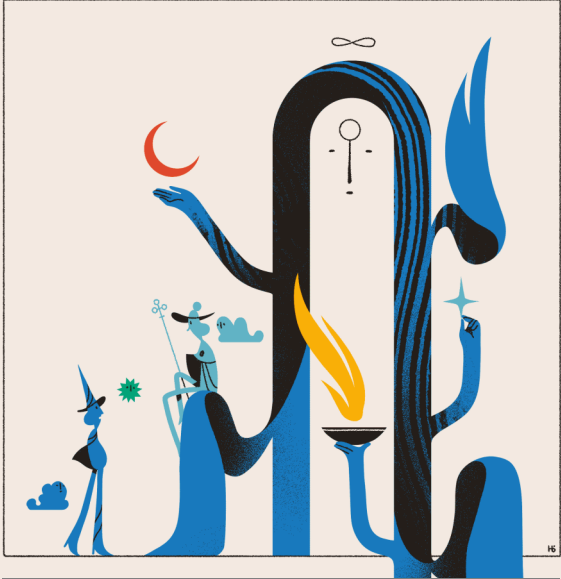
CSSE304 is meant to be a continuation in your journey of programming skill, a process you started all the way in CSSE120. As such, the heart of the course is regular programming practice. Practice is not something just for mad programmer-wizards—it’s an essential attitude that we all must strive to cultivate.

In the beginning of the class, there will be daily assignments focused mastering the Scheme language. After Exam 1, the assignments become slightly less frequent and become milestones in the Interpreter Project. After Exam 2, the course concludes with several challenging explorations of advanced topics. Broadly, your tasks become increasingly more difficult as the course progresses.

Given enough time and maybe a bit of help from myself, I believe every student can solve the homework problems. The main impediment is getting stuck in an unproductive approach—e.g. wanting so badly to be done with a particular problem that you stop exploring creatively and try to just throw a bunch of code down without thinking. Forcing you to spend hours brute-forcing an approach is the opposite what we seek to achieve. Take breaks, talk with peers, and above all work a few days ahead so you can get assistance from course staff if you get really stuck.

There are 3 2.5 hour exams in this class (Exam 1, Exam 2, Final). The problems on these exams are easier than the homework. The tests assess if you can use the techniques we’ve taught in a quick accurate way. Best way to prepare is practice—we have sample exams available that should show you exactly what to expect.

Grade cutoffs start at the usual ones (e.g. 90+ is an A, etc.) On some occasions I revise them downward (e.g. 88+ is an A) after reviewing all grades at the end of the term. Note that 65 is my usual cutoff for an F.



Homework	40%
Exam 1	15%
Exam 2	20%
Final	25%

(fig 3) Some say to pass PLC you must first pass through the the gate that is no gate, as you write the code that is not code. Others say you need to work ahead on the homework and study the problems on the practice exams. Who can know which is true?

Attendance Policy

I do think attending class has value, and I like to provide a gentle encouragement to do so. Attendance will be taken every day electronically and at the end of the term I will give overall extra credit bonus for students who pass a generous threshold. This threshold will allow for missed days for illness, interviews, sporting events etc. The bonus will be all or nothing—e.g. perfect attendance will not net you more than just a few days missed so as long as your overall attendance is good please do not stress.

You are responsible for electronically entering attendance. No matter how good the reason, I will not retroactively grant attendance (because if I do so, I will be quickly be overwhelmed with students’ attendance update requests).

Sometimes students might have special circumstances that makes full attendance impossible. If you have these, you should probably contact me as I describe in the Special Circumstances section. I resolve special circumstances by providing overall course-level rule exceptions—not by giving credit for individual missed days in the attendance policy.

Getting Help

There are variety of ways to get help in CSSE304.

- On the course Moodle are recordings from Claude's version of the class. If you found some topic confusing, definitely check out his take!
- I try to have a good number of **drop-in hours listed on the course Moodle**. You can also schedule a time with me.
- SRT are available for help with 304.

Please **cite any help you receive** as a comment in your submitted code.

Call me Buffalo!

Or Dr. Buffalo when you need help or need a favor. I can sometimes come across a bit abrupt, which is something I work on. You are not annoying me when you ask for help—even if maybe I can't help you exactly the time or way you wish. It doesn't hurt to ask!

Late Work

Because of the regular pace of assignments, getting behind in CSSE304 can be a big problem. But I will try to be as flexible as I can be. **Email me when something arises:**

- The usual extension is 24 hours or for extreme circumstances 48 hours
- We keep track of extensions per student and if you seem to be overusing the privilege we will stop allowing extensions.
- If the reason you are late is because you are having trouble completing the assignment (e.g. a bug you can't fix or just difficulty understanding) get help from an instructor. Don't assume that more time will fix the problem; plan to get help.

Special Circumstances

If circumstances have caused you to miss/fail several assignments OR if your current situation in PLC seems overwhelming, you should contact the me right away. Sometime I can find special accommodations but this is very difficult because of the tight schedule of PLC.

Academic Honesty

It is critical to maintain academic integrity. It is essential for all students to cite any and all sources of help received in completing coursework. This practice not only fosters a culture of honesty and transparency but also prevents misunderstandings that might otherwise escalate to formal proceedings. Students should also be aware of what is appropriate help on homework assignments – [see policy here](#). To ensure fairness and responsibility, any instances of suspected misconduct will be handled through the CSSE Integrity Committee.

If a case of suspected misconduct arises, it will be submitted to the CSSE Integrity Committee for review ([see policies and procedures here](#) and [possible penalties here](#)). The process includes an initial review of the evidence by the committee, a time for students to explain or admit to potential misconduct, and potentially a hearing to examine the circumstances and evidence. Students are encouraged to continue their studies and engage with the course material and instructor normally throughout the investigation.

Teaching Philosophy

I believe that developing yourself is something that happens mostly within yourself, as you attempt and reflect on challenges. I try to help you in a few ways:

- By selecting fruitful problems for you to work on, ones that I think will quickly get you to the heart of things
- By providing encouragement for you to engage with your work, in the form of grades and deadlines and holding you to those standards. Though, occasionally also, being flexible when it will help you grow.
- By helping you see other perspectives when you get stuck in a rut. Or by helping you see broader implications of what you are doing when you are focused on the practicalities.

Lecture wise, we'll focus on the strange and interesting. I will do my best to explain things in the simplest way I can, but I hope you will play with these ideas in your own head frequently. The best outcome for me is not you learning any particular part of Scheme but rather to spark an enduring curiosity for the expressiveness and strangeness of programming.



(fig 2) Generative AI has a propensity to spew plausible jibberish when it doesn't know the answer. Do not trust that it knows more than you!

Laws of Generative AI

This new crop of generative AI is a mysterious and sometimes monstrous thing. Be we shouldn't fear programming monsters, in moderation. You are allowed to use generative AI (e.g. ChatGPT Github Copilot) on **one** problem per problem set with the following laws:

- You must cite what you used in the comments in the particular question you used it
- You must focus on using the AI to understand and explore, not provide a solution that you do not understand
- You may not use generative AI on any of the Interpreter Project milestones. This problem is so classic that in this case I think generative AI simply will become a plagiarism machine.
- DO NOT use generative AI on exams
- If you are attempting any of the special, 1-point problems scattered through the homework assignments, you may use generative AI without restriction

Want more?

More policies and details are covered in the course syllabus available on the website.

Attribution

The art in this document is courtesy of the amazing Helvetica Blanc (<https://helveticablanc.com>). Licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International license.