CS 251: Programming Languages: Design and Implementation
Fall 2025

Instructor Information

Name: Anna Meyer
Office: Olin 301C
Email: annameyer@carleton.edu
For course-related communications, please DM on Campuswire instead
Office Hours: M 3:30-4:30PM, W 10:30-11:30AM, Th 9:30-10:30AM in Olin 308

Class Information

Time: 5A (MW 1:50-3:00PM, F 2:20-3:20PM)
Classroom: Hulings 316

Prerequisites: CS 201 (Data Structures)
Course materials: There are no required textbooks. Readings will be posted on Moodle.

Prefect: Cuong Tran (`tranc2`)
Prefect schedule: ✎ TBD

Course Description

In this course, you will be introduced to the study of programming languages. Much like how a course in linguistics focuses on the features of human languages (rather than learning a specific natural language like French or Chinese), this course studies concepts that underlie programming language construction. We'll learn about different design paradigms and tradeoffs.

The general outline of the course is as follows: first, we will learn Scheme, a functional language. Then, we will learn C, an imperative language. (Of course, as we learn each language we will also learn about general topics that are relevant beyond these specific languages.) The final half of the course will be devoted to writing a Scheme interpreter in C.

Course Objectives

By the end of this course, you will be able to...

- Describe the four main programming paradigms.
- Write complex programs in two new languages: Scheme and C.
- Explain the steps of interpreting a program, including tokenizing, parsing, and evaluation.
- Implement an interpreter (specifically, implement a Scheme interpreter in C).
- Identify trade-offs inherent to programming languages design, including features of memory management, parameter passing, and scope.

After completing this course, you will be better equipped to learn new programming languages in the future. You'll also be well-prepared to write high-quality and efficient code.

Class Attendance and Participation

You're expected to come to class and participate as long as you're healthy. If you're sick, please stay home! Early in the term, identify another student in the class who you will be able to ask about course materials. After missing class for illness or another personal emergency, talk with this student about what you missed and copy their notes. If you have remaining questions, post on Campuswire or attend office hours.

If your personal situation is preventing you from engaging with this class (including missing lots of class and/or not meeting deadlines beyond what is normally allowed with late tokens), you should reach out to the Dean of Students office and me.

You should come to class ready to participate. On some days, we will have in-class programming activities. If it's possible to bring your laptop to class, please do so. When we are not working on a programming activity, I request that you not use screens unless you are taking notes with a tablet/stylus setup or if you have a disability-related accommodation to take notes digitally. Research has shown that taking notes by hand leads to better material retention, plus screens can be distracting to your classmates.

Time commitment

A 6-credit Carleton generally requires 10-12 hours of work outside of class. Some students will require more time to learn the material; that is okay. However, if you find yourself spending 15+ hours a week on this class, please come talk to me! We can work on strategies to help you succeed in the class more efficiently.

Communication

Moodle is the central place for course information. It contains assignments, (links to) readings, and the daily schedule. Other platforms we use include Campuswire as a Q&A platform and Gradescope for assignment hand-in and grading. Links to these sites are available on Moodle. I will use Moodle's Announcements feature to send out announcements in between class meetings; please check your email settings to make sure these emails will reach your inbox.

If you have a question about course logistics, content, or assignments, you should post on Campuswire. If you know the answer to another student's post, feel free to reply to them. If you have a question that's personal (e.g., about grades), please send me a direct message on Campuswire instead. I will generally reply to posts and messages twice a day (morning/afternoon) during weekdays, but only sporadically on evenings and weekends.

Posts made on Campuswire should follow the same rules about academic honesty as you would use for in-person interactions: in particular, you should not publicly post your code. So, if you have a question involving homework code, consider the following: can you reframe your question to be more general? Or, can you come get help in-person at Office Hours or from a lab assistant? Alternatively, you can post your question (with code) but mark it as private. My response might be to come get help in-person though, as debugging asynchronously is often difficult!

Homework

Assignments in this course are coding-based. Some assignments are individual, while others are meant to be completed with an assigned partner. You can opt out of working with a partner, but I advise against this. In the first half of the course, homeworks are generally assigned and due each class period. In the second half of the course, homeworks are generally assigned and due every 2 class periods (these assignments are correspondingly larger).

Homework grades are based on whether you meet the project specifications that are outlined in each assignment. Each assignment is graded on a 4-point rubric: E (exemplary; 4), M (meets expectations; 3), SP (some progress; 2) or NA (not assessable; 1).

Late policy

Our late work policy aims to balance flexibility for you, structure and fairness for the class as a whole, and work/life balance for the instructor and graders.

Assignments will be due at 10pm on their due date. There will be a 1-hour buffer built into Gradescope; this is meant to account for occasional technical difficulties. Assignments will continue to be accepted for 2 days (48 hours) after their deadline. Assignments turned in during this 48-hour buffer will receive a maximum score of M *unless* you choose to use a late token on it (see below).

You get 5 late tokens to use throughout the term, and each token allows you to submit one assignment in the 48-hour buffer period without penalty. If you want to use a late token, *wait until after the assignment has been graded*. If you got a M on the assignment but would have gotten an E had the assignment not been late, submit a regrade request in Gradescope and indicate that you want to use a late token. In partnered assignments, late tokens are tracked separately (each partner who elects to use a late token will receive the grade bump).

If at any point you fall behind and the late work policies outlined here will not be sufficient for you to catch up, please reach out sooner rather than later. I want you to succeed in this course, and we will likely be able to work something out.

Exams

There will be three in-class exams, plus a final exam. The tentative in-class exam dates are October 1, October 17, and November 5. These dates may shift by up to one class period; if so, that will be communicated a week in advance. The final exam may be self-scheduled or taken during our assigned slot, Saturday November 22nd at 3:30PM.

Each exam will assess specific learning objectives. Each learning objective (except those that are new within the last two weeks of the course) will be evaluated on at least two exams. As a result, missing a single exam (other than the final) will not impact your final grade. Therefore, there are generally no make-up exam dates. (I'm willing to make exemptions on a case-by-case basis, e.g., if an exam is scheduled on a religious holiday or if the Dean of Students' office is involved in managing a prolonged series of absences.)

Revisions

Learning is not a "one and done" process and we all learn at different rates. Unfortunately, within the confines of a 10-week academic term it's not practical for myself or the graders to offer unlimited revisions on homework or exam questions. I also don't want to incentivize procrastination, because you learn best when you have time to let concepts sink in.

To try to strike a middle ground, we have the following revisions policy: If you receive a grade of 2 or 3 on an assignment for reasons that you could not have reasonably predicted before turning in the assignment, you may be invited by the graders to resubmit the assignment by a new due date (subject to the same late work rules as regular assignments). Reasons for receiving a resubmission opportunity include code style issues that we haven't discussed previously and edge cases not covered by the automated tests. If your submission fails the automated tests and/or some sections of the assignment are missing, you will not receive a resubmission opportunity.

Exams generally will not have revision opportunities because you have multiple opportunities to meet each learning objective. I may offer exam revisions on a case-by-case basis if you receive a score of 2 or 3 on a question due to ambiguity in my question or in your answer. I anticipate such opportunities to be rare, so don't purposely create ambiguity in your exam answers!

Engagement

Before most class periods, I will assign a reading and a short (2-3 question) quiz. You can reference the reading while taking the quiz, but not other materials. Please try to complete the readings and quizzes before class, as lecture and activities will assume some familiarity with the reading's topics. Class meetings will be a mix of lectures and activities. Participation in the activities (discussions, worksheets, and lab activities) is an important component of the course and your learning!

Engagement credits are a small component of your grade. Each pre-class quiz that is completed before class with a score of 100% earns 2 engagement credits, while each quiz that is completed late or with less than 100% earns 1 engagement credit. Attending class and participating in the activities earns 2 engagement credits. There may be additional opportunities to earn engagement credits throughout the term.

Grading

I will convert your scores on homework, exams, and engagement to letter grades using the following table. Specifically, I'll choose the highest row for which you meet *all* of the criteria.

Note that there are 18 homework assignments and around 100 possible engagement credits.

Grade	Homework (M or E)	Homework (E)	Exam score	Engagement
A	18	16	3.7	90
A-	17	16	3.4	<90
B+	17	13	3.0	80
B	17	10	2.7	80
B-	16	10	2.4	<80
C+	15	5	2.0	70
C	15	0	1.7	70
C-	14	0	1.3	<70
D+	10	1	1.0	60
D	10	0	0.7	60
D-	10	0	0.4	<60

Exam scores will be scaled to be between 0 and 4. (E.g., if an exam has 20 points, then you would divide your score by 5. So a raw score of 14 would correspond to $14/5 = 2.8$, which is sufficient to get a B as long as the homework and engagement requirements are met.)

I reserve the right to make this table *more generous for students*. For instance, if there are fewer opportunities to earn engagement credits than I anticipated, I may lower each engagement credit threshold by 2. I will not do the reverse: if you meet the criteria stated here, you will get (at least) that grade.

Getting help

I want you to succeed in this course! If you're stuck or struggling, there are lots of resources available. I recommend doing the following:

- Talk to your classmates. Learning is a collaborative process, and if you are struggling on something, it's likely that others are as well. (Alternatively, asking a classmate who thinks they understand a concept gives them a chance to solidify their knowledge.)
- Participate. Ask questions during class when you're curious or confused about something. Be engaged in the in-class activities.
- Go to prefect sessions. (From the Academic Support Center: "The Prefect Program offers optional collaborative learning sessions for participating classes. Prefect sessions review course concepts and often focus on critical thinking and problem-solving exercises centered on the course material.")
- Come to Office Hours. These are times I set aside for all students, and I want you to come with your questions (or general confusion, or just to chat). If you're not able to make it during the scheduled times or if you need to discuss something privately, schedule a meeting with me using the link on Moodle.
- Take breaks. Relatedly, start your homework early and try to work in small chunks rather than cramming everything in last minute. Oftentimes when you're stuck on a problem, taking 20 minutes or an hour to decompress can let you think of the solution later.
- Use other Carleton resources, including:

- CS lab assistants
- Class Deans in the Dean of Students Office are a great resource if you are dealing with personal situations or emergencies that impact your ability to succeed in your classes.
- SHAC, The Office of Health Promotion, and the Chaplain's Office can all provide support for maintaining your physical and mental health. Carleton terms are intense and challenging, and your health and well-being should always come first.
- The Academic Skills Center provides tutoring, accountability groups, and one-on-one coaching on study skills.

Academic Integrity and AI policy

The following sections outline our academic integrity and AI policies.

Guidelines for pair programming

When working on paired assignments, both partners should contribute equally and each be able to explain all parts of the submission. Additional details on the expectations for pair programming are available on the course Moodle page.

Collaboration

For **individual assignments**, you can discuss ideas with classmates, but not share code. In particular, you should be the only person who types code in the assignment, and you should not access code written by others or share your code with others. The same rules apply for **paired assignments**. You can (and should!) share code with your partner, but you should only discuss high-level concepts, not code details, with other people.

If you discuss an assignment with people other than an assigned partner, you should acknowledge this collaboration and which ideas you discussed in the `CollaborationsAndSources.txt` file that you hand in with each assignment.

Use of external resources, AI and otherwise

It is fine to use materials that I provide, including lecture notes, code from an in-class activity, or starter code from an assignment. You don't need to cite these sources.

It is also fine to use general reference resources (e.g., readings posted on the course webpage, documentation like the Scheme or C manuals, Wikipedia) to reinforce your understanding of course topics and/or search for language-specific guidance. You should cite these resources and describe how you used them in `CollaborationsAndSources.txt`. An inappropriate way to use these tools is to find an implementation (or pseudocode) of the homework solution and directly copy it, copy it with modifications, or use it as a template for your solution.

AI is where the academic integrity policy can quickly enter a gray area. For the purposes of this class, AI includes (but is not limited to) coding assistants like GitHub CoPilot, chatbots like ChatGPT or Claude, and even Google searches due to the integration of AI in the results. My core views towards using AI in coursework are as follows:

1. AI use should not reduce your intellectual engagement with the course

2. If receiving a certain type of help from a friend is prohibited, you also cannot receive that type of help from an AI

These principles lead to some concrete guidance and many, many, many gray areas. During the first week of class, we will collaboratively flesh out the AI policy in many of these gray areas, after which I will update this syllabus.

As a starting point, these are examples of some behaviors that violate this course's academic honesty policy:

- Copying code (from online, from a friend, from an AI tool, etc.) and turning it in as your own. Likewise, modifying someone else's code by changing variable names or restructuring lines is also plagiarism.
- Looking at code or pseudocode that solves the homework (from online, from a friend, from AI) and then writing your own code based on what you saw.
- Turning in code that your friend or an AI agent debugged for you (asking for help with debugging techniques is okay (e.g., "how do I use Valgrind to do XYZ") but you need to do the actual debugging work)
- Directly copying code from an assigned reading without citing it.

When constructing this syllabus, I consulted syllabi by Anna Rafferty, Dave Musicant, Anya Vostinar, and Bridger Herman. Thanks for making your materials available to me!