Syllabus

Course Code	CAS CS 320
Course Title	Concepts of Programming Languages
Semester	Spring 2025
Instructor	Nathan Mull
Teaching Fellows	Zachery Casey and Jared Pincus
Teaching Assistant	Vishesh Jain
Meeting Times	Tuesday and Thursday, 12:30PM–1:45PM (B1) Tuesday and Thursday, 2:00PM–3:15PM (A1)
Meeting Location	COM 101
Midterm Date	February 27
Grade Breakdown	30% Assignments 30% Mini-Projects 20% Midterm Exam 20% Final Exam

CAS CS 320: Concepts of Programming Languages is a course *about* programming languages, particularly the **design** and **implementation** of programming languages. In this course, we take up the programming language as an *object of formal study*. This course is *not* about how to program, though the principles we cover are generally useful for writing and reasoning about programs.

The first part of the course is on functional programming in OCaml, based on *CS 3110: Data Structures and Functional Programming* at Cornell University and its associated textbook. The topics include algebraic data types, higher-order functions, and polymorphism. It's during this part that we learn to think functionally, to view programs not as sequences of commands manipulating global state, but as compositions of functions which deconstruct and reorient data.

In the second part of the course we *implement* several interpreters for various fragments of OCaml; that is, we will write OCaml programs which *execute* OCaml programs. The topics covered include parsing, operational semantics, variable scope and binding, type checking and type inference. By the end of the course, you'll be able to execute some of the simpler programs we wrote in the first part of the course using your own interpreter.

General Information

This course meets 12:30PM–1:45PM (B1) and 2:00PM–3:15PM (A1) on Tuesdays and Thursdays in COM 101. Labs sections are held on Wednesdays; see the registar for information about meeting times and locations of labs. The coursework consists of assignments, a midterm exam, three mini-projects, and a final exam.

Prerequisites

The formal prerequisites for this course are

- ▷ CAS CS 111 & 112: Introduction to Computer Science
- ▷ CAS CS 131: Combinatoric Structures
- ▷ CAS CS 210: Computer Systems

You'll get the most out of this course (and will have the best time of it) if you've completed the 200 level CS major requirements beforehand. Experience with a high-level programming language (like Python or Java) is a must.

Learning Outcomes

- ▷ Learn the rudiments of OCaml, a functional programming language which is likely very different from other languages you've learned
- ▷ Learn to read and define a formal specification of a programming language
- ▷ Look at the theoretical basis for what makes a "good" programming language
- ▷ Implement interpreters for a collection of OCaml-like languages

CS320 is a *Creativity/Innovation* course according to the BU Hub. Programming language design is a perhaps surprisingly creative process, in which personal aesthetics play a large role in what is included/not included in a language. This is in part to say that what we cover is based on the instructor's (my) notion of what a "good" programming language is, which won't necessarily align with your own view or the views of other PL designers. This also must be understood in spite of the fact that much of what we do in the course will not feel very creative. All I can say on this is: you have to know what came first before you can change the state-of-the-art.

Course Structure

Lectures

We hold lectures each week on Tuesdays and Thursdays (see details above). During lecture, we cover the material that is presented in the reading and do some live coding examples. Lecture material is available in the course repository before the lecture meeting.

It's not guaranteed that everything we cover in lecture will be available in the associated reading. So although we will not take attendance, it is highly recommended that you come to lecture if you want to get the most out of the course.

Labs

We hold lab meetings once a week on Wednesdays (see details above). Please attend the lab meeting in which you are registered. This is an opportunity for you to engage with the material in a smaller setting with our TF/TAs. We will not take attendance, but again, its highly recommended that you attend; our TF/TAs can provide you with another invaluable perspective on the material.

Evaluation

The grade breakdown for this course is as follows.

30%	Assignments (6% each, 1 dropped)
30%	Mini-Projects (10% each part)
20%	Midterm Exam
20%	Final Exam

Assignments

There will be 6 graded assignments throughout the semester. They are made available on Thursdays and are due the following Thursday by 11:59PM. See the course schedule for more information. Most assignments are a combination of programming and written problems.

Assignments are to be submitted via **Gradescope**. If you're unfamiliar with Gradescope, see their Get Started page for more information. We will drop your lowest assignment grade so only 5 assignments will count towards your final grade in the course. This means we will **not** accept late homework assignments, under any circumstances.

Mini-Projects

In the second half of the course, we'll primarily be working on mini-projects, each of which requires you to build an interpreter. Each interpreter will be a bit more complex than the last. You should think of these as roughly the same amount of work as two programming assignments, but with a single large programming task (building an interpreter) instead of many small programming tasks. It is not possible to drop a mini-project.

Midterm Exam

The midterm exam will be held on Thursday February 27th during class. Please plan accordingly. The exam will be on the fundamentals of OCaml and typing/semantic derivations. It's meant to ensure that you are ready to build interpreters in OCaml during the second half of the semester.

Course Resources

Much of the course (particularly the first half) uses the textbook OCaml Programming: Correct + Efficient + Beautiful. All other course material will be made available on the course GitHub repository and on the course website. If you are unfamiliar with git and GitHub, see the GitHub documentation for information and tutorials. **Please check the course website and repository frequently.**

Note: I'm putting some stricter boundaries on course communication for myself this semester. I will not respond to anything (emails, Piazza posts, etc.) between the hours of 8:00PM–8:00AM.

Communication

Announcements and discussions will happen on **Piazza**. If you're unfamiliar with Piazza, see their support page for information and tutorials. Some policies regarding the use of Piazza:

- **Don't ask homework questions directly.** Formulate a question which will aid your understanding, and could potentially help others
- **Don't give homework solutions directly.** The answer-provider is just as culpable as the answer-receiver in cases of academic misconduct
- **Piazza is as useful as it is active.** We'll try to answer questions on Piazza in a timely manner, but don't hesitate to answer questions yourself

Policies

Academic Integrity

Please read BU's academic conduct code. This is taken very seriously at BU and I take it seriously in the courses I teach. If you work with others, consult materials found on the Internet, or use an AI assistant, you should cite your sources. This is a useful skill in any setting, and so I recommend being as conservative as possible regarding what you cite. In any assignment, these **citations should appear next to every corresponding problem** (in comments if the submission is code). Some examples:

- I discussed problem 1 and 2 with Leah Smith. She helped me understand X and Y aspects of the problem
- I saw the stack overflow post *stackoverflow.com/questions/6681284/python-numpy-arrays* which informed my solution
- I helped Zihan Guo with problem 4. I told them to try using X
- I asked chatGPT "what's the largest eigenvalue of this matrix?" but asked it not to show me how to solve for it

It can be tricky to determine when a citation is necessary. When in doubt, err on the side of longer, more descriptive citations. And again, **all parties are culpable**. Those who share their work with others (this includes sending files and screen sharing) receive the same attention as those who use that work. Please email me immediately if you are unsure about a particular situation.

The line between collaboration, abuse of online resources, and misconduct has become incredibly blurry, and cases are becoming more difficult to navigate. There are obvious prohibitions:

- Don't share your solutions with anyone else
- Don't use the solutions of other students
- Don't ask homework questions online on sites like Stack Overflow
- Don't ask an AI assistant for a solution to a homework question
- Don't consult a solution posted online, even to verify your own

It would be unreasonable to believe that these things never happen, or that I can recognize every case that it happens. But I have to depend on the integrity of all my students, which also means that I have to take seriously cases of academic misconduct if they do unfortunately come up. Use your best judgment and remember:

- We care about your success in this course. We provide a number of avenues to ask for help, please use them
- You will have to answer questions on exams without external aids (and in interviews when you apply for a job)
- If you don't know how to start thinking about a problem, *it's okay to ask for pointers on how to start* in office hours and Piazza. We can't give you any part of the answer, but we may be able to put you in the right direction
- We have safeguards (like dropped homework assignments) in the case you are unable to complete an assignment

Disability Statement

If you require disability accommodations, please contact me as soon as possible. You should provide me with the appropriate documentation, available though BU's Disability and Access Services. If there's a policy you see that I'm failing to comply with, please reach out with suggestions. And if you'd like accommodations that are not covered by existing services or policies, feel free to contact me and we can see what we can do. I want everyone to feel able to fully participate in the course.

Diversity Statement

My aim is to present material in a way that respects the diversity of the student body. If I fail to do this, please make me aware. Any suggestions are welcomed and appreciated. I also expect students taking any course of mine to appreciate and respect the unique opportunity they have to participate in a diverse student body.

Sexual Misconduct

Please read BU's Sexual Misconduct Policy and review the entire page for information about talking to someone confidentially about experiences of sexual misconduct, filing a report, and any other relevant information. Above all, you should feel safe, and able to be productive. If this is not the case, please reach out to me or someone else immediately.

The university is legally obligated to investigate reports of sexual misconduct, and therefore it cannot guarantee the confidentiality of a report, but it will consider a request for confidentiality and respect it to the extent possible. I am also required to report incidents of sexual misconduct and **cannot guarantee confidentiality**. I must provide our Title IX coordinator with relevant details such as the names of those involved in the incident.

A Comment on Generative AI

We're going to be talking about the role of AI assistants in education for the next decade, and the truth is that no one knows at this moment what we're going to do about it. You are the first generation that this is impacting, and not just in college, but likely when you leave as well.

This can be a good thing. You have the ability to shape your relationship with these tools (even if that means you choose not to use them, or even to fight against their use). But as this is happening, we still need to run courses at universities.

It would be drastic for me to prohibit the use of AI assistants, but they **should not be used to solve homework assignments directly**. I've seen some very powerful and creative ways that they have been used to aid learning. Some ideas:

- Ask explicitly for the solution not to be given
- Ask about high-level ideas, like analogies or mnemonics
- Ask for explanations about how code works, or why a piece of code is not working

If you have any other suggestions please let me know. The point is, it's possible to use these tools in productive ways. Remember, **they're not perfect**, they can be wrong, it can sometimes be easy to tell when a solution was produced by an AI assistant, and you won't have access to them in exams.

Student Resources

There are quite a few BU resources, it can sometimes be overwhelming. Here's a small list of the ones I think are important to know about, and I'll just add: if you're struggling in this course due to personal/health conditions, I can't guarantee I can help, but if you're comfortable reaching out, feel free to send me an email and I can see if I can point you towards the correct resources. If you're not comfortable reaching out to me, that's okay too, hopefully this list can help you find what you need. Also, keep in mind you can post anonymously on Piazza if you want to ask for help without including your name, and I can do my best to help.

- Disability and Access Services
- Student Health Services
- Health Promotion and Prevention
- Behavioral Medicine
- SARP
- Educational Resources Center
- International Students & Scholars Office