COLORADOSCHOOLOFMINES

CSCI-400: Principles of Programming Languages (PL)

Term:	Fall 2023
Hours:	3 cr
Class Meetings:	Section A: 2:00-3:15pm, Section B: 3:30-4:45pm Tue/Thu, Green Center Petroleum Hall
Instructor:	Section A: Dr. Tolga Can, Section B: Greg Davoll
Office Location:	CTLM 246K, CTLM 246D
TAs:	Alexander "Alex" Chenot, Nora Kelly, Ronan Rael, Sebastian Willowhawk, Sharfi Rahman

Contents

1   Overview and Outcomes

1.1   Course Description

Previous courses have examined how to write programs in individual languages such as Python or C++. In this class, we will take a broader view and study the key concepts and techniques that allow developers to design and implement programming languages. Ultimately, the course will improve your skill as a programmer and will deepen your understanding of how programming languages are designed and implemented.

Some students expect a "Programming Languages" course to be a survey of many different languages, but that is not the case. Learning "Programming Languages" is less about touring of the "zoo" of different languages and more about learning the "zen" that underlies all programming languages. Ultimately, learning these common and fundamental abstractions is the more valuable approach. We must all learn and use many different languages over our careers; understanding the fundamentals of programming languages will make this task easier and make us better programmers.

This course will emphasize *functional programming* for several reasons:

- Functional programming generalizes many programming constructs you have previously studied;
- Functional programming predicts the future of language development. Many programming languages that began in another style (imperative, object-oriented, etc.) are gradually gaining functional programming features over subsequent revisions;
- And functional programming helps us write correct (bug-free) programs. Functional languages offer many tools to reason about and ensure program correctness, both informally (in our heads) and formally (with an algorithm).

1.2   Learning Outcomes

Through the activities in this course, you will learn the following (Figure 1):

Part I Implement programs in the *functional* style and in a functional language, using functional programming features to support program correctness, including:

- understand the evaluation of functional programs, as based on the *lambda calculus*;
- write recursive functions, including recursive handling of structures such as lists and tree;
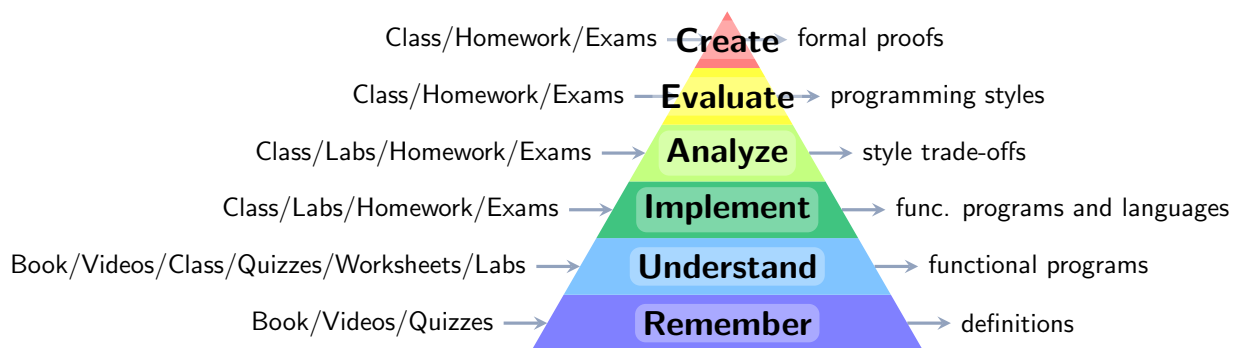

Figure 1: Bloom's Taxonomy of Learning Activities and Outcomes

- use *static typing* and *pattern matching* syntax;
- use common higher-order functions—such as `map`, `filter`, and `fold`—write new higher order functions, and use *lexical closures*;
- use and implement new *algebraic data types*;
- use and implement new *persistent data structures*.

Part II Implement the core of a programming language, including:

- use a lexer and parser generator to create a lexer and parser;
- implement expression evaluation;
- implement variables;
- implement function definition, function application, and lexical closures.

Throughout Analyze and evaluate questions about programs and programming languages, including:

- explain programming language theory terms including lexical closure, lexing, parsing, big-step semantics, small-step semantics, lexical environment, type checking, type environment, type inference, and references;
- relate common programming constructs and idioms to the *lambda calculus*, including variable definition, lexical closures, and currying;
- reason inductively about recursive functions and data structures;
- prove amortized running times for a data structure or algorithm;
- analyze, compare, and contrast the suitability of mutable/ephemeral/imperative vs. immutable/persistent/functional programming approaches and data structures for new problems.

2   General Course Information

Prerequisites   The prerequisites are CSCI-358: Discrete Math and CSCI-306: Software Engineering. If you have somehow registered without completing the prerequisites, please drop the course.

Textbooks and References   *Studying the textbook(s) is a valuable (and expected) learning activity.*

- **Primary Textbook** (main reference for the course) Clarkson et al. OCaml Programming: Correct + Efficient + Beautiful.

- **Supplemental Textbooks** (references for some advanced topics)

  - Chris Okasaki. Purely Functional Data Structures. Cambridge University Press. 1998. ISBN-13: 978-0521663502.
    Chris Okasaki.
  - Benjamin Pierce. Types and Programming Languages. 2002. ISBN-13: 978-0262162098
  - Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman. *Compilers: Principles, Techniques & Tools*. ISBN-13: 978-0321486813.

- **Programming References**

- OCaml Tutorial
- OCaml Manual
- JavaScript Specification
- Mozilla Developer Network

Office Hours

- Tolga's office hours will be in CTLM 246K.

  - Wednesday: 12:00-1:00pm
  - Thursday: 10:30-11:30am
  - Friday: 10:30-11:30am

- Greg's office hours will be in CTLM 246D.

  - Tuesday: 1:00-2:00pm

- TA Office Hours will be in the peer mentoring area or ALAMODE lab (CTLM B60).

Day	Time	TA	Location
Wednesdays and Fridays	5:00pm-6:00pm	Alex	Peer mentoring area - Area 1: Table near Confere
Mondays and Wednesdays	11:00am-12:00pm	Nora	Peer mentoring area - Area 1: Table near Confere
Wednedays and Fridays	3:00pm:5:00pm	Ronan	CTLM 224
Tuesdays	5:00-6:00pm	Sebastian	CTLM 246B
TBA	TBA	Sharfi	TBA

Online Resources

- Canvas: Written assignment submissions, Grades

- Ed Discussion: Announcements, Questions, Discussion, Homework/Project help

- MSOneDrive: Files

- Course Github Organization: Project code distribution and submission

- Isengard: ITS-managed Linux server with shell access

- MS Whiteboard: In-class discussion

- Zoom: Remote office hours

Technology Requirements   This course assumes you are able to access a GNU/Linux system (e.g., Debian, Ubuntu). If you do not run Linux on your personal workstation, you may use the ITS-managed Isengard server or install a virtual machine. The instructor does not recommend Microsoft's "WSL" due to problems encountered by past students. If you attempt to use a non-Linux platform, the instructor and TAs can provide only very limited technical support.

You will need to bring a device to class for coding, quizzes, and (electronic) discussion. On days involving programming, this device must be a laptop with Linux access. On other days, a mobile device with a web browser will be adequate to complete quizzes on Canvas and post discussion notes. Please contact the instructor if bringing a device to class would present difficulty for you.

Who should I email/contact?

- **Miscellaneous basic policy questions:** (when is the midterm? when is an assignment due?): Re-read the syllabus, check Canvas assignments, check Ed Discussion announcements, and ask any additional questions on Ed Discussion.

- **Help with assignments or course topics:** Ed Discussion, TA office hours, or instructor office hours. Private post on Ed Discussion if the matter should be hidden from other students (e.g., something about your code or questions about your grade)

- **Solutions to in-class exercises:** Slides with completed exercises will be posted to MSOneDrive after the lecture.

- **Anything sensitive or confidential:** (e.g., a health issue) Email the instructor about the issue and/or to schedule a meeting to discuss the issue.

- **Concerns/suggestions about course procedures:** Email the instructor or TAs about the issue and/or to schedule a meeting to discuss the issue.

Acknowledgments   This course draws heavily from Prof. Jed McClurg's offerings in Fall '19, '20, and '21 (thanks, Jed!). The current version of the course is inspired by Prof. Bor-Yuh Evan Chang's Programming Languages course and Cornell's course and associated textbook CS 3110: Data Structures and Functional Programming, which itself is rooted in the legendary Structure and Interpretation of Computer Programs.

3   Assessments and Grading

The course score (percentage) will be computed as a weighted average of scores (points received over points possible) as follows:

Participation	10%	(w)
Quizzes	10%	(q)
Homeworks	10%	(h)
Labs	35%	(ℓ)
Midterm Exam	15%	(m)
Final Exam	20%	(f)

$$\text{score} = .10 \left(\frac{w_{\text{recv.}}}{w_{\text{poss.}}} \right) + .10 \left(\frac{q_{\text{recv.}}}{q_{\text{poss.}}} \right) + .10 \left(\frac{h_{\text{recv.}}}{h_{\text{poss.}}} \right) + .35 \left(\frac{\ell_{\text{recv.}}}{\ell_{\text{poss.}}} \right) + .15 \left(\frac{m_{\text{recv.}}}{m_{\text{poss.}}} \right) + .20 \left(\frac{f_{\text{recv.}}}{f_{\text{poss.}}} \right)$$

Worksheets and Participation   Most lectures will have a worksheet to practice the material. Scan or photograph the worksheet (or complete electronically) and submit it on Canvas. Your participation grade will be based on making an honest effort on the exercises. Participation points may also be assigned for in-class activities.

Quizzes   Most lectures will include a quiz, graded on correctness.

Exams   There will be a midterm exam around the middle of the semester and a cumulative exam during finals week. The exams are an evaluation tool whose purpose is to produce a distribution of scores across all students that best distinguishes the extent of learning. Thus, please expect the exams to be very challenging.

Homeworks There will be several homeworks sets to practice mathematical topics covered in the course.

Labs There will be several programming projects (labs) to practice functional programming use and implementation.

Letter Grades Course letter grades will be based on a curve. It is expected—but not guaranteed—that score distributions will be normally distributed and letter grades will correspond to university and department norms. However, skewed student effort (including absenteeism) or score distributions may result in correspondingly skewed letter distributions. See Appendix B for score distributions in prior terms.

Additional Letter Grade Criteria In addition to the scoring and letter grade determination discussed above, the following criteria will be used to determine letter grades:

1. If less than 80% of participation assignments are attempted and submitted on time, the course grade will be reduced by one letter, to a minimum of D-. This requirement is to encourage timely participation and "keeping-up" with the course topics.

2. If less than 80% of of quizzes are attempted and submitted on time (irrespective of correctness), the course grade will be reduced by one letter (cumulative with participation), to a minimum of D-. This requirement is to encourage attendance and participation in the class meetings.

3. To receive a passing grade, the weighted average of the midterm and final exam score must exceed 40%. However, if exam scores are lower than expected, then this lower bar may be "curved" (e.g., to two standard deviations below the class average). This requirement is to address potential "free-riding" on project group members.

Late Policy Late work will not be accepted. Please take care to manage your time so that you are able to submit your best work by the deadline.

Fairness It is import to evaluate all students as evenly as possible. While we will attempt to accommodate disabilities and extenuating circumstances (physical/mental health, school-related travel, job requirements of self-supporting students, etc.) to the greatest possible extent, it would be unfair to offer any further special treatment.

Grading Corrections Grading changes will only be made for grading errors. It is not possible to change grades in response to disagreements about point allocation, partial credit, letter grade cutoffs, etc., because such changes would be unfair to the rest of the class. Grading corrections will only be made for the following errors:

1. *Arithmetic:* The grader incorrectly summed your points.

2. *Code:* An error in the grading environment or scripts incorrectly tested your code.

3. *Written:* The grader incorrectly understood your answer.

The deadline to request a grade correction is two weeks from the date an assignment is returned or the grade is posted.

Projects Expectations and Grading

- Code must produce the correct output to receive credit. Incorrect output, no output, compilation errors, or runtime errors will not receive credit. **Please double-check your submitted code to ensure that minor errors will not result in major test failures.**

- Code tests will include edge cases. Think through all possible conditions for your program.

- Projects will include a peer evaluation of group members. In cases where the is evidence of unequal contributions among group members—based on peer evaluations, git commit history, student interviews, etc.—the instructor may adjust project scores to better reflect individual contributions.

Written Work   Format and submit your written work as follows. Improper submission or formatting may result in a penalty on assignments.

- For FERPA compliance, all work submitted on physical paper must include a cover sheet that contains only your name and no answers or other work. Electronic submissions do not need a cover sheet.

- Write your name on *every page* of all written work. If the work cannot be matched to you, you cannot receive credit for it.

- Handwritten work must be *clearly legible* to receive credit.

- Submit electronic homeworks, reports, etc. in PDF format. Do not submit word processor files because these are inconsistently formatted by different software.

- Work must be readable when printed in black and white.

4 Tentative Schedule

Sure to change as the semester progresses. Schedule updated 2023-08-03.

4.1 Tentative Lecture Schedule

W	Date	Topic	Files	References
		Part I: Functional Programming		
1	Aug 22	00: Programming Languages Introduction	✏️	Clarkson 1
	Aug 24	01: Basics of Programming Languages	✏️📺	Clarkson 2, Pierce 5.1
2	Aug 29	02: Programming with Lambda	✏️📺	Pierce 5.2
	Aug 31	03: Linux and Git (Lab 0 Day)		Git Docs
3	Sept 5	04: Functional Programming	✏️📺	Okasaki 1–2
	Sept 7	05: OCaml and Javascript (Lab 1 Day)		OCaml tutorial & ref., JS spec, MDN
4	Sept 12	Career Day (no class)		
	Sept 14	06: Higher-Order Functions	✏️📺	Clarkson 4
5	Sept 19	Lab 2 Day		
	Sept 21	07: Algebraic Data Types	✏️📺	Clarkson 3, Pierce 11
6	Sept 26	08: Persistent Data Structures	✏️📺	Clarkson 5.6, 8.3, Okasaki 2–3
	Sept 28	Lab 3 Day		
7	Oct 3	Lab 4 Day		
	Oct 5	09: Amortized Analysis	✏️📺	Clarkson 8.2, Okasaki 5, Tarjan
8	Oct 10	Midterm Review		
	Oct 12	Midterm Exam		
		Part II.a: Language Implementation—Syntax		
9	Oct 17	Fall Break (no class)		
	Oct 19	10: Syntax	✏️📺	Clarkson 9.2, Aho 4
10	Oct 24	11: Lexing	✏️📺	Aho 3
	Oct 26	12: Parsing	✏️📺	Aho 4.4, 4.5, 4.8
		Part II.b: Language Implementation—Semantics		
11	Oct 31	Lab 5 Day		
	Nov 2	13: Induction	✏️📺	Clarkson 6.7, 6.8, Pierce 2.4, 3.1–3.3
12	Nov 7	14: Semantics	✏️📺	Clarkson 9.3, Pierce 3.4–3.5
	Nov 9	Lab 6 Day		
13	Nov 14	15: Environments and Scope	✏️📺	Clarkson 9.4, Pierce 3.4–3.5, 7
	Nov 16	Lab 7 Day		
14	Nov 21	16: Typing	✏️📺	Clarkson 9.5, Pierce 8–9, 22
	Nov 23	Thanksgiving Break (no class)		
15	Nov 28	17: Type Inference	✏️📺	Clarkson 9.6, Pierce 8–9, 22
	Nov 30	18: Mutability	✏️📺	Clarkson 7, Pierce 11.2–11.3, 13
16	Dec 5	Lab 8 Day		
	Dec 7	Classes end on Dec 6 (no class)		
17	Dec 12	Final exam: 12/12 @8am, place: TBA		
	Dec 14	Semester ends		

4.2   Tentative Lab List

- Lab 0: Git and OCaml "Hello world"

- Lab 1: Functional programming basics

- Lab 2: Higher order functions

- Lab 3: Algebraic data types

- Lab 4: Persistent data structures

- Lab 5: Lexing and Parsing

- Lab 6: Semantics and Evaluation

- Lab 7: Environments, Functions, and Lexical Closures

- Lab 8: References and state mutation

4.3   Tentative Homework List

- HW 1: Lambda Calculus

- HW 2: Types and Higher-order functions

- HW 3: Persistent data structures

- HW 4: Syntax

- HW 5: Induction

- HW 6: Type Inference

5   Policies

5.1   Flipped Classroom

We will run this course as a "flipped classroom" to provide students with the additional exposure and activities supporting course learning outcomes. In particular, repeated exposure, study, and practice supports learning the formal and mathematical topics in this course.

- **Prepare before class:**
    1. Watch the lecture video.
    2. Attempt the practice exercises (worksheets).
    3. Post lecture and exercises questions on the MS whiteboard to discuss during class.
    4. Submit the worksheet attempt on Canvas. Worksheets are graded on effort, not correctness.

- **Participate during class:**
    1. Ask questions about the video and worksheet exercises.
    2. Discuss the lecture topics.
    3. Take a quiz.

4. Participate in additional activities such as coding exercises and group proofs. **Many in-class activities will closely resemble exam questions.**

Preparation *before class and* **participation** *during class are both expected and required parts of this course.*

5.2   Mines Policies and Resources

Mines Policies and Resources

5.3   Generative AI Policy

Generative Artificial Intelligence (genAI) tools such as ChatGPT are important resources in many fields and industries. Because these tools will be used in professional and personal contexts, we believe it is valuable for you to engage critically with these tools and explore their use in generating content submitted for evaluation in this course, including worksheets, homework assignments, and lab projects. You remain responsible for all content you submit for evaluation. You may use genAI tools to help generate ideas and brainstorm. However, you should note that the material generated by these tools may be inaccurate, incomplete, biased, or otherwise problematic. We encourage you to consider how genAI complements, supplants, or fails to replace your contributions and abilities.

If you include content (e.g., ideas, text, code, images) that was generated, in whole or in part, by Generative Artificial Intelligence tools (including, but not limited to, ChatGPT and other large language models) in work submitted for evaluation in this course, you must document and credit your source. Failure to properly cite sources, including AI tools for generating content, would be considered Academic Misconduct in violation of Mines Academic Integrity/Misconduct Policy.

Additional language regarding the potential pitfalls of genAI, strategies for citation, and methods for centering the human dimension of learning can be found in this resource provided by the Trefny Center.

5.4   CS Collaboration Policies

CS Collaboration Policies

5.5   Course Policies

5.5.1   Communication

Ed Discussion is the primary communication tool used in this class and will be used for announcements, questions, and discussion. Students are expected to regularly monitor Ed Discussion and their university email (at least once a day) for announcements and changes such as modifications to class meetings. The instructor will attempt to give at least 24 hours notice (and more if possible) for such changes, but emergency situations may not allow such advance notice. If in doubt, check your email and Ed Discussion.

5.5.2   Laptop and Smartphone Policy

- Lecture slides are posted in advance. You are strongly encouraged to use your laptop or phone to follow along during lecture and to review slides during exercises.

- Note-taking on laptops, tablets, etc. is welcome if you find it useful.

- Please refrain from using laptops, phones, etc. for non-class activities, e.g., email, web browsing, games, during classtime, as it is distracting to other students.

- Some class activities (e.g., coding activities, lab days) require the use of a laptop. If you will not have access to a laptop for such activities, please contact the instructor about possible arrangements or alternatives.

5.5.3   Netiquette

Text DOs

- Ask questions and engage in conversations as often as possible—feel free to contact the instructor and TAs via the discussion forum for questions.

- When asking "tech support" questions, provide sufficient detail to diagnose and, if possible, reproduce the issue, including commands that were run, output of those commands, log files, and operating system and software versions.

- Be patient and respectful of others and their ideas and opinions they post online.

- Remember to be thoughtful and use professional language. Keep in mind that things often come across differently in written text, so review your writing before posting.

- Be prepared for some delays in response time, as "virtual" communication tends to be slower than "face-to-face" communication. Ask questions well in advance to deadlines to ensure sufficient time for a response.

- If the instructor does not respond to an important email for a few days, please send a reminder. Faculty receive a large number of emails, and sometimes messages get lost or overlooked.

- Contact the instructor if you feel that inappropriate content or behavior has occurred as part of the course.

Text DON'Ts

- Use inappropriate language—this includes, but is not limited to, the use of curse words, swearing, or language that is derogatory.

- Post inappropriate materials—for example, accidentally posting/showing a picture that is not appropriate for the course content.

- Post screenshots (images) of text output. Instead, post text as text. Compared to text, screenshots are slower to download, harder to read, and cannot be copy/pasted.

- Post in ALL CAPS, as this is perceived as shouting, and avoid abbreviations and informal language (e.g., "I'll C U L8R").

- Vent, rant, or send heated messages, even if you feel frustrated or provoked. Please instead communicate any specific concerns privately to the instructor or TAs; we want to improve the course and to accommodate any extenuating circumstances. Similarly, if you should happen to receive a heated message, do not respond to it.

- Except for course content questions on Ed Discussion, send an email or post to the entire class, unless you feel that everyone must read it.

Video DOs

- Find a quiet place to log in.

- Use headphones. Echo cancellation doesn't always work, and it is distracting to a speaker to hear their voice echoed.

- Test your microphone beforehand to ensure that the recorded audio is clear. Some builtin microphones produce speech that is difficult to understand, and it is fatiguing for listeners to try to decipher noisy audio.

- Mute your microphone when not speaking to avoid inadvertent noise that may distract others.

- Turn on your camera. Nonverbal communication is important.

- Engage in the discussion. Ask questions; ask followup questions; acknowledge responses.

- Position any light source in front of you and behind the camera to best illuminate your face.

- Use a wired network connection if possible. Wireless connections may be less reliable.

- Plug laptops or mobile devices into wall power – battery use can adversely affect video quality.

- Dress appropriately.

Video DON'Ts

- Post zoom links publicly, on social media, etc. Bad actors may join the meeting and post distracting or inappropriate material.

- Post offtopic messages in the chat. It is distracting to others.

- Share private windows such as personal email.

5.5.4   Privacy and FERPA

The university and instructor value students' rights to privacy, and this course must specifically comply with the Family Educational Rights and Privacy Act (FERPA). To support FERPA compliance, please mind the following:

- Include a cover sheet on all work submitted on physical paper. The cover sheet must have the student's name and no answers or other work. Electronic submissions do not need a cover sheet.

- Use Canvas for electronic communication containing specifics about grades. Canvas is the system chosen by the university to manage students' grades.

- Do not disclose the private information (e.g. grades) of other students.

5.5.5 Collaboration Policy

- Worksheets may be completed in groups. You are encouraged to discuss worksheet exercises with others in the class.

- Homeworks must be an individual effort. You may not copy or share solutions. However, per the CS collaboration policy above, you may consult others in the class under the "empty hands" requirement.

- Projects may be completed with your project group. Per the CS collaboration policy above, you may consult other groups in the class under the "empty hands" requirement. Copying code will be considered academic misconduct.

- Exams must be an individual effort. Copying solutions or consulting others on an exam will be considered academic misconduct.

6 FAQ

- **Q:** Is the textbook "required?"
  A: Studying the textbook is an expected learning activity for this course.

- **Q:** When/where are office hours?
  A: The instructor will post office hours on Ed Discussion the first or second week of the semester (it takes us some time to rearrange meeting schedules each semester). Instructor office hours are in the instructor's office (CTLM 246K, CTLM 246D) and/or on Zoom. TA office hours will be posted and/or on Zoom.

- **Q:** What's on the exam?
  A: Exam questions will be similar to homework assignments and in-class activities. Exams will focus on evaluating understanding, application, and synthesis of the course topics (i.e., the upper levels of Bloom's taxonomy). Questions will not focus on memorization, but one must know the key definitions and concepts to apply them. For the midterm, all topics covered up to the exam may be included. The final will be cumulative but will focus on topics covered after the midterm. The instructor will post a specific list of topics after preparing each exam, typically about a week before the exam date (in past semesters, the topic list included 80-90% of the lecture material).

- **Q:** When is the midterm?
  A: Please see the tentative schedule in this document for an approximate time. The instructor will announce firm details about the midterm closer to the date and will post the details on Ed Discussion.

- **Q:** When/where is the final exam?
  A: The registrar schedules all final exams. Please see the registrar's website.

- **Q:** The syllabus says that Linux is the supported platform for course projects, but can I use macOS?
  A: MacOS does provide some unix-like features, but it also does some things differently. If you encounter a mac-specific problem, the instructor and TAs won't be able help. In other words, it might work just fine, but if it doesn't, you'll have to figure it out yourself.

- **Q:** Which Linux distribution and version is best?
  A: Debian and Ubuntu (which is based on Debian) are good choices for distributions and have

easy and robust package managers. Debian "Stable" and Ubuntu "Long-term Support (LTS)" versions will have the most thorough testing and least likelihood of encountering problems.

- **Q:** Debug my code for me.
  A: The instructor and TAs are here to help you with projects but typically cannot do the job of debugging for you. Plus, learning how to debug your own code is an absolutely necessary skill.

- **Q:** What's my grade?
  A: The exact answer is unknowable until the end of the semester. For an approximate answer, see section 3 and compare your scores to the class distribution, which will typically be posted on Ed Discussion for major assignments. Historical score and grade statistics are listed in Appendix B.

- **Q:** Can I have an extension on an assignment?
  A: In case of extenuating circumansances (medical issue, personal emergency, etc.), of course; please contact the instructor/TA. In exceptional cases, it may be appropriate to extend a deadline for the entire class, but such extensions may also be unfair to students who completed work by the original deadline. (see "Fairness" in section 3). If you think there is reason to extend a deadline for the class, please make the request well in advance of the deadline.

- **Q:** How can I improve my grade?
  A: Participate in lecture, come to office hours, study, ask questions, and start assignments early. Score changes after-the-fact are not possible, i.e., do not ask about grade improvements after the semester has ended but instead prepare throughout the semester so you can best achieve the course's learning outcomes. See also: "Fairness" and "Grading Corrections" in section 3

- **Q:** Why are exam scores so "low"? (Half the class "failed!")
  A: This course does not use an 90/80/70-percent scale. Such a scale is (1) arbitrary and (2) poorly-aligned with open-ended and challenging nature of upper-level and graduate courses such as this one. In particular, rubrics (see Appendix A) for problem solving and proofs (both of which are a focus in this course) do not align well with a 90/80/70 scale, so lower scores do not necessarily indicate "failure." Rather, this course is graded on a curve based on the statistical distribution of course scores AND observed student effort.

- **Q:** Why does this course grade on curve?
  A: While there are arguments for and against curved grading, certain factors in this course support grading on a curve. Overall, curving supports *robust* determination of letter grades that are *fair* and *consistent*. Specifically:

  - The open-ended and challenging nature of assessments in an upper-level course results in a wider distribution of scores than low-level courses that evaluate more limited outcomes (i.e., the lower levels of Bloom's taxonomy). Curving accommodates this wider distribution to produce grades that reflect learning outcomes.

  - Average scores change slightly over different terms, e.g., based on variations in difficulty of exam questions. Curved grading ensures that letter grades remain consistent.

  - Many instructors employ ad-hoc curving if letter grade distributions don't match their intent. Instead, the systematic curved grading used in this course determines letter grades based on score statistics, eliminating ad-hoc decisions about what, when, and how to curve and thus providing better consistency and fairness in the final letter grades.

- **Q:** Why does this course use. . .

- **Q:** ...Microsoft OneDrive?
  A: MSOneDrive is the file storage system that ITS has chosen. RClone supports MSOneDrive, and the result is adequately usable.

- **Q:** ...Git and Github?
  A: In previous years, when students submitted tarballs on Canvas, they often struggled to share code with each other, and groups occasionally submitted incorrect versions of their project (resulting in much lower scores than the group expected!). Git (and Github) are critical tools to collaborate on code and to reduce the chance of submitting an unintended version. Moreover, Git is pervasive in professional software development.

- **Q:** ...OCaml
  A: OCaml is a functional programming that is common in industry and offers a syntax similar to the textbook Lambda Calculus (the mathematical foundation of programming). Thus, OCaml aligns well with the *theory & practice* philosophy of Mines.

- **Q:** ...Linux?
  A: Primarily, the programming tools used in the course are best supported on Linux. Secondarily, the instructor is unable to provide support for non-Linux systems (limited support for unix-like systems such as Mac OSX may be possible). Additionally, Linux proficiency—though not explicitly a learning outcome of this course—is vital for computing professionals, given the pervasive use of Linux in mobile devices, cloud computing, high performance computing, robotics, etc.

- **Q:** How does the instructor prefer to be addressed?
  A: Preferred: *FIRST-NAME*, {Dr., Prof., Mr.} *LAST-NAME*

A   Proof Rubric

This course includes proofs. We will use the following rubric to grade proofs.

Logic and Reasoning: 50%

- 0%: Totally wrong, all steps in the argument are illogical.
- 10%: At least one step is valid, but the argument is mostly wrong.
- 20%: Partially valid argument, but significant errors in the steps or conclusions.
- 30%: Overall argument is generally valid, but major errors present.
- 40%: Conclusion is valid, but minor errors present.
- 50%: All steps are correct and the argument is valid.

Communication – Terminology and Notation: 20%

- 0%: Major errors in terminology indicating a lack of understanding.
- 10%: Minor errors in terminology or imprecise/informal language.
- 20%: Correct, formal, and precise use of terminology.

Communication – Structure: 20%

- 0%: Difficult or impossible to follow.
- 10%: Follows basic structure for the type of proof.

- 20%: Flows well, suitable concise, and easy to read.

Communication – Grammar: 10%

- 0%: Grammar problems impede understanding.
- 10%: Sufficiently correct grammar. No issues that affect understanding.

B   Historical Score Distributions

Below are score distributions from prior terms. Letter grades are based on the Course Score, which is the weighted average as described in section 3 (weights may vary between terms).

Spring 2023
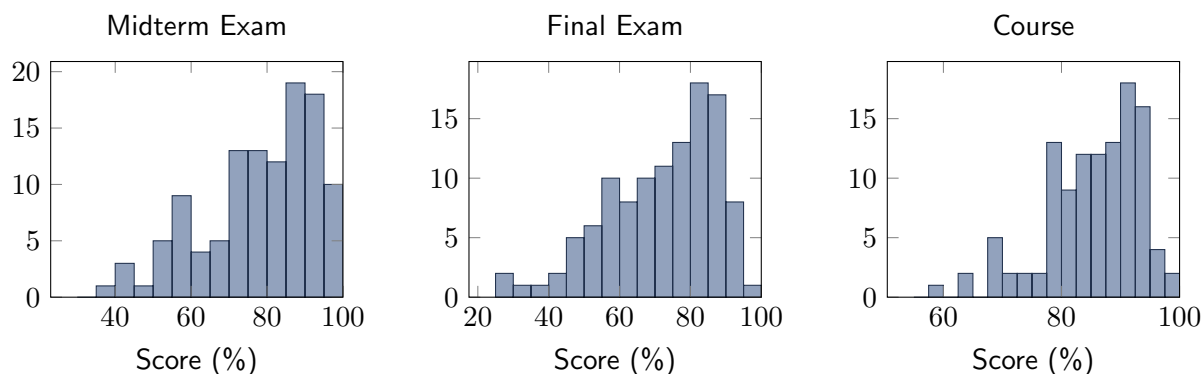
	Midterm	Final	Course
Mean	78.4	72.0	85.1
Median	82.0	76.1	86.8
Std. Dev.	14.1	15.3	8.2

A	[88.0, 100]	51
B	[80.0, 88.0)	32
C	[70.0, 80.0)	17
D	[60.0, 70.0)	11
F	[0, 70.0)	4
W	N/A	3
	Total	118

Spring 2022

Note: all sections returned to fully in-person classes, and students had access to pre-recorded lecture videos, with additional discussion and activities during class meetings. However, due to continuing COVID, many students missed class meetings for isolation or quarantine.
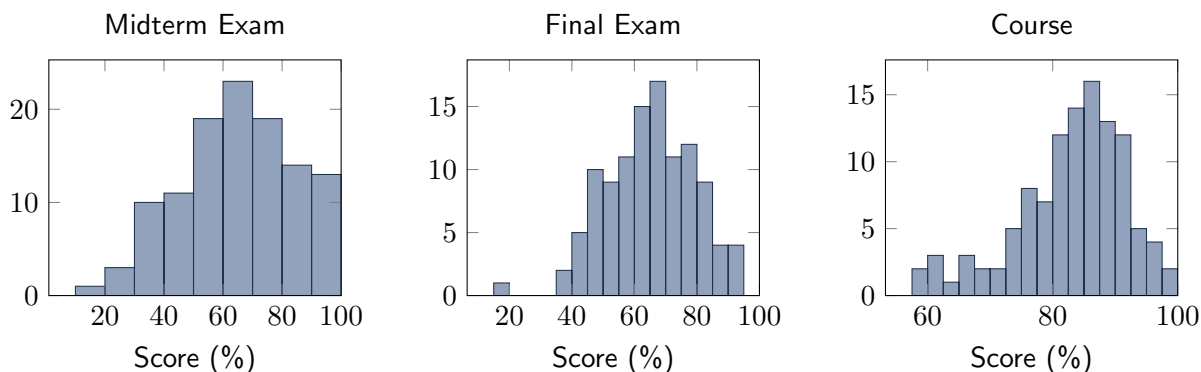
	Midterm	Final	Course
Mean	65.4	65.6	82.9
Median	66	66.0	84.1
Std. Dev.	19.0	14.2	8.9

A	[86.0, 100]	45
B	[78.0, 86.0)	33
C	[70.0, 78.0)	22
D	[63.0, 70.0)	9
F	[0, 70.0)	8
W	N/A	2
	Total	119

Spring 2021

Note: due to continuing COVID, one section of the course with 28 students was taught face to face and one section with 52 students was taught remotely. Both sections had access to pre-recorded videos, with discussion during class meetings.

	Midterm	Final	Course
Mean	59.5	53.2	78.1
Median	61	50.75	80.5
Std. Dev.	18.5	22.2	10.1

A	[85.0, 100]
B	[76.5, 85.0)
C	[66.0, 76.5)
D	[55.0, 66.0)

Spring 2020

Note: the university switched to emergency remote learning partway through the Spring 2020 semester. Letter grades in this course were all increased by roughly half a letter to compensate for the increased difficulty of this emergency switch.

	Midterm	Final	Course
Mean	51	70.5	74.1
Median	53	76	76.5
Std. Dev.	18.5	23.6	14.6

A	[80.0, 100]
B	[70.0, 80.0)
C	[60.0, 70.0)
D	[54.0, 60.0)