CS1200/CSCI E-120 Fall 2024:
Introduction to Algorithms and their Limitations

Syllabus

Basics

This course is offered for both Harvard College, as CS1200 (formerly CS120), and the Harvard Extension School, as CSCI E-120. Anything specific to CS1200 is highlighted red. Anything specific to CSCI E-120 is highlighted blue.

Course website (including past years' materials): https://harvard-cs-1200.github.io/cs1200/

Shopping/registration information:

- Watch the course preview video.
- ~~Send any questions to cs1200-staff-fall2024@g.harvard.edu~~
- ~~Salil's office hours: drop-in OH to be listed at https://salil.seas.harvard.edu/~~
- ~~Anurag's office hours: send an email if you would like to chat - I will be generally available between 8 - 10 AM EST.~~
- ~~Maxwell's office hours: send an email if you want to chat - I'm free on most weekdays at 4-6PM EST.~~

Lecture times: TuTh 9:45-11am Eastern, Science & Engineering Complex (SEC), 150 Western Ave., Allston, Room 1.321. In-person attendance at lectures is required. Lectures are recorded and can be (re)watched asynchronously, except the "Sender-Receiver Exercises": see below. Lectures are also live-streamed on Zoom. .

Course Staff

Instructors:
Anurag Anshu (he/him)
anuraganshu@fas.harvard.edu
Salil Vadhan (he/they)
salil_vadhan@harvard.edu

Head Teaching Fellows:
Maxwell Zen (he/him)
maxwellzen@college.harvard.edu
Roy Rinberg (he/him)
royrinberg@g.harvard.edu

Patel Fellow (1:1 tutoring for Harvard College students):
Joyce Lu (she/her)
joyce_lu@college.harvard.edu

Other Teaching Fellows:
Ashley Gong
Bridget Ma
Conan Lu
Elizabeth Zhong
Emily Gao
Eric Hwang
Gili Rusak
Gabe LeBlanc
Jackson Moody
Jayden Personnat
Katherine McPhie
Liam Norman
Michael Moorman
Natalie Abreu
Prin Pulkes
Rhea Acharya
Richard Zhu
Teresa Lu-Romeo

Faculty Coordinators: Emma Sachi Duncan, eduncan@seas.harvard.edu, Allison Choat, achoat@seas.harvard.edu

Whenever possible, please post questions for the course staff on Ed (privately if needed) rather than emailing us, so that we can all see the question and responses.

Overview

Looking at the world around us, we see computers solving problems on incredibly large scales: finding webpages relevant to our internet searches and returning them in sorted order, computing the quickest way to reach a destination given current traffic conditions, matching people on dating sites. How is this possible? More computing power? Intensive application-specific engineering? While these certainly have had a role to play, in CS1200, you will see and learn how to use general algorithm design principles that cut across application domains and remain relevant even as computing technology changes.

First among these principles is mathematical abstraction, whereby we capture the essence of a computational problem (as well as the notion of a "computer") so that we can develop and analyze solutions independent of an implementation. Given these mathematical abstractions, we can apply a toolkit of basic algorithmic techniques in the search for solutions, and then gain certainty in their correctness and efficiency through rigorous mathematical proofs. Furthermore, the powerful concept of reductions will allow us to identify relationships between computational problems that seem very different on the surface, and thus automatically transfer solutions from one to another.

At the same time, some important computational problems have defied the search for algorithmic solutions. Computer scientists would love to have debugging tools that determine whether their programs can crash, natural scientists would love to have simulators that quickly determine the energy-minimizing states of physical or biological systems, and university registrars would love to be able to automatically schedule classes in a way that optimally maximizes the use of the best classrooms. Why have no scalable algorithms been found for these problems?

In the last part of CS1200, we will see that many important computational problems are inherently unsolvable—they have no general algorithmic solution whatsoever! Others are solvable, but have no efficient algorithm—the minimum computation time inherently grows exponentially with the size of the problem instance. Uncovering these phenomena (known as "uncomputability" and "intractability", respectively) are unique benefits of a mathematically rigorous approach to algorithms. While we may sometimes be satisfied with empirical demonstrations of the performance of an algorithm we have found, a proof seems to be the only way to convince ourselves that there is no algorithm whatsoever.

To summarize, CS1200 aims to give you the power of using mathematical abstraction and rigorous proof to understand computation. Thus equipped, you should be able to design and use algorithms that apply to a wide variety of computational problems, with confidence about their correctness and efficiency, as well as recognize when a problem may have no algorithmic solution. At the same time, we hope you will gain an appreciation for the beautiful mathematical theory of computation that is independent of (indeed, predates) the technology on which it is implemented.

Learning Outcomes

By the end of the course, we hope that you will all have the following skills:

- To mathematically abstract computational problems and models of computation
- To design and implement algorithms using a toolkit of algorithmic techniques
- To recognize and formalize inherent limitations of computation
- To rigorously analyze algorithms and their limitations via mathematical proof
- To appreciate the technology-independent mathematical theory of computation as an intellectual endeavor as well as its relationship with the practice of computing
- To engage effectively in a collaborative theoretical computer science learning community, supporting your peers' learning as well as your own
- To clearly communicate mathematical proofs about computation to peers, conveying both high-level intuition and formal details.

Prerequisites

Experience with proofs and discrete mathematics at the level of Computer Science 20, and (Python) programming at the level of CS32/CS50. If you haven't taken these courses, we recommend using the CS20 Placement Self-Assessment and Problem Set 0 (PS0) to gauge your preparation. You can see PS0 from past years on the course website; this year's will be posted before the start of the semester. Note that while PS0 does not require any new material from CS1200, it is a full-fledged problem set that may require substantial effort to complete. The course staff will hold review sessions and office hours and review sessions during the first week of classes to support you in doing PS0.

Curricular Context

CS1200 is meant to provide a gentler entry into theoretical computer science than CS1210 (limits & models of computation) and CS1240 (algorithms). CS1200 can be taken before CS1210 or CS1240, but these courses will *not* assume CS1200 as a prerequisite. See this FAQ for a more detailed comparison between CS 1200, 1210, and 1240, and the CS Advising site for how CS1200 fits into the CS concentration requirements.

Expectations and Format

This is the fourth offering of CS1200, and the course is still somewhat experimental. By joining the class, you will be partners in the continued development and improvement of the course. We will solicit feedback from you periodically and expect to make adjustments based on it, so everything below is subject to change.

The main planned components of the course are as follows:

Main class meetings (TuTh 9:45-11am): About a third of our class meetings will begin with an approx. 30min *Sender-Receiver Exercise (SRE)* done in pairs, where the partners, the "sender" and "receiver," engage in an interactive dialogue to develop a joint understanding of a mathematical proof (related to recent class content) that the sender has studied beforehand. In the last 5min of each SRE, each of you will fill out a short reflection survey on takeaways from the SRE and ways you can improve them in the future. Students in the extension school are expected to complete each of these exercises with a partner over Zoom in the weekend before the corresponding lecture. At times to be determined before any SRE, staff are available in the "Study Lounge" in Canvas->Zoom to assist, but pairs of partners can do the SRE at other times (including in the "Study Lounge", which is always open and can also be used for other remote collaboration, e.g. on problem sets).

The remainder of our class time will consist of interactive lectures on new material.

Regular attendance in class is required. The SREs will develop important skills such as deconstructing and communicating proofs. Additionally, your peers will be depending on your participation and these will be a part of participation grading (described below).

Sections (weekly, times TBD): Each week, a section handout consisting of practice problems and review of difficult concepts will be provided. For students in the college class, TF-led sections will be held weekly, at times and places TBD to accommodate as many students as possible. Attendance is optional. For students who don't go to one of those sections, we recommend working through the section materials independently. You will find sections most useful if you have read and started thinking about the problem sets and the corresponding material in advance.

Office hours (every week, times TBD): The teaching staff will have regular office hours, both in-person and on Zoom, to help guide you as you work through the material. Before coming to office hours with questions about the problem set (other than clarifications on what is being asked), it is important to invest real effort in trying to solve the problems without staff assistance. If you get stuck, the teaching staff will help you get unstuck, not by giving hints, but by probing your thought process and understanding of the content and/or reviewing material and related

problems. The goal is to help you build the skills that will enable you to discover solutions for yourself.

Problem sets: There will be approximately 10 weekly problem sets, typically due Wednesdays at 11:59pm. Some of the problems may require deep thought, so you are strongly encouraged to begin them early and to brainstorm in groups of 2-3 students. (See Collaboration Policy below.) Most problem sets will include a qualitative question for you to reflect on how you are engaging with the course (see Participation below) and/or on what you are learning in the course.

Asynchronous discussions: We will use the *Ed* platform for discussions, Q&A, and announcements.

Participation: The last three learning objectives listed above ("To appreciate…", "To engage effectively…", "To clearly communicate…") go beyond the technical content of cs1200 and involve the development of virtues (collaboration, communication, thoughtful learning, etc.) that will serve you well beyond the confines of this course, as a student, as a computer scientist, and as a citizen. To develop these virtues, it is important that you are an active participant in the class, reflecting on how you are engaging with the course, and what you can do to better support your own and your classmates' progress in the class. The qualitative questions on the problem sets and the 5min in-class surveys after each SRE are meant to foster such reflection.

Exams: There will be an in-class, 75min midterm exam on October 17, and a standard 3-hour final exam scheduled by the Registrar. Extension students will choose a 75-minute and 180-minute block, respectively, to take each exam, on the same day as that exam.

Grading

Breakdown. Half of your grade in CS1200 will be based on the problem sets. The midterm and final exams will comprise one third of your overall grade in CS1200, weighted proportionally to their length (75:180). The remaining sixth of your grade will be based on your participation, as evaluated by the reflection questions on the problem sets and the SRE surveys.

Problem set grading rubric. The problem sets will be graded using the following rubric:

- N: Not assessable. The assignment is too fragmentary or incomplete for us to assess the level of effort and understanding reached while working on it.
- L: Learning. Significant effort is evident, but there remain important misconceptions or gaps that will limit your ability to apply the skills and/or knowledge in the future (whether in the rest of CS1200, in later CS courses, or outside of your CS education).
- R-: Nearly ready to move on, but with review and revision recommended. You have achieved the main learning objectives of the assignment, but there are some gaps that should be filled in on your path to mastery.

- R: Ready to move on. Your work meets all of the learning objectives of this assignment, and contains only minor errors (which we will note in our feedback to you).
- R+: Beyond ready. The work is nearly perfect, goes beyond expectations in its clarity or insight, and/or explores optional extensions.

Exams will be given percentage grades.

The reason for this coarse grading scale is to move away from nitpicking over point deductions and focus on the end goal of acquiring sufficient understanding to competently apply what you have learned.

Revisions. The material in CS1200 is largely cumulative, so we wish to see you all reach an R-range grade on all assignments. But mistakes and misunderstandings are a part of the learning process, so we will allow you to submit short revisions (in the form of short videos) that clearly explain the misconceptions or gaps in your problem sets and the corrected understanding you have developed by studying the solutions. These revisions can increase up to five of your problem set grades from L to R- or R- to R (not both). On problem sets 0 and 1, these revisions can instead increase your problem set grades from N/L/R- to R. We encourage you to submit revisions on all of your L and R- grades (even if you receive more than five) to ensure that you have achieved the learning objectives of the assignments; at the end of the course, we will use your budget of five in the way that maximizes your final letter grade. To make sure you do not fall behind, there will be a deadline for revisions on each problem set, approximately one week after graded assignments are returned. We will not accept revisions on assignments that receive a grade of N (except problem sets 0 and 1) or R.

Participation grading rubric. For consistency, we will use the same N/L/R-/R/R+ scale for grading the elements of your participation grades (namely, the reflection questions on the problem sets and the sender-receiver exercises), but with the following interpretations:

- N: Your submission is missing or shows no engagement with the question or exercise.
- L: Your submission shows only superficial engagement with the question or exercise.
- R-: We will not use this grade for participation.
- R: Your submission demonstrates solid engagement with the question or exercise. We expect that the vast majority of participation grades to fall in this category.
- R+: Your submission goes well beyond expectations, in the thoughtfulness of your reflections and/or your contributions to the course's learning community. We expect that R+ grades on participation will be rarer than on problem sets.

Participation survey submissions cannot be revised, but we will drop the lowest grade received among the ~17 reflection questions and SRE surveys.

Letter grades. The table below describes the expected performance on each of the course elements corresponding to different letter grades.

letter grade range	psets	exams	participation
A to A-	all problem sets (revised) should have R range grades with at most 4 R-'s	indicate full mastery of the subject	~90% R range grades, no Ns.
B+ to B-	problem sets (revised) should have at most one N grade, with at most two Ls and a majority of Rs or R+s	indicate good comprehension of the course material	~70% R range grades, at most 1 N.
C+ to C-	problem sets (revised) should have at most two Ns, with a majority of R range grades	indicate an adequate and satisfactory comprehension of the course material	majority R range grades, at most 3 Ns.
D+ to D-	problem sets (revised) should have more R range grades than Ns	indicate some minimal command of the course material	more R range grades than Ns
Grades of R+ are equivalent to R for the letter grade ranges defined above, but the difference between R+ and R may affect final grade +/-s. Students whose scores are a mix of the above categories will be proportionally averaged—for instance, problem set grades including three Ls and an N but exams that indicate full mastery of the subject would likely yield a B-range grade. Before averaging, grades are converted to GPAs (e.g. a B-range grade is converted to a number between 2.5 and 3.5, with performance that just meets the standards for a B-range grade converted to 2.5, performance that essentially meets the standards for an A-range grade converted to 3.5, and interpolation in between).			

The exams do not have predetermined cutoffs (e.g. percentages) corresponding to letter grades because, with this being a relatively new course, we do not have enough data from past exams to calibrate the difficulty level. We will draw the cutoffs for each exam by qualitatively assessing the level of mastery shown by the submitted exams at different scores.

Late days. You have a total of eight (college)/forty (extension) late days that you can use for problem sets, using at most three (college)/four (extension) on any one assignment. (That is, an assignment beyond 3/4 days late will automatically receive an N grade.) Do not needlessly use up your late days at the beginning of the semester; they are meant for accommodating unforeseen circumstances or crunches from your other classes or other aspects of your life. Any extensions beyond these late days require a note from your resident dean (or advisor, in the case of graduate students).

Diversity and Inclusion[1]

We would like to create a learning environment in our class that supports a diversity of thoughts, perspectives and experiences, and honors your identities (including race, gender, class, sexuality, socioeconomic status, religion, ability, etc.). We (like many people) are still in the process of learning about diverse perspectives and identities. If something was said in class (by anyone) that made you feel uncomfortable, please talk to us about it. If you feel like your performance in the class is being impacted by your experiences outside of class, please don't hesitate to come and talk with us. As a participant in course discussions, you should also strive to be open-minded and respectful of your classmates.

Health Accommodations[2]

If you have a physical or mental health condition that affects your learning or classroom experience, please let us know as soon as possible so that we can do our best to support your learning (at minimum, providing all of the accommodations listed in your DAO letter if you have one).

Support Structures[3]

Everyone can benefit from support during challenging times. If you experience significant stress or worry, changes in mood, or problems eating or sleeping this semester, whether because of CS1200 or other courses or factors, please do not hesitate to reach out to Anurag, Salil, or other members of the course staff. Not only are we happy to listen and discuss how we can help you cope in CS1200, we can also refer you to additional support structures on campus, including, but not limited to, the below.

- Academic Resource Center
- InTouch (for SEAS graduate students)
- Counseling and Mental Health Services, 617-495-2042 (24/7 support line)
- Peer Counseling, including Room 13, 617-495-4949

Collaboration & Generative AI Policies

Students are encouraged to discuss the course material and the homework problems with each other in small groups (2-3 people). Discussion of homework problems may include

[1] Based on text by Dr. Monica Linden at Brown University.
[2] Based on text by Prof. Krzysztof Gajos at Harvard University.
[3] Based on text in the Harvard CS50 Syllabus.

brainstorming and talking through possible solutions, but should not include one person or large language model telling the others how to solve the problem. In addition, each person must write up their solutions independently of each other and of generative AI tools like ChatGPT,[4] and these write-ups should not be checked against each other or passed around. While working on your problem sets, you should not refer to existing solutions, whether from other students, past offerings of this course, materials available on the internet, or elsewhere. All sources of ideas, including the names of any collaborators (including AI tools, if used for brainstorming), must be listed on your submitted homework along with a brief description of how they influenced your work. Github Copilot and similar tools should be turned off when working on programming assignments.

In general, we expect all students to abide by the Harvard College Honor Code. We view us all (teaching staff and students) as engaged in a *shared mission* of learning and discovery, not an adversarial process. The assignments we give and the rules we set for them (such as the collaboration policy) are designed with the aim of maximizing what you take away from the course. We trust that you will follow these rules, as doing so will maximize your own learning (and thus performance on exams) and will maintain a positive educational environment for everyone in the class. We welcome and will solicit feedback from you about what more we can do to support your learning.

Content and Textbooks

The content covered in CS1200 this semester will be very similar to the previous offerings, possibly with some reordering to smooth out the pacing of the course. The lectures, lecture notes, SREs, problem sets, and section notes will capture all of the content that you are expected to learn during the semester, but we recommend the following textbooks for supplementary reading and practice exercises:

1. Background (to review or fill in material from CS 20 and CS 50)
    a. Harry Lewis and Rachel Zax. Essential Discrete Mathematics for Computer Science.
    b. CS50 OpenCourseWare and the Python documentation.
2. Algorithms (approx. the first two-thirds of the course).
    a. Tim Roughgarden. Algorithms Illuminated, especially Volumes I & II.
    b. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. Introduction to Algorithms, Fourth Edition.
3. Limits of Algorithms (approx. last third of the course)
    a. John MacCormick. What can be Computed?
    b. Michael Sipser. Introduction to the Theory of Computation.

[4] Non-native English speakers wishing to use LLMs for English writing should contact the instructors.