Stuff you need to know about CS144 (Fall 2019-20)

CS144 Staff

October 6, 2019

Prerequisites

The formal prerequisite for CS144 is CS110. CS144 is a systems course: 45% of your grade is based on programming assignments in C++, which means you need to be very comfortable with C++ and using standard debugging tools (if you are reading through thousands of lines of `printf` output you are doing something wrong). CS107 is not sufficient preparation: you need more programming experience. CS144 is offered every year, so please wait until you are ready before taking it. That being said, if you did pretty well in CS110 you should do fine in CS144.

Summary

CS144 is an introductory course about computer networks. You will learn about the basic *principles* of computer networks, for example packet switching, layering, encapsulation and protocols; and you will learn how applications such as the world-wide-web, video streaming and BitTorrent use the network to communicate. You will spend quite a lot of time learning about the specifics of how the Internet works – which is of course by far the biggest computer network ever built. You will learn how applications communicate reliably over an unreliable Internet. And you will build portions of the Internet yourself! In fact, we believe that in CS144 you build more parts of the Internet infrastructure than in any other undergraduate networking class anywhere. It's really fun to see how the individual pieces work: You build an Internet router, and a reliable data delivery service, and then you use it to communicate with remote servers.

CS144 is taught using a combination of lectures and videos. In previous years, it was entirely "flipped"; *i.e.* all the lecture material was taught by videos. This year things are different and we are going to mix things up: Some weeks, including the first week, will be based on recorded videos that you are required to watch in your own time. We will call these *Video Weeks*. Other weeks, including the second week, are based entirely on in-class lectures, and you don't need to watch any videos. We will call these *Lecture-only Weeks*. So why mix things up? We are teaching this way because we have found that some of the material (e.g. the basic principles you learn in week 1) are most efficiently learned by watching videos - the concepts are fairly simple and the material is descriptive; a video is a more efficient use of your time. Other material, such as when you learn about congestion control in week 4, is best learned in person, interactively in a lecture.

In addition to lectures, we will also have a few in-class guest lectures by outside speakers. For example, the head of networking at Netflix will come and tell us how they stream videos to our devices. All the guest lecturers are excellent speakers with many years of experience making networks work at huge scale. We will also have a few in-class exercises, which you will complete during the regular lecture time. These are designed to give you hands-on experience with tools that are useful for your labs.

Course Website

The course website can be found at https://cs144.stanford.edu.

Credits and Workload

If you are an undergraduate, you must enroll for 4 credits. Graduate students may enroll for either 3 or 4 credits. 4 credits means that we expect you to spend, on average, approximately 12 hours per week on CS144. This time is not uniform through the quarter. But generally speaking, we expect that you'll spend:

- 0-3 hours/week on the videos and quizzes,
- 3 hours/week in required class meeting lectures, and
- 6 hours/week on lab or exam studying,

for a maximum of 12 hours. You should expect to spend about 45 hours total on the labs.

Weekly Units and Videos

The course is organized into units, with each unit lasting one week. For example, week 1 is about the basic principles of networking, week 3 is about packet switching. Week 1 is a *video week*, which means you need to watch a number of short videos, which have embedded quiz questions. You also need to come to lectures on MWF. The embedded questions are to give you feedback on whether you are understanding the material and they are *not* graded. At the end of every *video week* you will take on end-of-unit quiz which *is* graded and is part of your overall grade. You will find the online quiz after the last video for the week, on the online system. You have until **2:00PM (PST) on the following Monday** to complete the videos and the online quiz. For example, class begins on Monday September 23rd and you have until 2:00PM on Monday, September 30th to complete the quiz in Unit 1 for credit. Note that the online course materials use UTC rather than California time: remember they are due at 2:00PM California time, regardless of what the website says.

Weeks 2 and 3 are *lecture-only weeks*, which means you don't need to watch any online videos. You need to come to lectures on MWF; we will give lectures to cover the material.

Every Friday, for both *video weeks* and *lecture-only weeks* the lecture will start with a 5-minute pop quiz.

Website: You can reach the course website, with a pointer to where you can find all the course videos, integrated quiz questions, and unit quizzes at: `https://cs144.stanford.edu/`.

Textbook: The optional course textbook is: Kurose and Ross, *Computer Networking: A Top Down Approach*, 7th edition. We will not be assigning readings, but it is an excellent reference to supplement the lectures and videos.

Class Attendance

All classes are **required**; your attendance will be part of your final grade. Throughout the quarter there will be class meetings where we hold an in-class exercise or guest lecture. The other class meetings will be lectures covering some of that week's material.

You have two free absences from required class meetings. Attending all except two will give you full attendance credit; each additional absence past the first two will deduct $\frac{1}{18}$ of your attendance grade.

At some point during each lecture, we'll announce an attendance code. You need to submit that code on the attendance form on the class website before 5 PM (PST) on the lecture day in order to get credit for the lecture. Credit will not be awarded for late codes.

We do not allow laptops to be used in class, except for specific in-class exercises (we will ask you to bring your laptop) and by the teaching staff. This is because using your laptop distracts you and the people sitting around you — we have noticed in the past, by standing at the back of the class, a sea of laptops working on assignments for other classes, Instagram, job applications etc. It creates a poor learning experience for everyone. If you have something you really need to get done during class, go do it and use one of your two free absences.

Exams

Exams are closed book. You may bring two double-sided pieces of paper to exams as notes. You must be able to read these pages without visual aids (e.g., magnifying glasses). Exam questions that depend on very specific facts (e.g., the header format of a protocol) will generally provide these facts to you.

Grading

Your grade in CS144 is based on class attendance, online quizzes in the class videos, weekly pop quizzes in-class on Friday, written problems (a midterm, a final) and 9 programming labs.

Class attendance	Each week	10%
Quizzes	Each week	10%
Lab 0	Warmup	5%
Lab 1	Reassembler	5%
Lab 2	Receiver	5%
Lab 3	Sender	5%
Lab 4	TCP	5%
Lab 5	Congestion control	5%
Lab 6	Link layer	5%
Lab 7	Router	5%
Lab 8	Capstone	5%
Mid-term Exam (90m)	In class	15%
Final Exam (120m)	In scheduled slot	20%

If you feel you were graded incorrectly on a homework, lab, or exam, please let us know as soon as possible. At the end of the quarter, we take the distribution of numerical grades and decide what ranges correspond to what letter grades. We don't decide on grade ranges a priori because sometimes exam questions are harder or easier than we thought they would be, and so we want to be able to adjust accordingly. However, CS144 is not graded on a curve: we decide on grade ranges, not class population percentages. We do not publish the numerical grade ranges corresponding to letter grades.

Late Policy

Programming labs are due at 5pm. We do not accept or grade late programming assignments. However, we know that you have other classes and sometimes you have time conflicts. To give you some flexibility, we give you three late days that you can use as you wish through the quarter, except that you can only use two for a given assignment - you cannot use all three for one lab. Using a late day means you have 24 hours extra to submit your assignment. We ask you email the staff list (cs144-aut1920-staff@lists.stanford.edu) by the lab deadline, letting us know if you are using late days. With weekly labs, the CAs want to be able to get feedback to you quickly. As such, we will start grading as soon as the deadline passes, and want to avoid redundantly grading submissions that weren't intended to be final. After you use up all three late days, you must turn your assignments in on time, else it will not be graded.

If you struggle with Lab 0 or Lab 1 because you find it difficult to write the code, you might want to consider dropping the class. The first two labs will give you early feedback on whether your programming skills are sufficient for the course.

If a real-life event (wedding, funeral, hospitalization, etc.) disrupts your ability to turn an assignment in on time, please let us know as early as possible. Clearly, some such events, such as a trip to the emergency room, are less expected than others, and we understand. Emailing the staff 48 hours before an assignment is due asking for an extension because you have a wedding to go to might be met with a frown; email us two weeks before the assignment is due and we'll do our best to accommodate. Our goal is to make sure you don't fall behind on the next assignment.

Incomplete Policy

Our policy is to never give incompletes for CS144. If you are falling behind or something life-changing comes up, please contact us immediately and we'll try to work something out. Contacting us early is better than late. Generally, taking too heavy a course load is not a sufficient justification: courses last a quarter for a reason and you are expected to be responsible for your own schedule. We don't allow incompletes because grading programming assignments outside the normal quarter is exceedingly difficult (and inconsistent).

Office Hours and Email

If you have a question about the class material or a programming assignment, you have three ways to ask: in person (office hours, after class, etc.), Piazza, and the staff email list. Here are some guidelines on how you can ask questions to maximize the amount and quality of help we can provide.

Please use Piazza for questions about programming assignments and general course questions. Using Piazza means that everyone can benefit from the answer; it may be that other students had the same question. Please do ask questions about the requirements of the assignment, the provided code, or the expected behavior of your system. Please don't ask questions that relate to how to implement a solution. For example, please don't ask questions that include or ask for source code. If you have any uncertainty about whether a question is OK, please email the course staff. You can find answers to almost any general C++ question on the web.

If you have questions about your particular solution to an assignment, you should come to office hours, or ask on a Tuesday evening lab session. For personal questions (e.g., arranging an appointment, questions about grading), please send an email cc'ing both instructors. Email is better than office hours for questions on grading because it may be the staff member at office hours wasn't the one who graded your assignment.

Generally speaking, it's almost impossible to answer programming assignment questions over email. The round-trip-time is too long, and it's not interactive. Email discussions often boil down to needing a TA to find a bug for you, which isn't very educational. Therefore, please come to office hours to discuss programming questions.

Honor Code

Our goal for this year is for zero violations of the Honor Code. In 2016 and 2017, we succeeded. But last year we had over 5 violations, which made us very, very sad. Honor Code violations are taken very seriously at Stanford and we want to encourage you to never venture over the line. We would love to have a quarter free of any Honor Code Violations. Let's make that our goal!

Most Honor Code Violations in CS144 happen when students reuse and modify code written by other students, including solutions posted on-line. In this course, we take the Honor Code very seriously and we expect all students to do the same. The good news is that the vast majority of students do follow the Honor Code. The bad news is that historical evidence indicates that some students will submit work that is not their own, not only shortchanging their own learning, but undermining the atmosphere of trust and individual achievement that characterizes Stanford's academic community. To protect academic integrity and the interests of all students, the course staff will investigate all possible Honor Code violations and refer them to the Office of Community Standards as necessary.

We will explain this in the first class *and* in a **mandatory video you must watch**, explaining the Honor Code for CS144.

If you have any questions or doubts about the Honor Code, please come and talk to either of the instructors. Honor Code violations are no laughing matter at Stanford and it is much better to ask what might seem like a silly question now than to risk your academic career. The Honor Code has a long tradition at Stanford dating back to Spring 1921 when the University first adopted the honor system. Today the Honor Code continues to govern academic conduct of both students and faculty at Stanford. The Honor code reads as follows:

THE STANFORD UNIVERSITY HONOR CODE

1. The Honor Code is an undertaking of the students, individually and collectively:
   - that they will not give or receive aid in examinations;
   - that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
   - that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.
2. The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.
3. While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.

The underlying premise of the policy is that all academic work represents independent, original work of the author and the Honor Code aims to foster an academic environment that encourages adherence to these principles. As we are all bound to respect and uphold the Honor Code, it is important to define acceptable and unacceptable behaviors with regard to this course so as to eliminate any ambiguity.

Permitted Collaboration: The following items are encouraged and allowed at all times for all students in this class:

- Discussion of material covered during lecture, problem sessions, or in handouts
- Discussion of the requirements of an assignment
- Discussion of the use of tools or development environments
- Discussion of general approaches to solving problems
- Discussion of general techniques of coding or debugging
- Discussion between a student and a TA or instructor for the course

Collaboration Requiring Citation: Two students engaging in more detailed discussions must be careful to document their collaboration. Students are required to include the names of those who provide specific assistance to properly credit their contribution, in the same manner as one would cite a reference in a research paper. The expectation is that even with a citation, the author must be able to explain the solution. Some examples of collaboration that require citation include:

- Discussing the "key" to a problem set or programming assignment. Problem set questions are often designed such that the critical concept takes careful thought and gaining that insight from someone else must therefore be documented.
- Discussing the design of a programming project. Design is a crucial aspect of the programming process and discussion can be valuable. Any design input received from others must be cited.
- Receiving assistance from another student in debugging code. While the TAs are the preferred source for advice, any detailed assistance from someone else must be credited.
- Sharing advice for testing. For example, if someone provides important information on lessons learned ("my program didn't handle the case where the value was 0") that source must be credited.
- Research from alternative sources. Researching related topics, such as through the Internet, must be documented if the solution submitted is derived from the research information.

Unpermitted Collaboration: All submissions must represent original, independent work. Some examples of activities that do not represent original work include:

- Copying solutions from others or knowingly allowing others to copy your solution. In particular, do not ask anyone to provide a copy of his or her solution or, conversely, give a solution to another student who requests it. Similarly, do not discuss algorithmic strategies to such an extent that you and your collaborator submit exactly the same solution. Use of solutions posted to websites, such as at other universities, is prohibited. Be aware that we photocopy some of the exams prior to handing them back. Also be aware that **placing your source code for the course in a publicly accessible repository where others can copy it is unpermitted collaboration.**
- Using work from past quarters. The use of another student's solution or the posted class solutions from a previous quarter constitutes a violation. We use a sophisticated software tool that cross-checks every assignment against every other assignment submitted this year, and previous years. It catches common code, even if comments and variable names are changed. In fact, in order to "fool" it, you have to change so much code that it would be quicker to do the assignment yourself (we tried it!). Developing good problem set questions and programming assignments often takes years and new assignments invariably have problems and that require polishing. To provide the most effective exercises, questions and assignments are commonly reused. Students retaking the course are expected to notify the course staff to avoid coming under suspicion. If you looked at solutions before taking the course, delete any such code you might have and email the course staff mailing list immediately to let us know.
- Studying another student's solution. Do not read another solution submission whether in electronic or printed form, even to "check answers."
- Debugging code for someone else. When debugging code it is easy to inadvertently copy code or algorithmic solutions. It is acceptable to describe a problem and ask for advice on a way to track down the bug. "What would you do to try to find this bug?" is an acceptable question; "Can you help me find my bug??" is not.
- Collaborating on or discussing the online graded quizzes before you have completed them. These are intended to be relatively easy, simple questions that test your basic knowledge.

This section on the Honor Code was based on policies written by Tom Fountain, Eric Roberts, Julie Zelenski, and the Computer Science Department at Brown University.