Everything you need to know about CS144

CS144 Staff

September 28, 2016

Prerequisites

The formal prerequisite for CS144 is CS110. CS144 is a systems course: 45% of your grade is based on programming assignments in C. Core, low-level systems (OS kernels, cloud services, databases, networking stacks) have been written in C for 30 years, and still are, for very good reasons. It is very important that you are comfortable with C and using gdb as your standard debugging process (if you are reading through thousands of lines of `printf` output you are doing something wrong). CS107 is not sufficient preparation: you need more experience. Otherwise you will find the programming assingments exceptionally difficult, they will take 3-4 times as long as the course workload plans, and all of your academics will suffer. Please don't make this mistake! CS144 is offered every year, so please wait until you are ready before taking it.

Summary

CS144 is an introductory course on computer networking, specifically the Internet. It focuses on the principles of computer networking, grounding them in practice by explaining how the Internet works from how bits are sent on wires to how protocols like HTTP and BitTorrent work. Students implement a handful of low-level protocols and services, including reliable transport, IP forwarding, and a Network Address Translation device. Students gain experience reading and understanding RFCs (Internet protocol specifications) as statements of what a system should do. The course explores many of the concepts in current practice and recent developments, such as net neutrality and DNS security.

CS144 is taught using a mixed classroom model. All instructional material is contained in videos recorded by the instructors, which students are expected to watch outside of class. The class meets twice a week. In one of these meetings (Thursdays), the instructors overview the week's material in lecture, revisiting the major concepts and answering questions. The other weekly meeting (Tuesdays) is a mix of in-class exercises, demonstrations, and guest lectures. In-class exercises include examining HTTPS certificate chains, TCP bufferbloat behavior, or examining a web browser's network behavior in detail. Guest lectures will focus on larger issues in the Internet today, such as surveillance, data centers, and net neutrality. Speakers include Jon Peterson, a former member of the Internet Architecture Board (IAB), ex-head TAs for CS144 who now work at Netflix, leading networking researchers, and Jeff Mogul, a key designer of HTTP/1.1 who now works in Google's platform networks group.

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. Lab 1, for example, is the hardest lab and is due at the end of the second week. But generally speaking, we expect you'll spend

- 3 hours/week on the videos and quizzes,

- 1.5 hours/week in required class meeting, and
- 8 hours/week on lab or exam studying,

or 12.5 hours. This means that you should expect to spend about 64 hours total on the labs. In our experience, students spend an average of 25 hours on lab 1, 10 hours on lab 2, 15 hours on lab 3, 10 hours on lab 4, and 2 hours on lab 5.

Course Videos and Quizzes

The course videos are organized into units, with each unit corresponding to a week of course time. Most videos within a unit have integrated quiz questions. These integrated questions are not graded. Each unit has one end-of-unit quiz which is graded. You have until 2PM on the following Monday to complete a unit. For example, class begins on Tuesday, September 27th. You have until 2:00PM on Monday, October 3rd to complete the quiz in Unit 1 for credit. Note that for internationalization the online course materials use UTC rather than Pacific time: remember they are due at 2PM, regardless of what the website says.

The motivation for the unit quizzes is that the flipped classroom model makes it very easy for students to fall behind on the material. This results in students trying to watch big bursts of many videos, which is a poor long-term learning strategy. As course exercises, labs, and guest lectures assume you are familiar with the material covered in units so far, we want everyone to stay up to date. The goal of these quizzes is not to be extremely difficult material that requires a lot of effort, but rather simple thought experiments that lead you to think through the material a bit.

Class Attendance

Each week, one class meeting is an in-class exercise or guest lecture. Attendance to these classes is **required**: your attendance will be part of your final grade. The second class meeting is a lecture lecture covering some of that week's material. Lecture classes are **not required.** The required classes are:

Tue, 10/4	Exercise: wireshark
Thu, 10/13	Lecture: Jon Peterson (Neustar)
Tue, 10/18	Exercise: bufferbloat
Thu, 10/27	Lecture: Rodrigo Fonseca (Brown University)
Tue, 11/1	Lecture: Ken Florance (Netflix)
Tue, 11/8	Lecture: Keith Winstein (Stanford University)
Tue, 11/15	In-class exercise: routing
Tue, 11/29	In class exercise: DNS
Tue, 12/6	Lecture: Jeff Mogul (Google)

You have one free absence from required class meetings. Attending 8 of the 9 will give you full attendance credit, each additional absence past the first will deduct $\frac{1}{8}$ of your attendance grade. This adds up! Missing a total of 4 will deduct 3.75%, which will often drop your grade one notch. To receive credit for an exercise, you must enter a code into a web page. To receive credit you must enter the code by midnight of the day of class. We encourage you to enter it at the end of class.

Thursday (sometimes Tuesday) classes are lectures that cover the week's material. Attendance at these lectures is not recorded and is not part of your grade.

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, written problems (a midterm, a final) and five programming labs.

Class attendance	Each week	10%
End-of-unit quizzes	Each week	10%
Lab 1	Oct 7	10%
Lab 2	Oct 19	10%
Lab 3	Nov 9	10%
Lab 4	Nov 30	10%
Lab 5	Dec 7	5%
Mid-term Exam (90m)	Nov 3, in class	15%
Final Exam (180m)	Dec 13, 3:30PM	20%

Additionally, you can earn up to 5% of extra credit (enough to move you 1-2 grades up) for answering questions well on Piazza. Being able to clearly answer questions to your fellow students shows a good understanding of the material. Try to write answers that staff think are "good answers"! A maximum of 5% can be earned by the most helpful, with a sliding scale depending on how helpful students are.

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 what numerical grade ranges correspond with letter grades.** We keep these secret because we've found publishing them sometimes leads a few students on the edge to try to scrape up a few points here or there through persistence rather than a mistake in grading, and this reduces our ability to address other student questions.

Late Policy

Programming labs are due at noon. You may turn in any programming lab late, until noon two days later. If you turn a lab in late, its maximum grade is capped at 90%. This does **not** mean 10% is deducted. A 75% will be a 75%, but a 95% will be a 90%. The idea is that you can turn in something that mostly works and receive full credit for your work, rather than fall behind in class in order to make up for a penalty. Programming labs turned in after this late date receive no credit. If you have to turn in Lab 1 late because you find it very difficult to write the code, you might want to consider dropping the class. Lab 1 is intended to give you early feedback on whether your C programming skills are sufficient for the course.

If a real-life event (wedding, funeral, sports match, 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 major concern in such circumstances is your falling 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. But, for example, a few years years ago H1N1 knocked out a few students for

2 weeks, and we were able to make accomodations and give advice on how to proceed. We don't allow incompletes because grading programming assignments outside the normal quarter is exceedingly difficult. Furthermore, chances are you'd rather not spend the spring holiday filling in protocol header fields.

Office Hours, Forums, 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.), the course forum, and email to the staff email address. Here are some guidelines on how you can ask questions to maximize the amount and quality of help we can provide.

Please use the forum for questions about programming assignments and general course questions. Using the forum 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.

Please use email to the staff list for personal questions (e.g., arranging an appointment, questions about grading). 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 2015, we succeeded. Please help us succeed again this year!

We've found that most cases are because students referred to solutions from 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.

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.