Course Information

Course Overview
This course is designed as a deep dive into the design, analysis, implementation, and theory of data structures. Over the course of the quarter, we’ll explore fundamental techniques in data structure design (isometries, amortization, randomization, word-level parallelism, etc.). In doing so, we’ll see a number of classic data structures, as well as some more modern ones. By the time we’ve finished, we’ll have seen some truly beautiful strategies for solving problems efficiently.

Instructor
Keith Schwarz (htiek@cs.stanford.edu)
Office: Gates 178
Office Phone: (650) 723-4350

TAs
Anton de Leon
Ryan Smith

Anton and Ryan are CS166 veterans. Both of them have taken CS166 and TAed for the course in the past, so feel free to ping them with questions!

Email
The course staff can be reached at cs166-spr1920-staff@lists.stanford.edu. Please don't hesitate to send us emails! We're here because we genuinely love this material and want to share it with you. If you have any questions on the material, or if you're interested in exploring more advanced content, please get in touch with us. We'd be happy to help out.

Lectures
We will be holding lectures remotely via Zoom on Tuesdays and Thursday from 3:00PM – 4:20PM using meeting code 553-965-838

Those lecture time slots are designed to be interactive – we’d like you to ask questions, and we’ll have some time built in for folks to think over ideas in small groups. We know that not everyone will be able to call in at that time, either due to time zone differences, due to other academic commitments, or due to personal circumstances. We will therefore be making independent recordings of lectures and posting them on Canvas. You are not being recorded if you call into the in-person lectures, so feel free to speak freely. If you do call in, we encourage you to participate in small group discussions and to ask questions, though you are not required to do so.

Stated differently – you’re encouraged to participate online in the regular lectures if you can, and if you can’t you’ll have access to recorded videos you can use instead.

Units
CS166 is offered for either three or four units. Undergraduates are required to enroll for four units, while graduate students can enroll for either three or four units. The course content and requirements are the same in the three-unit and four-unit versions of the course and the unit flexibility is purely to help graduate students stay under unit limits.

Website
The course website is http://cs166.stanford.edu and it’s loaded with resources for this course. There, you’ll find all the course handouts, the syllabus, links to readings, and all sorts of other resources.
Office Hours

It's important to us that everyone in the course feels that they can easily get in touch with the course staff when they have questions. We will be holding a large number of office hours time slots throughout the week, and we're making an effort to set them up at times that will work well for everyone in the course regardless of time zone. The office hours calendar can be found online here.

Prerequisites

The prerequisites for this course are CS161 and CS107.

From CS161, you should be able to design and analyze nontrivial algorithms and write proofs of correctness. You should be comfortable using asymptotic notation ($o$, $O$, $\Theta$, $\Omega$, and $\omega$), solving recurrence relations, manipulating inequalities, and simplifying summations. We'll also expect that you're comfortable with divide-and-conquer, greedy algorithms, and dynamic programming; that you're familiar with randomized algorithms (and related concepts like universal families of hash functions); and that you're comfortable writing correctness proofs for algorithms of each of these types. You should also feel comfortable with standard algorithms like Dijkstra's algorithm, Prim's algorithm, quicksort, mergesort, etc.

The CS161 prerequisite, by transitivity, also means we assume you have the equivalent of CS103 (discrete mathematics, automata, and proofwriting) and CS109 (probability and basic combinatorics) as well. If you have never written a formal mathematical proof, or if the phrase “linearity of expectation” doesn't ring a bell, you may want to come talk to us before jumping into CS166.

From CS107, we expect that you're comfortable writing and testing nontrivial programs and working from the command line. You should also feel comfortable with binary representations of numbers. We'll expect that you've at least heard of the memory hierarchy and are comfortable with the idea that not all memory accesses take the same amount of time. Additionally, we expect that you'll be comfortable writing code in both C and C++.

If you're unsure whether CS166 is the right place for you, please feel free to get in touch with the course staff.

Readings

The recommended reading for this course is Introduction to Algorithms, Third Edition by Cormen, Leiserson, Rivest, and Stein. We understand that not everyone has a copy of this book or can get a copy, and that's okay. There's nothing in that textbook that you'll absolutely need for the course, and it's mostly there as a reference in case you'd like to look at certain topics in more depth.

Additionally, there will be a variety of readings posted online (papers, course notes, slides, articles, etc.) Check the website for details on the readings for each lecture. I will try to present the salient features of each data structure in lecture, so depending on your learning style, you may find it useful to do the readings right before or right after lecture.

Assignments

Routine assignments in CS166 this quarter come in two flavors. First, there are problem sets. Problem sets can be completed either individually or in a pair. Whether you work individually or in a pair, you'll submit exactly one copy of the problem set. You're welcome to talk to the course staff about the problem set questions and get help and support as needed.

Second, there are individual assessments. Think of the individual assessments as mini take-home exams. As the name suggests, you must complete the individual assessments individually. You're welcome to ask the course staff clarifying questions on the individual assessments, but otherwise we'll leave it up to you to figure out how to solve these problems.
Research Project

CS166 culminates in a research project in which you'll do a deep dive into a data structure topic of your choice.

Historically, in the research project, we've asked students to explore a particular topic in depth, becoming the "resident experts" on the topic. We've then asked students to do something "interesting" with their topic, whether that's running experiments that elicit something not well-known about the topic, or proving a broader result about some data structure or algorithm, creating a visualization of the data structure that aids in understanding, or uncovering some pattern or trend that is not obvious at first glance. The project then concluded with a written report and a 30-minute presentation to the course staff.

We will be spending the first few weeks of the quarter working out the logistics of how the research project will work this quarter. Our goal will be to find a system that lets people explore a topic in depth that also works well when courses are run remotely.

Exams

We will not be holding any exams this quarter.

Grading

CS166 is offered on an S/NC basis this quarter. Accordingly, we've structured the grading in CS166 around the following principles:

- We want you build a broad competency with all the topics from the course rather than a deep competency in only a few areas.
- We want to balance your need for flexibility during difficult and unpredictable times with our internal logistical needs for predictable deadlines.

With that in mind, we will be grading each individual assignment this quarter on an S/NC basis. To earn an S grade in CS166, you must earn an S grade on all five problem sets, five of the six individual assessments, and the research project.

To help you reach this goal, **everyone is permitted to resubmit up to four problem sets or individual assessments.** If you earn an NC on something you've submitted, we'll reach out to you to let you know what, specifically, we'd like you to patch up. You'll then have some time (precise amount TBA) to revisit the work you submitted, make edits, and then resubmit for credit.

We don't anticipate that you'll run out of resubmits. If that happens, we'll reach out to you, set up a time to chat, and see if we can find a way to move forward.

You should think of resubmits as a supercharged version of the "late day" concept we use in other CS courses. If you can't complete the assignment on time, do your best and submit what you have by the deadline. If that's good enough to earn an S, great! You're done. If, on the other hand, it earns an NC, then we can give feedback about what specifically you need to patch up, and you can then fix those parts and submit the revised version later. As a result, we won't be giving out traditional late days in CS166. If you'll need more time to submit some work this quarter, let us know at least 24 hours before the deadline and we'll see what we can do.

A note on working with partners: if you work with a partner on a problem set and end up receiving an NC on that problem set, both you and your partner will be charged a resubmit. In particular, please make sure that you and your partner have both vetted everything you submit.

**Resubmissions cannot be used on the research project.** We'll be offering feedback on the research project at several stages during this quarter, so you'll have ample feedback before you have to turn everything in at the end of the quarter.
Incompletes  If you have a medical or family emergency and cannot complete the work in this course, you may contact Keith (not the TAs) to request an incomplete. We reserve incompletes only for emergencies, so we do not grant incomplete grades for poor performance on the assignments or project, nor do we offer incompletes for busy work schedules.

In order to be eligible for an incomplete, you must have completed all of the assignments (except possibly the most-recently-due assignment) and must have a satisfactory academic performance as determined by the course instructor.

All incompletes are worked out on a case-by-case basis, and the instructor retains final discretion to approve or reject any requests for an incomplete.