Answers to your questions

(minus those I answered in class or which are already answered on the syllabus—I also combined some questions that were essentially the same).

What are your ambitions for your YouTube channel?

Pretty standard. Teach the world to program. Have fun in the process.

What is something you know really well but don't understand? I.e. I'm comfortable with the actual process of coding, but to me compilers are essentially black magic donated by our electron overlords.

Operational amplifiers are a bit like that for me. I use them a lot, and would definitely struggle to explain the physics behind how they work.

I would like to know if you think hotdogs are sandwiches because I am in a heated debate with my brother. I don't think they are but I would love to hear your opinion.

I think it depends what else you put on the dog. Oh, no. My wife saw this one, and now we're in a heated debate over it. She thinks it's never a sandwich. I sort of think of bologna as a big fat hot dog, and people call bologna sandwiches, sandwiches. So, maybe.

Where did you get your dress shoes?

Johnston and Murphy

Considering you are a system's oriented guy, have you ever tried the Rust language? And if so, what's your opinion on it?

I've been playing with it a bit lately, when I get some time. That said, I'm not sure I have enough experience to have a strong opinion yet. Seems promising.

What's your astrology sign?

Aries.

What were you doing when you were 20-21?

I was living in Cambodia, as a missionary for my church.

What is your biggest principle in terms of coding practice?

Stick with it, get your hands dirty, and explore. Oh, wait. That's 3. Sorry.

What made you want to start a YouTube channel?

Like I mentioned in class, it started with 3220. Well, I first created a channel so I could share little videos my kids and I made (privately) with family members. Then I was traveling for a conference, and recorded a video for my 3220 students to help with one of their projects. They liked it, and so I decided making more, and things just sort of evolved organically from there. It's been fun.

Do you think that MacOS is a good operating system? All of my business major friends love it but I think it is much less efficient than Windows or Linux.

There are a different ways an OS can be "efficient". I like MacOS, mostly because it makes me more efficient (in terms of how much time I spend on things). I haven't done a head-to-head comparison in a while. So, I'm not sure how they stack up in terms of other performance measures.

What was your favorite past-time in college?

I did a few different things. Indoor soccer, fly fishing, hiking. In grad school, I picked up cycling and cheesemaking.

When did you have your "ah-ha" moment (when things started coming together and making more sense) in Computer Science?

I took a somewhat nontraditional path. I discovered code when I was 12 — that was a sort of ah-ha moment when I discovered that I could create software (rather than just use it). Through middle school and high school I had a lot of little moments when stuff clicked. I don't remember the year (sometime in high school), but I remember when I wrote my first linked list, and suddenly really understood how pointers work. Compilers class in college was also mind-blowing.

What's your favorite asm instruction?

I'm not sure I have one, but that's no fun…so…MOV, maybe? I'm easily entertained.

Are you the guy that makes the code smells videos?

I've made a few videos about code smells. Not sure if they were the ones you watched. Did the person on the videos look like me?

What is something you do to avoid tech burnout over time? You seem to enjoy Comp Sci and I would like to maintain that enjoyment also.

Hmmm…choose to work for organizations with great cultures — organizations that look out for their people. Also, developing strong tech skills helps, because you have more choice in what you do for work, who you work for, and you often can do more in less time, all of which help avoid burnout.

Do you prefer Windows or Mac? What is your least favorite operating system?

I prefer Mac, so far. I started my career as a Windows developer and user. But, I really like programming in Unix-style systems. So, I've moved toward Linux and MacOS and really haven't looked back. At this point, using Windows is a pretty painful experience.

How do you define success in your course?

Well, I assign grades based on the syllabus, but I consider myself a success when—after the experience—you tell me that your life/job/confidence/prospects are better because of this class.

How do you best build relationships with students?

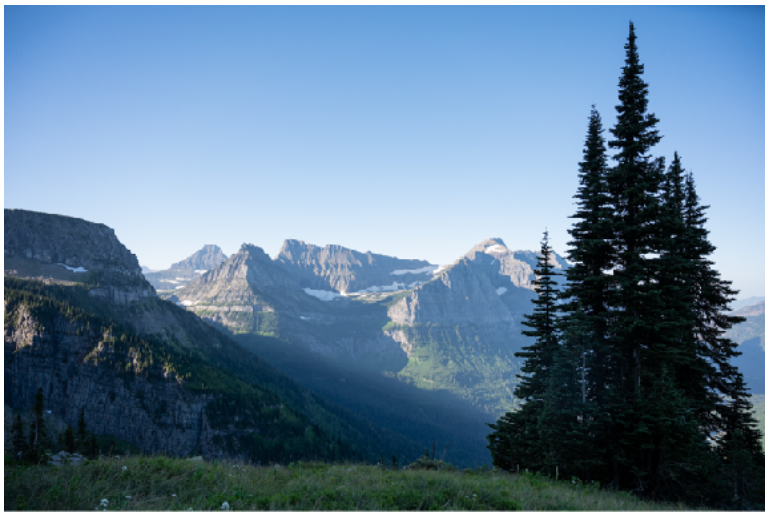
With 50–100 students, this is tough. I require office hour visits, in the hope of talking at least once with all of you. I try to take time to talk with students about things both in and outside of class. I try to give you work that eventually blesses your life (in the moment, it might not always feel like a blessing).

Will there be a review session before exams?

Yes, but it will be driven by your questions. I'm not just going to go over stuff again, unless you ask about it.

Do you like to travel? Where have you been? Where is your favorite place you have traveled?

Yes. A lot of places. That's almost impossibly tough. Highlights outside the US include…Cambodia, Botswana, Namibia, Korea, Belize, Sweden, Germany, France. Glacier National Park is one of my favorite places inside the US (photos below).



Will I be able to make my own OS after this class? Have you written your own OS?

Yes. Well sort of. You will have the conceptual background to design an OS. You'll also have the programming background (minus some assembly for low-level initialization and interactions) that you would need. But, there are a lot of implementation details that we probably won't get into. Someday, I hope to evolve this class into one where we actually design a simple OS. Maybe in a few years.

As for myself, I have written OS pieces. I also wrote most of the first version of the OS for Amulet (one of my research projects). That said, it was pretty simple. Some would call it a runtime system rather than an OS, since it doesn't provide a lot of common OS functionality (memory isolation, for example).

Did you graduate from Clemson?

Nope.

How difficult do you usually try to make your projects? Some classes the projects take a couple hours, others are 50+ hour chores, and some classes do both throughout the semester, but the professor usually has expectation for how long the various projects should take to complete.

The time the projects take depend on your preparation. I will give you roughly 3 weeks per project, and I expect them to take the average 3220 student one week (maybe 10–20 hours) to complete. That said, I've had students finish projects in 3-4 hours, and I've heard of people taking 50+. I would expect the middle two (#2 and #3) to be the most challenging. That said, people often don't take me seriously on project 1, and so project 1 usually ends up doing the most damage to grades.

What (in your opinion) has been the coolest thing a student of yours has gotten involved in?

That's tough. A lot of my students work on stuff they can't tell me about. My students have worked on software for airplanes. One is designing technologies for microsatellites. One of my students is a professor at Northwestern, working on a bunch of really cool batteryless tech for healthcare.

A few of my former students created the Clemson Makerspace and CUHackit, which are both really cool in terms of their potential to transform Clemson for the better.

What is the most interesting topic in this course? What is the hardest topic you will cover, this year?

I have a hard time picking a favorite. Threading and concurrency, memory management, and file systems are all great (I know I just described 75% of the course, but that's the best I can at choosing a favorite).

As for difficulty, there is a lot of variance here. Threading and concurrency usually causes the most difficulty (but some have more trouble with scheduling and file systems. Memory management (paging, segmentation, and address translation) can sometimes take students a little while to wrap their heads around, but it's not really hard.

What will the programming assignments be like?

They will be moderately complicated programs or software libraries. I will give you a functional description, the autograder, and three weeks to complete the project. Each project will be designed to help you better understand some facet of the class (like scheduling, resource management, concurrency, mutual exclusion, or file system structure).

Respect the assignments and start early, and you should be fine.

How would you suggest that we prepare for quizzes? Are you mainly pulling content from in-class discussions, or the textbook reading?

I pull content from both the discussions and the readings, with emphasis on stuff that we have discussed and read. Prepare for quizzes by studying right after class meetings and using active recall (as discussed in class on Tuesday).

Why did you make the autograder? How long did it take you?

Students were always super stressed about projects, because they had to wait and didn't know if their code crashed and resulted in a bad grade. Also, manually grading projects is mind numbing. I feel bad asking a TA to do that. So, it just made sense. Version 1 took me a week during the holiday break back in … 2014 or 2015, and I've done a lot of work on it since. Totally worth it, though.

What was your favorite class in undergrad? Favorite class in grad school?

Three-way tie. Operating Systems (seriously), Algorithms, and User Interface design. My networks class was pretty great too.

Grad school is a bit different. Classes aren't so much the focus, but David Jensen's research methods class was a definite highlight. Micah Adler's Algorithms class made me question my sanity and value to the world as a computer scientist.

What classes have you taught at Clemson aside from this one?

Embedded Systems Prototyping, Graduate Embedded Systems. Software Dev. Fundamentals (CPSC 2150), Networks/Network Programming (CPSC 3600). I also lead the Embedded Networked Systems Seminar.

What is your favorite course that you've taught?

It's tough to pick. I love teaching this class. It's probably tied with Embedded Systems Prototyping, which I teach less often.

What's something you've never taught that you really want to?

I would love to teach Computer Organization/Architecture (the assembly class), but an embedded systems-oriented version.

Did you learn most of your coding from classes or on your own?

I learned a lot from my classes. And, I've learned a lot on my own. I don't think you really know much in CS until you do it (or use it). Many of my classes gave me a lot of hands-on experience, and helped me see what was possible. My various jobs and projects have taught me a lot as well.

Are you a sports fan? Favorite sports?

I like sports. I'm definitely not a super-fan, but you will occasionally find me watching or playing something.

Favorites:

To play—soccer, running, cycling, ultimate frisbee.

To watch—soccer, college football (when the teams are good), high school track and cross country (when my kids are competing).

What is your favorite thing about programming?

The combination of a puzzle (the challenge) and creativity. I like producing things that make things work better (and feeling like I can speak an alien tongue).

What is the coolest thing you have made?

My three daughters (can't take all the credit, though). :)

My first successful wheel of mountain-style cheese (Emmenthaler) was pretty cool. Mostly it was just really cool that it actually worked.

I'm also fond of the chicken coop that I built with the automatic solar-powered door (to keep the raccoons and foxes out).

Most challenging project?

Being a father. It's also the most rewarding project, so far.

Favorite programming language?

That's a tough one. It depends on the job at hand. For low-level, or embedded stuff, probably C. For high level OO-style programming, probably Ruby. For quick and dirty shell scripting, either ruby or python.

Do you have dogs? Why or why not? Pets?

No dogs. I like dogs, but I prefer them to belong to someone else.

My basic pet philosophy has long been that I prefer pets that 1) do something useful for me or my family, and 2) have a straightforward (and legal) way out for me, if my kids stop helping me care for the pet they asked for.

Naturally, I eventually broke down, and was adopted by an abandoned cat (The Great Catsby). I'm definitely his favorite family member (with zero effort on my part), which really bugs my 13-year-old daughter, who won't leave the poor guy alone.

I also keep chickens (see #1 and #2 above) and Shubunkin goldfish (still trying to figure out how to get them to do something useful).

I think it would be great if someone would develop a breed of pigmy cattle or bison that you could raise in a normal-sized backyard. They would mow your grass and provide meat and milk, and you could claim it was an odd breed of dog if the neighbors started asking questions. That might be my ideal pet, if it existed and wasn't an inbred fragile genetic mess.

Why teach OS (assuming you have a choice)? Why do you love OSes so much?

Yes, I do have some choice in the matter. I love the topic. It's close to my research. I like low-level hands-on classes, and OS is both low-level and hands-on. The OS is the foundation on which we compute (if you don't count the hardware). Improvements at that level have impact far beyond a single application.

Why do you think you have a reputation as a "hard" teacher?

It's one of life's great mysteries. I think students learn the most when challenged. And, so I try to challenge you. Past students tell me that they worked hard in my class, but felt that they came out better prepared for life and career.

I have been told that I write challenging tests, though students seem to do fine on them (no shortage of As). One thing that you will find, is that I try to write test questions that evaluate your ability to apply the principles and tools from the class. I don't like pure memorization questions, because out in the real world, memorization doesn't get you very far.

I have also been called "hard" because I don't accept late work and will give you a zero (0) for code that almost works, but doesn't compile because you forgot a semicolon. Start early and test your code, and I think you will find me to be a fair and reasonable grader.

How did you get into computers? Why CS?

Very cliche. In the 5th grade I wandered into the computer lab one afternoon and noticed some of the older students (11th or 12th graders) programming. I asked them about it. One of them gave me a

Pascal compiler (on a 3.5-inch floppy disk). I took it home and played with it. I eventually moved on to C/C++, and the rest is history. I've always enjoyed how computing is both creative and systematic. I also love that I can use my skills to work on just about any of the world's many problems.

What did you struggle with most when you were a novice computer scientist like us?

I struggled with the same stuff you all do—pointers. :) Well, that and understanding scoping and stack/heap issues. Back then, the main difference was that it was much harder to find good documentation. The Internet was barely a thing.

What you do with programming outside of teaching computer science at Clemson?

Well, I do a lot of research (see other answers). I also have some hobby projects that I work on. In recent years, I've helped with a programming and electronics course at RC Edwards Middle School, which is currently on hold due to the pandemic. I try to teach my kids a little programming, as well, when they're interested.

What research do you do? What are your research interests? What is your specialty?

My research focuses on mobile computing, sensor networks, and embedded systems. Specifically, I try to make small, flakey, resource-constrained computing devices perform better and do more, in spite of their shortcomings. My current projects are many. I have developed wildlife tracking technologies with the USGS (desert tortoises and sea otters). I am working on security and privacy issues in wearable medical devices. I have worked on making manufacturing environments smarter and more efficient. I'm also generally interested in exploring techniques for making RFID-scale batteryless devices more generally useful, even though they reboot all of the time and have trouble keeping time and making forward progress. If you are interested in research, and this sounds interesting, then we should talk.

Do you do research with undergrads?

Yes, and quite a few former 3220 students have worked in my lab. I usually have between 1 and 5 undergraduate students working with me, though I currently don't have any (they all just graduated). Some are working at places like Apple, Google, and Amazon, as well as some smaller startups. Others have gone on to Ph.D. programs at places like CMU, Dartmouth College, and the University of Michigan.

Why did you choose to work in academia, rather than industry? I assume industry would be more lucrative.

It's a fairly complex question.

When I started graduate school, I was sure that I was headed to industry after my PhD.

But, I found that my research fits best in academia. I also enjoy the intellectual freedom, and I really like working with students (both undergrads and grads). My experiences in industry were all positive. The money was fine in both. But, you really don't get a PhD for the money.

I think the following article (by Nick Feamster at Univ. of Chicago) spells it out pretty well.

http://greatresearch.org/2013/08/30/industry-or-academia-a-counterpoint/

What was it like for you in graduate school?

Wow. I could write pages on that one. It was a big part of my adult life. It was fun and challenging. I learned to be a parent, survived cancer, and learned how to be a scientist and a better communicator. I published my first papers, learned how to design and debug circuits, met a lot of great people, designed my first programming language, wrote my first compiler, and made solar-powered tracking devices for turtles and mongooses. It was a great experience, but not an easy one. I'm happy to talk more about it, if you have specific questions.

What are your students' most common pitfalls/mistakes? How to avoid them?

The most common mistakes are simple.

- Procrastinating
- Not taking the time to learn the material, well, outside of class.
- Not asking for help.
- Striving for mediocrity.
- Not taking responsibility for their own choices and their own lives.
- Not using active learning methods

You'll notice that most of these involve **not doing** something (one just involves not caring).

The solution to all of these is really to take responsibility for your life. Learning to manage time, stay healthy, and put first things first is critical. I regularly hear students explain how they performed poorly on an assignment because they had another project in another class (this happens; you need to plan

ahead), because their computer “crashed” (this also happens; backup your code, have a plan B), or because they got drunk with their frat brothers instead of starting early (yes, I have heard this one in office hours).

Occasionally, a student has something come up that is truly out of his or her control (usually a medical issue), and I try to accommodate those issues. Most of the Cs, Ds, and Fs, in this course are the result of students not following my advice or not doing something that needs to get done.

What’s the best way to learn OS-related topics outside of school?

Download the Linux kernel and start messing with it.

Seriously, just do stuff. Dig into OS code and figure out how the machine handles something (reading/writing to files, for example). An operating system is just a program.

Do you have a faith?

I do. I am a member of the Church of Jesus Christ of Latter-day Saints.

What kind of music do you enjoy? What are your musical interests?

A wide variety. It’s always changing, but today I was listening to Grits, Matt Kearney, Dr. Dre and Snoop Dogg, Counting Crows, The Mighty Mighty Boss Tones, Pentatonix, Matisyahu, Assorted Khmer music (wishing Spotify would add some Meng Keo Pichenda), John Rutter and the Cambridge Singers (classical choral), and some gregorian chant. Last few years, my kids brought home a lot of Imagine Dragons, Taylor Swift, and the Hamilton soundtrack. These days, they’re in a classic rock phase that’s proving interesting. My daughters also recently discovered that I was into ska back in the 90s, which has led to some interesting walks down memory lane. I like music that inspires and energizes me. I also like music that celebrates the things that are great and noble about the human experience. I’m also an OK singer, a rusty pianist, and occasionally a novice composer/arranger.

Favorite type of food?

Hard to say. Khmer, Thai, Indian, Arabian, Greek, Mexican and French are all outstanding. But really, I just like good interesting food. Taqueria Texas and the new Thai place in Central are welcome additions to the Clemson scene. I miss Yolk.

Where were you born? Where did you grow up? Where have you lived? What was your favorite place to live?

Born in Provo, UT. Grew up mostly in Utah, with a few brief stints in NV and OK when I was little (Dad was in the military). I’ve lived in Utah, California, Nevada, Oklahoma, Massachusetts, New Hampshire,

and South Carolina. I also lived in Cambodia for two years, Germany for a summer, and spent the 2019-20 academic year living in Botswana. I've thoroughly enjoyed all of them that I can remember (all but NV and OK). Can't really pick a favorite, but Cambodia was probably the most life-altering.

What did you want to be when you were young?

How young? Early childhood, I was interested in being a scientist, astronaut, or forest ranger—maybe occasionally a professional soccer player. I've wanted to be a technologist for a long time—it started coming on in high school.

What is your least favorite student behavior? Pet Peeve?

Cheating. Followed by striving for mediocrity and complaining. See pitfalls comment above.

What industry jobs did you have before Clemson? What experience do you have working in the field?

During college, I worked for IBM (first as a tester and then as a developer), and then for two startups: one that provided an online document management service for companies, and another that made road-side radar sensors for traffic counting. At the latter, I wrote a little firmware for the sensors, but mostly developed traffic prediction algorithms and code that handled the data from the sensors. I also got to build a video sequencing server that put news shows together for a local TV station (but that's a long story). After graduating, I worked for another startup, that made diagnostic software for automotive technicians. During my PhD, I also worked for one of Intel's research labs for a summer.

What has your career path been like up to this point? What experiences can you share?

I'm sure you'll hear some experience throughout the semester. It's hard to even know where to start. I mentioned my industrial jobs in the previous answer. When I got out into the workforce, I worked with a few PhDs (machine learning experts). It didn't take long for me to notice that those guys were having more fun. They were working on more interesting problems. My work life wasn't bad, but I started to think I wanted to be doing more bleeding-edge type stuff. So, I decided to go for a PhD. After my PhD, I worked as a postdoc at Dartmouth College for 2-years while the recession ran its course, and then got a faculty job at Clemson.

Do you speak any foreign languages?

German (a little and very rusty), Cambodian (reasonably fluent with everyday conversation topics), a very little Setswana, and a phrase or two in Spanish and Vietnamese.

What school did you go to?

High School: Mountain View HS in Orem, UT

Undergrad: Brigham Young University (BYU)

MS & PhD: Univ. of Massachusetts Amherst

What was your major? Is your background CS or EE?

Computer Science. I consider myself a self-taught electrical engineer, as well, though I don't have a formal EE education. I've just done a lot of hardware hacking in my research, because I couldn't buy the hardware I needed. So, I learned how to make it.

Whats your favorite breakfast cereal? What do you eat for breakfast?

I actually make my own. I know it's a bit odd, but I buy grains, seeds, and nuts in bulk and mix my own whole-grain cereal.

Hobbies?

[In no particular order] gardening, running, cheese making (and eating), cycling, hiking, cooking, playing soccer, travel, playing the piano (badly), singing, composing/arranging music, flyfishing…oh and yes programming and electronics. I used to be a bee keeper. I keep chickens — my current batch are by far the stupidest I've ever had.

If you never had to work again what would you do?

I might still do computing research and teach students—I really do love my job. I might negotiate away some of the less fun parts. I might also take any one of my hobbies to an unhealthy extreme. I think the thought of me retiring makes my wife a bit nervous.

What is the worst piece of advice you have ever been given?

It's hard to know what the **worst** one was.

I was told that research on batteryless intermittent computing was too hard and that I should switch to something less risky. Now, it's the bulk of what I do. I'm really glad I didn't follow that advice.

Did you always want to teach? How did you get into teaching CS?

Nope. I was convinced that my career was going to be in industrial research, until half way through graduate school. I just didn't have enough experience until then to make an informed decision. I just realized that 1) I love to teach, and 2) that my research fits better in academia than industry.

Do you enjoy teaching this course?

I do. It's one of my favorite topics.

What other field would you be in, if you weren't in computer science?

Does computer engineering and electrical engineering count? Probably something medical, math, or physics, but I'm really glad I'm in CS.

What advice do you have for someone interested in a career in academia?

Start planning now. Get advice now, from several people. Start doing research, now.

Seriously, there isn't enough room in this document for all of the advice you need to hear, but too many students wait to start asking for advice until it's too late to do much about it. Let me know if you want to chat about it sometime.

What is your passion?

I have a few. My family. My faith. Science and Teaching.

Other than OS/Embedded Systems, what is your favorite CS topic?

That's a tough one, akin to asking me to pick a favorite parent. I would probably have to say it's a toss up between programming languages, compilers, and algorithms. I think machine learning is pretty cool, too.

How did you end up at Clemson?

I applied, and they gave me a job. I chose Clemson, because there were strong faculty here that I could work with, and both faculty and students seemed genuinely happy to be here.

How long have you been teaching? How long have you been a professor?

Since 2012 at Clemson. Two years before that at Dartmouth College. A little bit before while in grad school at UMass Amherst.

How much do you enjoy making YouTube videos? You said you started it to help teach your Operating System students, but it seems like it sparked an interest in you.

Yeah, I like it. I've been interested in photography for a while, and I like playing around with the processing, lighting, and audio. Seeing how things play out with the YT algorithm is interesting. And, I also like that it helps me reach a wider audience. It's a pretty solid mechanism for recruiting graduate students.

Is there a certain routine you would do if you have a problem that you cannot figure out?

This is tough, just because "problem" covers a lot of different situations. I would slow down, and break it down into pieces, and see if I can make the larger problem into one or more smaller problems that are easier to figure out. Most tricky software (and hardware) issues boil down to either 1) something I don't clearly understand or 2) multiple things interacting in ways I didn't anticipate.

Do you post class/lecture notes online?

I don't post notes or slides. I don't use slides, and I have found that when I post notes, students disengage (assuming they can get the details later). Unfortunately, when they try to get the details later, there's nobody to clarify, and they don't learn it as well. That said, I don't mind if you post your notes on Piazza.

Is there a way to get involved in your research?

Yes, let me know that you're interested. Show me that you're a strong programmer? Learning a little about hardware can be helpful (not always necessary)? Then if we can find a project that we're both excited about…sure. Very doable.

Does your autograder test for malicious software?

No, but it keeps good records and has some limited protections. I have never had a problem with people trying to hack the autograder. I think most students would have trouble pulling it off without

getting caught. Please don’t try it. You’ll make more work for me, and it will probably not end well for you.

Have you worked in any other fields?

One of the things I love about computing, is that it makes it easy to work in just about any field. In the course of my research, I have worked with medical doctors, biologists, automotive technicians, transportation engineers, civil engineers, and farmers. Does that count? I also worked in landscaping and as the freezer manager for a local grocery store in a former life.