MSU Computer Science Senior Survey 2021-2022

June 7th 2022, 3:35 pm MDT

Q4 - Which subjects(s) or course(s) in the curriculum do you think were the most valuable? Why?

Which subjects(s) or course(s) in the curriculum do you think were the most valuable? Why?
Learning new languages and learning algorithms. most relevant in life.
132 232 246 432 - Classes that teach students to learn how to problems solve using algorithms and learn how to code. 112 - Teaches a language that is vastly important to the industry.
I loved taking graphics, computational topology, advanced algorithms, and CS theory. They were the courses that provoked the most personal growth for me as a computer scientist. I especially loved classes like graphics, which are challenging, taught by passionate professors. and effectively combine the underlying theory and algorithms in a topic with their implementation. This kind of instruction I think is the best to receive as a CS undergraduate.
test
Definitely 232, theory, and the capstone. Respectively, fundamentals, understanding algorithms and mathematical basis, and working with a company.
I think the courses that dived into different languages and why they were designed differently (the C class, Programming Languages) as well as how they are implemented (Compliers) were the most valuable because I think I learned how to use different languages in different applications and how to take full advantage of them.
I think compilers and robotics were the most valuable courses. They were really pushing expanding my understanding and abilities in computer science and pushing my to not be afraid of new and complex things but rather to be confident in attempting them. I also found that the classes focused on a single project throughout the whole semester to be very very helpful as they leveraged learning practical skills and research abilities and were much less punishing than many of the concepts and test based classes. I think that these kinds of classes like Databases and Compilers helped me learn the most
Software Engineering. It was one of the classes that I feel like I use everyday at my job.
CSCI366 It is a practical application of student knowledge.
440 - Database systems, because it felt the most applicable to real world programming. 331 - Web development, also because of the real world applications and freedom to work on a large project of your choice
My computer security class because it's why I wanted to study CS in the first place and what I'm going to grad school for.
Discrete Stuctures and datastructures. While every other class had indepth knowledge, these classes set the foundation for thinking like a computer scientist.
I really enjoyed CSCI451 (comp bio), as it felt like the material was both interesting, how to implement algorithms wasn't an afterthought, and the applications were both obvious & presented well.
AI, computer security
Data Structures, Data Mining, Web Dev Tech. Discussing technologies that are highly relevant today, and understanding the principles involved in higher level applications.
EGEN 310(R): Felt like a real-world experience. Got to work with people of different disciplines and was a lot of fun. ESOF 423: Prof. Clemente Izurieta is probably the best professor I've ever had, he pushed me and my team to

new heights of understanding code. CSCI 305: Prof. Hunter Lloyd made the class a lot of fun, but moreover I learned more about computer languages than in any other class. Even though the languages taught were brief they gave me the confidence to explore more languages. CSCI 491: Dan DeFrance was a fantastic fit for this course, I felt like I actually learned how to code for websites.

Computer Systems - I learned about how the Computer works at a low level. I gained programming practice and experience in Git. Database Systems- I gained programming and debugging skills working with a JDBMS. I learned SQL and database theory concepts. Software Engineering - This class is when I began to see the value of the skills I'd obtained so far. It gave me a sense of the big picture of software development. I was introduced to new vocabulary, tools, and technologies, and learned about how to model software visually with UML.

Over my years at MSU, the most valuable courses that I have taken include AI, Databases, ESOF 423, Data structures, and Computer Security. These classes all had passionate teachers who know the content inside and out. Furthermore within these classes, I found the information to be at a level of hard that required pushing myself to grow and become a better computer scientist. I have already used all the classes above within my private computer science life and am sure they will be useful in the future.

The Capstone class (481) was a definite boon as it gave a hands-on use to much of the stuff we had learned.

The Data Structures and Algorithms courses were the most valuable because the knowledge gained in those classes guide how I write code.

The courses that were most valuable were the more subject specific classes like Data Mining, Systems Administration, etc. These courses not only gave the most valuable information for the real world but actually got me excited about CS.

I would say that Compilers and Databases were definitely two of the most valuable, giving projects to list on resumes. They gave real world application that pretty much the majority of other classes were building up to. Test- driven development seemed to really help the learning process, and also gave real world project application.

The first course that come into my mind is CSCI 132. It provides the idea of object oriented programming, and developed more data structures and simple algorithms based on CSCI127. Another course that were most valuable is CSCI 432. From that class, we knew how to develop advanced algorithms, prove the correctness of an algorithm, and different concepts regarding graph, network etc. for solving more complicated problems. Lastly, the CSCI451 can be one of the most valuable courses for me, since it constructed basic knowledge in computational biology field and I developed my interest in that class.

Data structures 132, 232, Computer Systems 366, Compilers 468, Webdevelopment 331. I believe these were the most impactful to me since data structures covers the core concepts of programming and 366/468 explain the concepts of what's happening in the hardware as well as applying important programming concepts, and webdevelopment introduces working with new frameworks and http requests.

Databases, the cloud portion of 366, Data Mining, Systems Administration, Software Engineering Applications, CSCI 498 All courses taught practical material using modern tools. .

The introductory Python course was great when it was taught by John Paxton. It gave me a very solid place to start building computer science fundamentals. ESOF 322 was the best class I took, with Carson. I learned the most about how to actually write good code in this class.

CSCI 366: Computer Systems - this class gave me the first taste of what working on a real software project is like, it definitely made me a better developer. CSCI 440: Database Systems - this class approached the subject matter as a purely practical curriculum, not a lot of theory I'll never remember. CSCI 347: Data Mining - this class gave me a glimpse into the world of data science, again, from a very practical approach - not heavy on theory. CSCI 351: Systems Administration - this class made me more proficient with Linux CLI, and I use those skills almost every single day.

data structures and algorithms and databases and esof 322 were all helpful and relevant to things I have to do in the real world.

The courses that come to mind when I think of valuable were Data Structures & Algorithms, Software Engineering, Data Mining, and Compilers. Data Structures & Algorithms and Software Engineering were crucial to develop the fundamental skills and techniques that I use. Data Mining and Compilers were just extremely interesting and well planned out courses.

Cybersecurity gives a very good understanding of how development can affect the security of site or software. Databases give the ability to be used in multiple areas and was able to help me understand how to create and use a database. Web Development helped to solidify the principles in databases and other courses which was really helpful for my realization of how much I actually know

Databases - This was very valuable because databases are used in the CS industry on a daily basis. Compilers - I have learned how Java works under the hood and become much more proficient with OO languages. CS Theory (338) - I learned how time complexity works on a greater level and what problems are impossible to solve in polynomial time

I feel that classes dealing with topics that can lead to employment (web development, databases, software development, etc) were the courses which I took the most from. These classes helped me to develop my abilities in subjects that can most efficiently lead to career opportunities in the future.

I believe that the most valuable material was the upper division classes. I believe this is the case because the lower level classes are able to be learned elsewhere, online, or through trial and error if students are diligent enough. The upper division classes is where the material got more interesting and more value as subjects.

CSCI 366 (Computer Systems) and CSCI 440 (Databases) were the most valuable classes that I took. Carson was the professor for both of these course and his format of teaching was the most efficient method of teaching I experienced in college. I loved the course projects in these classes and felt like I truly learned the material.

Both of the data structures and algorithms courses were valuable. The database course and web development courses were valuable too.

Learning the basics of programming via Programming with C, Basic Data Structures and Algorithms, and Joy and Beauty of Data. Databases, Web development and Networking have been the most applicable real world classes.

I think the data structures classes were particularly useful. Knowing the proper data structures and algorithms to use is the bread and butter of computer science. Esof was pretty useful in teaching me UML diagram and how professional software comes to be. 366 and 468 taught me how to use my IDE and debugger very well, in addition to greatly understanding how to produce great code.

I thought that CSCI 232 Data Structures and Algorithms was the most valuable class I took. I gained knowledge on coding that allowed me to complete projects in every other class.

I thought that the first two years were a pretty good base for all of the other electives that I took. 232 and ethics specifically I thought were very valuable. 232 was the first time that I felt the coding was very challenging and so I definitely grew a lot as a coder during that class. For ethics I really enjoyed and I thought that it was important to learn about all of the "power" that developers have and how you really can be a positive or a negative contributor.

CSCI 112 and Admin Systems. These were the most applicable to my professional work using the command line. Also Industry Methods seminar, Devin Gray gave important details to software engineering that all other CS courses blissfully ignore

As someone who came into the department with no coding experience, the Programming courses were all essential to the field. Any course subjects covering common industry tools/practices (programming IDE's, version control, supplemental software, etc.) seemed especially valuable as well. The best things I learned throughout the curriculum were how to troubleshoot or research software problems on my own, through Google, Programming Forums, or navigating APIs/other documentation.

I think it was great to get a solid background in theory. I was unable to take OS when Travis left so the security course was helpful for a basic background.

I found both Human-Computer Interactions and User Interface Design to be extremely valuable as they focused more on the users who would be using the software than the software itself.

The Compilers course was one of the most valuable courses I took. Carson, helped me gain skills that are indispensable. The AI course with Dr. Sheppard was another course that was extremely valuable. It was an insane amount of work (~40+ hrs/wk) but it helped me become much better at programming efficiently and intelligently. A course that inspired me was CS Theory, specifically with Sean Yaw. He packed an incomprehensible amount of energy and enthusiasm into his lectures that brought the material to life. Finally, the last course I would like to mention is the Joy and Beauty of Data. When I started that course I was enrolled as a BS in Electrical Engineering.

That course caused me to realize that what I was really interested in was Computer Science. Not, a month after the end of the semester I switched to CS.

I found that CSCI 132, 232, and 305 provided a great basis for the projects that I would later do. That being said, CSCI 466, 446, and 455 really put those skills to the test to prove that I am a computer scientist capable of understanding the way computers work and manipulating them to serve society. The way that Charlotte, whose last name I don't remember taught WRIT 221 was also critical to being ready for the way real companies work.

The Math and Theory courses seems particularly important to me. It is all well and good to be able to write some for loops but understanding the concepts behind some of the more complicated concepts is incredibly valuable. CSCI 338was very challenging but also very useful for classes like compilers and Ai.

Compilers (CSCI 468) - This class really gave insight into how programming languages work deep down inside. It also required learning and using a lot of debugging techniques which will help solve problems in code later on. Software Engineering (ESOF 322) - This course taught how to design programming systems in an efficient manner. Computer Systems (CSCI 366) - Seeing the inner workings of computers and a little on the computer engineering side of things for hardware gives some additional insight into how computers work and how to code for that. Concepts/Programming Languages (CSCI 305) - It was helpful to see many different programming languages and getting a simple understanding of them. Having a broad understanding of what can be done in certain languages will help to decide what languages are best suited for what tasks.

I would say that CSCI 132, CSCI 440, and ESOF 322 are the most valuable courses as they provide a strong foundation in object oriented programming, database management, and the process of software engineering, which are all extremely relevant and important topics in the computer science industry.

CSCI 331, CSCI 440, ESOF 322, ESOF 423, and Industry Methods, these classes provided insight into the industry and what tools and skills are necessary to succeed.

intro coding courses like c and python were incredibly valuable since they were the building block for the rest of my time here. Web development and software engineering courses were valuable because they had challenging semester-long projects that forced me to learn new tools that will be used in the real world (GitHub, etc.)

Required: 246 - provides a strong mathematical foundation for CS topics. 232 - provides useful algorithms and data structures, fun implementations. 366 - Provides the HOW for how computing works, which is pretty fundamental to understanding memory and computer operations (plus Carson is great). Not required - M221 - Linear algebra is so important to everything, especially in CS. M441 (numerical linear algebra) super important because it talks about linear algebra and is also the only course that has EVER talked about numerical precision and error which is fundamental to computing. The rest is just elective, they can all be useful depending on who you are. I like CSCI 446, 447 as they provide a good foundation for machine learning (hot topic). CSCI 466 is also good as it is unique. Grad courses that obviously not everyone can take but I loved: CSCI534, M507, M508 are some of my favorite classes I have ever taken.

107, 127, 132, 232, 112. These were the most valuable as these classes was where i learned how to write code. I had never written code before college.

Ethics. Maryann is wonderful

I thought databases, UI/UX, and Concepts of Programming Languages were some of the most valuable classes offered in the CS program. Since I've started working in the field my junior year, I've noticed a lot of backend developers fail to consider how users interact with the software that developers create. This can create a lot of issues if the company doesn't have a designated UI/UX person, so having a basic understanding of this seems rather important. Databases are something that every company has to use, where on-premises or in the cloud. Knowing how to navigate those databases is very important so that as a developer, you can find the data you need to call in your code. Concepts of Programming Languages covered a very wide variety of languages, which I think is very important. This allows students to explore different languages they may want to consider moving forward with in their career. Exposure to more languages also makes it easier to pick up new languages in the future.

The courses I took that challenged my programming ability and made me practice coding solutions to problems were the most valuable. They are valuable to me because this is what I anticipate I will be doing for the majority

of the time during my career. I think practical experience and knowledge often trumps theoretical knowledge and this is also what I enjoy studying.

Any course that could direct more of career path in computer science. Such as learning something more applied such as Computer Security or Networking to determine what area of computer science interests me.

The three courses that I found to be the most useful are CSCI351, CSCI232, CSCI440. The basic algorithm classes were incredibly helpful and useful. The databases and system admin classes were very practical and extremely pertinent to my future endeavors.

By far the class I took the most away from was System Administration. I understand that this class was minimal in the programming and problem-solving area of CS, but I think it should be a requirement. Get EVERYONE on Linux ASAP. Additionally, 305 concepts of programming langs. was a super valuable class. I find it applicable in almost every other class and I should have taken it much earlier on.

Most classes relating to tools in the field (such as 322, 440 and 466) were very valuable to me just because they introduced me to some of the real world applications for the major. I also think 366 and 468 were valuable just because of the in depth explanation of systems and how they're made.

The intro courses (like CS 111, 132, 112) are obviously valuable since they are the basic building blocks that the rest of everyone's coding journey are built on. I'm in the Industry Methods seminar which I think is also incredibly valuable, but I think it should be pushed for students to take this earlier (like year 2) since git is widely used in a lot of upper division courses. Databases was super helpful because data is everywhere. Systems Administration was fantastic for learning and becoming comfortable with the command line. And Concepts of Programming Languages was super helpful to begin to experience programming languages that may be older or different than ones we normally use.

Computer Systems and Databases because they made me an all around better coder and also taught me how to use debugging tools. All the other classes that are upper division electives were valuable because they allowed me to take wide array of subjects and it really helped me find what I enjoyed most when I was coding.

Computer Security and system admin. both of these coursed provided the most real world examples and application.

Any of the courses taught by Carson Gross. The real-life relations and perspective were invaluable in my education because he instilled very important concepts that any of the other professors failed to recognize as important enough to teach.

All of the courses that Carson Gross taught, I not only learned well from his teaching style but the life advice/concepts he taught were the best things that I learned during my time at MSU.

CSCI 112 and CSCI 468 because they cover the most basic concepts (and also the most in depth) needed for me to understand how an IDE interprets code and how to effectively write code.

Computer Security, System Admin, 494 Industry Methods, Software Engineering were the classes that gave me the best 'bang for my buck'. Computer security introduced me to a whole new area of computer science that I felt could be applied to almost any career in computing. I almost wish it was a mandatory class and that there were other classes on similar topics. System Admin gave me my first introduction to Linux and a whole bunch of other tools that have come in handy throughout the years. The Industry Methods seminar I think needs to be a requirement for all CS students. It gives a brief overview of a lot of popular tools and methods used in actual workplaces and its goal is to help prepare you to work in a professional environment. Lastly Software Engineering (with Carson Gross) was one of the best classes I have taken. It taught me so many useful skills that I am honestly mad that it is not required to be taken earlier. I think you could easily have students in 132 also take a sort of software engineering class and that would help them immensely. It also helped me answer the question of 'why am I a CS student?'. The first few years of my degree I could write code and solve problems, but I saw no real application for what I was doing. Software engineering showed me a lot of the applications that a CS student can have.

Software engineering course were helpful over the years, they are incredibly helpful when entering the capstone courses. The Java courses in data structures was helpful as well when I started interviewing with companies. Most companies had questions from data structures courses. Requiring rigorous math courses allowed me to understand the data structure course at a quicker rate.

CSCI 232 - very important basis for data structures and algorithms CSCI 432 - helpful expansion on CSCI 232 CSCI 466/ CSCI 366 - Coding projects strengthened skills, useful topic CSCI 338/CSCI 246 - helpful theory

CSCI 351, CSCI 476 (Computer Security), and CSCI 440. CSCI 351 and CSCI 476 really helped me feel confident in using the command line. In addition, CSCI 476 was really great course to take because it introduced me to old exploits and existing ways to counter those exploits. The class really renewed my interest in computer science and cyber security. CSCI 440 was an amazing class because of how applicable it is to the job field.

CSCI 232 Data Structures and Algorithms - Every software engineering interview I had asked questions from content in this class. CSCI 215 Social and Ethical Issues in CS - Super interesting topics and an important side of computer science to remember as we create.

ESOF 322 and CSCI 366 because they solidify the fundamentals of computer science

I think that data structures and algorithms, discrete mathematics, and web development were the most valuable courses that I took. Data structures and algorithms gives a necessary foundation to computer science, while discrete mathematics offers a deeper look at some of the logic present. I found web development to be a hands-on and applicable course. This was what seemed like the first time I got to create real-world applications using the knowledge I learned in previous courses. I learned about many platforms and softwares to help with development and got to work in a group scenario that allowed me to experience what working in a development team would be like.

I really enjoyed and learned a lot from AI and ML. This could be because John Sheppard is a really good teacher who knows his stuff but also I think that doing 4 large projects and nothing else over the semester just kick started my learning process into high gear. Writing everything from scratch and having a month to do so is great, and it also allowed me to learn and use python instead of java.

CSCI 132 and 232 - These two courses I feel cover the most important knowledge a programmer needs to make a working program. It discusses efficiency, something many coders today sometimes do not care about given the speed of modern computers. Databases / Computer Systems - These two courses have the most field, practical knowledge you need in many jobs... EGEN310 / ESOF322 - Very important knowledge for working with a team or with other programmers, how to work with others. Programming with C - Absolutely critical because it is used everywhere except Microsoft and Apple. Any electronic work will be in C pretty much, and mostly everything in Linux uses C in some way.

CSCI 366, CSCI 132 & 232, Technical Writing, Ethics. To me these were the most valuable because they provided a fundamental explanation of important techniques and topics in computer science (132 & 232) and overall context for how computers allow for coding to happen (366). I also found technical writing and ethics valuable because they taught about softer skills that you would employ in your career that don't get taught in other classes.

CSCI 440, CSCI 476, CSCI 446, CSCI 366, : These classes were all very practical and taught useful skills and practices.

Data Structures, Algorithms,

Computer Security and Web Development

C++ This is my favorite field. The LLVM project is so successful that it is difficult to find a popular language that neither uses the LLVM library to write the front-end nor the LLVM to generate bytecode, and these are all C++.

CSCI 232. Literally all of my interview questions are on things I learned in that class.

Overall, I believe all courses are valuable to provide insights to what it is doing.

I think the intro classes such as 127, 132 and 232 were the most valuable as they introduced the basics of coding while also introducing important algorithm topics like sorting algorithms and red black trees. Along with these intro classes I also thought personally that the UI Design class was the most beneficial as it was my realization class for what I would love to do after I graduate MSU, but everyone is going to have their own class with this experience as well.

Most of the core classes did a good job laying the necessary groundwork and developing the understanding needed to succeed in more advanced and interesting classes. I thought CSCI 442 Robot Vision was a particular stand out because of all of the concepts that came together in it and how well it was presented. CSCI 338 was also

fantastic, and really helped me understand a lot of the underlying math and theory behind computer science. Overall though, I found value in almost all of my classes.

C++ Any software developed with C++ can be well controlled in terms of timeliness, stability, and scalability. This is not achieved by any high-level language.

CSCI 132, 215,232 246,305, 338 and ESOF 331. These teach the basics of computer science and are the foundation of what goes on in the other classes. If these are not taught well then the other classes can become cumbersome or overwhelming.

ESOF 322: Being able to design software using UML is extremely important. This course also let us dive into the different development styles that are used in the field: Agile, Kanban, etc. CSCI 468: Understand a coding language all the way down to machine code really puts things in perspective. After this course I just felt like I had a strong understanding of everything that made a coding language a coding language. CSCI 305: This course allowed us to dive into several new coding language. Some were even new and super popular. This course allowed me to practice switching languages. The ability to understand the changes in syntax between coding languages just makes you a better programmer all around. CSCI 442, CSCI 446, CSCI 447: this computer vision course is super relevant to modern technological advancements. Computer vision and artificial intelligence are becoming more and more important when developing software. Machine learning is just another added topic that is growing in our field.

I thought the basic data structures and algorithms and data structures and algorithms courses were the most valuable because they taught us how to use various data structures that are very important to know for more advanced courses. I also thought the social and ethical issues in CS was very valuable because although it was not a programming class, it was very helpful learning the ethics behind designing and implementing software.

CSCI 132, CSCI 232 and EGEN 310R. these three courses all had good hands on approaches to programming that i learned a lot from.

Any of the courses that covered the intro into different languages, i.e. python or c. The ideas from these classes built a base for everything else. CS theory is also a course that comes to mind that had lots of important ideas that I kept seeing throughout my time in the CS program. System administration also introduced me to lot's of key concepts that always seemed like we were expected to know but never discussed in depth, mainly linux and various commands that were very useful.

Data Structures and Algorithms because they introduce fundamental CS concepts and good programming practice. Operating systems were really helpful for me in understanding how a computer works, a lot of which helped me as a programmer (i.e understanding heap/stack/memory allocation, program lifecycle, concurrency). ESOF 322 with Clem was a great intro software design patterns. I also got a lot of value from Advanced algorithms, AI, and ML because they are topics I am interested in pursuing in graduate school and my career.

Ux Design and Human Computer Interaction were two of my favorite because rather than just coding, it taught us how to think about products and how that relates to our coding. These I think were two of the most helpful, because most CS majors can coding and think logically, but that doesn't necessarily mean we can think like others and make products that are useful for them. Also the class with Hunter about all the different coding languages was very helpful because it taught us to see the different uses of different languages, and take the same principles and bring it to lots of different languages.

I think that the most valuable subject are principles of designing a software system like what is taught in ESOF and general data structures and algorithms. I also think CSCI 440 does a great job teaching the concepts of databases as well as connecting a web page to them.

Databases, Systems Administration and Computer Systems all felt very helpful and relevant to what I want to do with my degree.

All of the programming classes. C programming, Java, all of the classes that Carson teaches. The reason why I said anything that Carson teaches, is because he uses big projects. These are the ideal sort of things that you will be seeing in industry. This gave so much more perspective on how to actually code for the career.

csci 232, csci 466, csci 460. 232 is just super useful for any career path in software. 466 exposes you to a lot of internet protocols that are very useful to know in general for whatever you end up doing (eg. web dev, api dev, etc.). Both 466 and 460 are also great at presenting you with projects that use multi-threading, which is important to conceptually understand for real-world applications.

CSCI 232, CSCI 466, CSCI 460. 232 because it's essential for problem solving and all paths in CS. Networks and OS were very valuable because they give you hands on experience dealing with multi threaded programs and concurrency. Also basic network concepts are essential in many fields (eg. web dev, API dev, etc.)

Any of the classes that had coding as part of the work done. I don't particularly want to design new computers and chips but just write code.

ESOF 423 - I thought this course was the most valuable because we had time as a group to dive down into a project from creation and planning and following through with development and testing. This course was fantastic because it forced us to use what we'd learned in previous courses but also forced us to grow and evolve our skills as unexpected challenges exclusive to the project appeared. I loved this course because it was challenging in very good ways and was an excellent motivator to research and grow our knowledge of tools that would be best to use for our project.

I use the command-line stuff from Systems Administration CSCI 351 all the time. I also really appreciated Data Mining for diving into interesting data topics with Python (which I hadn't used since 127). Carson's Database Systems class is a riot and I'm already using SQL at my job (which is NOT a CS job) to write queries, and his database fundamentals are enabling me to build a database for my final project. All three courses taught different skills but all are extremely useful, even when I'm not doing a software-building job.

The interdisciplinary capstone was amazing. I've always wanted that level of freedom to pitch my own ideas and make my project relevant to myself and my minor! This was especially important because the CS program is sometimes described like a factory. Compilers sounds very much like that from what I've heard. Concepts of Programming Languages with Isaac Griffith. I've been using Java from the start (just before you started teaching python), but his project finally made everything click, with why you'd ever actually want to use objects and inheritance, and it's become my favorite language. I actually use all the languages we learned in that class (except for ML): Java, Ruby, and Prolog! Each has its own use case and I love them all. It was so important to see what all was out there and learn how to use some specific tools properly, which is something this degree lacks by design. Artificial Intelligence with Maryanne Cummings. I had Sheppard for machine learning later, and I'm honestly very glad I took AI the one semester she taught it instead. He's clearly brilliant! I just don't think his teaching style works well with me personally. I was so damn proud of my final Wumpus World project and report with my little MS paint graphics I made to represent my agent's actions. I still think about and use the things we covered. Computer Architecture wasn't perfect, but it taught me a small handful of VERY important things nobody else seemed to even touch on. -Kilobyte KB vs Kibibyte KiB. Incredibly important as I named my first ever pet Kibibyte that semester. -How floating-point numbers are actually stored! Yeah, everyone knows there's rounding errors, but why? Also, how signed and unsigned numbers work. -That assembly is even a thing at all; that it depends on processor architecture and that compilers are saintly for allowing me to not worry about stuff like that. Finally, CS Theory with Sean Yaw. Even if I don't become a scientist, that's kind of what this major is about, right? And I struggled a lot! Once again, incredibly grateful I had Yaw, otherwise I don't think I would have made it. I'm no stranger to being a bad student. But his explanations and lectures made a ton of sense, and he worked with me and inspired me to keep working at it. Even though I was failing things in the middle, I finished strong feeling like I had caught up, learned everything we covered, and somehow came out with a B+.

I feel the content covered in data structures and algorithms, computer systems, and operating systems were the most valuable, as most of the content is what I found in my technical interviews, as well as what I used in my internship.

I am currently working as a software engineer for Oracle in C++. So 112, 127, 232, 366 and 468 have been the most directly valuable for me. Essentially all the classes that directly involve writing source code.

CSCI 232 and CSCI 366. Hands down. 232 prepared me for job interviews and was the moment where everything I had learned up to then clicked, and i felt as if i had the tools to be able to tackle any problem. 366 was valuable because it exposed me to a lot of new and different technologies, and the project we did challenged us but was very doable.

Computer Vision, Compilers, CSCI 232, CSCI 440, and CSCI 366. Up until taking the computer vision course I had no idea what I wanted to do with my career. After taking that career I immediately found a position I love and give Lloyd's course 100% credit for getting me there. I found the other courses: CSCI 468, 232, 440, and 366 to be

extremely valuable as they were the most practical classes. While I have an interest in going to grad school and beyond, these courses provided hands on learning to technology we only ever hear about before.

Web Development courses – Every internship that I've had throughout my undergraduate degree utilized concepts and skills from web development. It is obvious that most of the industry is heading in this direction so it is an incredibly valuable class subject. Introduction courses -- This is where I really decided that Computer Science was the path I wanted to take. I felt that doing the labs for these classes really got me interested in pursuing CS as a degree. Computer Security (CSCI 476) – I believe that this should be a required class for the curriculum. Understanding even just basic computer security concepts should be a minimum requirement for any graduating CS student. I went into that class with little-to-no knowledge about cybersecurity and left feeling like I had a much greater understanding. I feel that what I learned from this course will help me to develop with much better cybersecurity practices in the future. UX Design (CSCI 494) – This was one of my favorite courses from the entire curriculum. The deliverables that were required for this class are projects that I proudly display on my own portfolio (which was also created as a deliverable for this class). I loved getting to see a different side of CS and UX Design is now something that I would like to consider pursuing post-graduation.

Q5 - Which subject(s) or course(s) were the least valuable? Why?

Which subject(s) or course(s) were the least valuable? Why?
my technical writing class was a joke and I didn't even learn anything relevant.
None, all required courses are important for this industry
Programming languages, technical writing, and EGEN 310, and software engineering I thought were the least valuable. The biggest problem with all of the mentioned coursework is that it felt like it was just checking off a box in our educational journey, rather than something that was done strictly to benefit us as computer scientists. I think technical writing should be waived by another writing class if desired (eg. texts & critics), programming languages should focus a bit more on the actual structure of different languages (which is super interesting!), and EGEN310 and software engineering should be more dictated by projects that are student chosen and led, rather than making some meaningless arbitrary thing, if possible.
test
SysAdmin was pretty useless if you already spent the time to learn how to navigate Linux, but I guess that's with any other subject too. This one's pretty much just for me.
I think the CS theory class was the one that felt like the most waste of time because it felt way too academic and/or too historical. I also think architecture, while interesting, wasn't very useful to me as a high level developer.
I think classes focused on using tests as a measure of learning metrics were a big issue. They were very punishing for topics that in real work you would be looking up the niche concepts and question ideas as you go to make sure you are correct. I would not apply this to the strictly theory classes necessarily as you have to be able to have a measurable metric but i believe that any class based on being able to design or develop software should focus on your ability to develop it with the resources and tools you would be able to have in a work environment. Focusing on actual projects rather than odd and confusing quiz and test questions.
Proofs. Not applicable to what I do.
CSCI 338 It is a class that has merit in academia but not much applies to a desk job.
338 - CS Theory, because it does not seem applicable to many real world scenarios and I think the most important aspects were covered already in 246 - Discrete Structures
I don't feel that I got much out of my technical writing class, mostly because it was interrupted by Covid.
Just my extra curriculars. All of the cs classes were good.
EGEN310 was completely pointless in my opinion, though it may have been due to the covid structure of the course. Additionally, CSCI215 I felt I didn't learn anything in, as it was all what I would think to be pretty self explanatory content, though I understand why it would be required in a curriculum.
anything taught by hunter lloyd, daniel defrance, or millman
Artificial Intelligence: the heavy focus on group projects made an already difficult subject even more difficult. Students are not typically experts at working effectively with a collaborative tool like Github. Make it a requirement, or touch on it in the coursework.
CSCI 246: The professor I had (blanking on the name), was rude, sexist, and a complete failure as a professor. I still don't remember anything that I learned in that class or why discrete structures even matter. I have never been so embarrassed to be a MSU student than when I was in his class. I heard from someone recently that he was let go due to complaints which is awesome. If he hasn't, I would highly recommend he be removed from all classrooms ASAP, I don't remember his name but I took it in Spring of 2018. PLEASE GET RID OF HIM. CSCI 145RA: Had a lot of

potential and I could see it being a very helpful class, but I don't feel as though I learned all that much about web design, it's supposed to be an intro class but felt much too easy compared to the other intro-level classes.

Calc II - I haven't found it applicable to other things I do

The course that I thought was least valuable was EGEN 310R. This class was extremely frustrating as there are not enough electrical engineers in it thus the computer science student is tasked with figuring out a lot of things that really aren't things that are taught to a CS major unless they have a strong personal passion for robotics or embedded systems. Furthermore, this class will be even more challenging for those that chose to take it as it will reduce the rare chance of having both a CS and EE in your group. The teachers are not helpful in getting the actual content needed out to students to complete the final project and it felt like a third grade class being taught how to analytically watch a movie again. To make matters worse while being tasked to watch a movie or read the script of a movie, the over arching rover project continues to creep up on students with no mention from professors until about 2/3 of the way done with the class. Overall, it felt as those I as a CS student was tasked with redesigning a product similar to what we worked with in CSCI-455. I know that this class is currently being transitioned out of the CS required program but I truly think it has the potential to be a great class but each team should be provided at least one of following engineers(EE, ME, CS, ChE or at least provide some sort of help sessions!

Discrete Structures was the least valuable class not because of the provided information but because of the way it was taught. Brittany Fasy had very little idea of what she was doing and the class was a mess. I ended up taking it again because none of the info from that class was retained. Computer Systems also had the same issues, though less extreme, again the information the class provided was very weak.

I think that the Concepts of Programming languages course was likely the least valuable because the integral topics about programming languages that I learned from the class were taught to me again in Compilers.

Much of the early classes (132, 232, 112) left me learning much about actual programming to myself. Many of these classes gave me the frustrating experience of learning the hardest way of doing things before finding out the best way.

I would say I was not a huge fan of theory courses, it seemed that those classes did not need a full class dedicated to them, as the things we were learning usually had been covered in other classes at some level. The same is true for real world application that if theory is what you were going into, the class was needed, but if not it seemed like information that was easy to forget.

The least valuable course to me is CSCI112, since I didn't use C language in programming after taking that class, and the concepts from that course are already covered by CSCI127 and CSCI132.

Computer Ethics 215, Programming languages 305. I did not take anything valuable away from these classes. There weren't any new concepts introduced to me other than new syntax of languages.

Embedded Systems - focused entirely on writing python scripts for raspberry pi, didn't actually learn practical embedded software skills Compilers - I understand its a national requirement but really doesn't seem like practical knowledge ESOF 422 - Spent majority of time learning/navigating very niche proprietary software EGEN 310 - Spent most my time teaching myself poor electrical engineering. CSCI-338 - Only relevant for people getting into theory, not much that transferred to realworld.

Ethics in Computer Science *could* be a good class, but the current implementation has a lot of fluff and largely unnecessary sections that make the class really unpleasant and useless.

CSCI 460: Operating Systems - this class was super high level with a lot of theory and I had a hard time understanding how the programming assignments were reflective of what we learned. CSCI 455: Embedded Systems, Robotics - this class taught me nothing and was needlessly stressful with unclear objectives that did not come together in any sort of meaningful way that made me feel the class was worthwhile. CSCI 338: CS Theory - yes, I understand it's a theory class. But I can't see why it's a required course seeing as how the content taught in it is almost never utilized throughout the rest of the CS program.

I don't think any weren't valuable.

The first course that comes to mind was Social & Ethics. This course was very straight forward and felt almost like a filler class. It could of easily been implemented into another course like Software Engineering or something. The

other course, though not required, that I took that I felt wasn't very valuable was Systems Administration. This class was pretty much the Linux for Dummys manual and it felt more like a 100 level course than a 300 level course.

Concepts of Computer Programming - I was only able get a small taste of languages and did not give me enough to really be able to find any useful

Discrete structures - I did not learn anything from this class, could have been bad teaching but was completely useless to me. The way that Java was taught did not seem to be a good introduction. Many people struggled with this especially compared to my peers at other schools. I think there is a better way to go about introducing students to OO languages then trial by fire.

The courses I felt were least valuable were not necessarily courses I did not enjoy, but overall courses revolving around theory or algorithms felt as though they would not be applicable much in the future. They were some of my favorite classes and I definitely found them interesting, though it seems unlikely I will need to carry much from them forward into my future.

The lower level classes because the material covered in them can easily be learned and found elsewhere with little deviation in knowledge

CSCI 432 and CSCI 232 were the least valuable classes in my opinion. The curriculum in these classes was taught so quickly and not nearly in-depth enough. Fasy taught 432 and felt like I barely made it through the class. Every week a new algorithm topic was taught and the weekly homework assignments were often so in-depth that I felt like I didn't understand anything. CSCI232 was similar in the sense that a lot of complex data structures were taught very quickly. Granted I took CSCI232 Spring of 2022, so the pandemic really affecting the teaching. However, after only one semester worth of Java experience, I felt like the class was extremely fast paced and I didn't have the experience or skill set to complete assignments and understand the material.

The security course wasn't super valuable. I'm glad I took it but I left the class feeling like I didn't have much of an arsenal to defend against hackers. Also, programming languages wasn't very valuable.

Computer vision was a sub par course. We didn't go in-depth into content. I also had to self teach myself everything because the instructor was sub par. Technical writing I thought was just busy work and not thought provoking.

Any of the math classes that we had to take were a complete joke. That is Calc 1, Cal 2, and Linear Algebra. While its fun to sit there and just memorize stuff that we have built in functions for it really isn't all that fun. There is no reason we should be memorizing formulas in this day and age. Instead of learning to be thoughtless students that just memorize stuff they should teach us practical application of the mathematics. For example, a student can go through data mining without taking linear algebra. It's more important they understand what they are doing and how to use the mathematics to solve problems rather than just memorizing the equations. Generally, most computer scientists do not use mathematics in their field. I would rather have 2 more coding classes than learning how to compute derivatives. I would like to have another data mining class instead of linear algebra.

I thought Egen 310R multi disciplinary engineering was the least valuable course. I unfortunately was not paired up with a electrical engineer like every other group and the project partners did not have any way for me to contribute to the final.

Discrete structures, I didn't really see the application of that class as much in my other classes

Calculus, haven't used it yet. Also all my core classes, nothing is applicable. Although some classes were good for generic knowledge, the workload conflicted with CS courses that were more important. Data mining has been only confusing, nothing seems to translate to something actually applicable.

While I didn't have the theoretical CS classes, it was at times difficult to see the value of super theoretical subjects. I did find these interesting and helpful at times, so I wouldn't call for these to be removed from the program, but they could potentially be paired down.

Ethics - the course could be super interesting but the class fails to address current ethical issues in a meaningful way. It would also be better taught as a seminar type course similar to honors college courses. 112 -

EGEN 310 largely felt like a waste of time, as there were several other classes that had groups of students working together on a project. Additionally, it kind of felt like the only reason the CS students were there was to make

everyone's project work, as opposed to being a contributive member. CSCI 305 also seems largely redundant as much of at least the conceptual stuff is covered in other classes such as theory and compilers.

I'd argue that I didn't take a class that was less valuable than another. They were all valuable in their own way. However, if I must choose CSCI204 was not particularly useful to my professional development.

I found EGEN 310 to be worthless. I was told to learn a new field and implement that. I didn't use CS at all, and only used what I learned in WRIT 221. The new curriculum removed 310 though. CS 145RA was too easy to be useful. We covered the content again in 331 in 2 weeks.

Unfortunately anything that had us working with C is of lesser value. C is not as commonly used now and I only see it being pushed further in the background in the years to come.

N/A

The least valuable course I took would probably be CSCI 130, Programming in C. I found that with the way the class was taught using extremely old technologies was not very relevant to any other course or subject I came across. It could simply mean that I haven't encountered a use case for ideas from the class and it does still have some value.

N/A

Least valuable courses were social and ethical issues, web design, and discrete structures because for the most part they didn't cover real world concrete skills that transfer over to like as a software developer.

I am torn on 338 it is very useful in learning what even is computable but I don't think that most CS students have the math background to fully appreciate it. 112 was also not that useful, it was just trivial and slow.

I didn't find much use in social and ethical issues. Nor did i find it very important to take calculus and statistics. Statistics should be replaced with data mining as a requirement. There are similarity's but data mining is more computer oriented.

CSCI 338

Administrative systems and some of the options for the 13 credits of math and science. Administrative systems offers great understanding of the command line general Linux based operating systems, however I don't see how this is class will be helpful to me in the future. It was very fun to learn how to use the command line, thought I feel a more practical application of the commands learned would make the class more useful. Multidisciplinary engineering design really only taught me the cycle of which things are designed, which is covered in the UI/UX class. Computer science students are also very often pushed to do electrical engineering student tasks when those students aren't present in the group. I did not feel I learned anything valuable from this class and it was a waste of my time. I think it should be replaced with a mobile development class.

The software engineering courses were not very valuable to me. I think much of that information can be learned in an easier format online, and I think the course was taught poorly (however this was during 2020 COVID times so I can't put too much blame on the course format). I struggle to remember the concepts in that course related to UML diagrams, etc.

I found more theory focused courses such as Computer Science Theory or AI to not have much application but they may play more of a role in graduate school.

There were a few classes in particular that I found to not be particularly helpful, namely CSCI215, EGEN310, and ESOF322. The social and ethical issues in CS class seemed unnecessary and not super useful. EGEN310 was pretty stressful and I did not learn much from the experience. The content in the software engineering class seemed pretty irrelevant to the average graduate. I don't know how much ER diagrams and class diagrams are used in most companies these days.

310R - Which has just been removed from the curriculum... thank goodness. What a joke.

EGEN 310 wasn't very valuable for CS specifically due to the limited interaction and large amount of knowledge we aren't taught for the subject matter. I also think 215 wasn't terribly necessary as a full course. Maybe if it was integrated with another course, but on its own it felt like a waste of time.

Computer Science Theory I think was the least valuable, but that's probably because it had been a while since I had taken Discrete Structures so I was trying to relearn old concepts and learn new concepts at the same time

The only course that I don't think was very valuable was calculus 2 because I haven't needed to use nearly all of the topic covered in that class since the class. The only useful part of calculus 2 for me was the summations but all the other topics were not useful to me.

Egen 310, I took during covid semesters everything felt like busy work for the sake of busy work.

A full semester of computer ethics was a complete waste of time not because the course was boring or long drawn out but the fact that when out in the field the code of ethics comes down to business practice and personal morals. In other words, it was a pointless waste of time because of how blatantly obvious the content was the majority of the time.

The ethics course, because it turned into busy work, and a full semester on it was a waste of time and I learned nothing valuable after a week of lectures because it became redundant.

CSCI 215 because it became a political discussion rather than a discussion about todays current CS environment

CSCI 305 and EGEN 310R are the 2 courses I have always thought were not needed. 305 would be a substantially better course if it focused on relevant programming languages and how to pick them up in a short amount of time. I just felt that using the older and more obscure languages did nothing for me academically. 310R is a class that I felt was a joke. When I took it the 'group project' was decided by each group and it seemed that you could choose a project with very little work if you wanted to. My group chose a project with a relatively heavy amount of work and almost all of it fell on myself and the electrical engineer in the group. I think to fix this class there should be a very specific goal in mind that fully incorporates each discipline equally. I also think that professors from each discipline should be involved. How does a general engineering professor know what good code looks like or how to help a student in a discipline that is not their own?

CSCI 127 was nice for getting use to coding but there should have been more focus on data structure and how to use GitHub. Upper division course are difficult without having Git instruction in the earlier courses. Discrete structures also didn't seem extremely helpful in my later course, but this could be due to the fact that I took this course during Spring 2020 when the pandemic started. The C course was unhelpful, was able to complete CSCI 366 without it.

ESOF 322 - Topics touched on are rarely used in the professional world CSCI 215CS - Failed to touch on the current ethical issues in CS CSCI 305 - did not learn about how languages are structured, just learned about random languages that anyone could pick up on their own time

CSCI 338 didn't really seem all that applicable to anything that I would do. Learning about how to prove that something is NP-Complete just didn't seem all that helpful. Personally, I think CSCI 338 should be a recommended elective but not a required course. That said, the professor I had, Sean Yaw, did a really great job breaking down the more difficult aspects of the class and made them much easier to understand. He did a good job trying to make the course interesting and engaging.

CSCI 246 Discrete Structures - I don't remember anything I learned in that class and haven't needed to think about it again. If it weren't for complaints for underclassmen I think I would have forgotten the class existed.

Theory because it has little connection to applications in the modern world. It's more of a history class with obscure math

I strongly disliked calculous. I find it interesting that calc 1 and 2 are required for this major. I understand that knowing the math behind some of the functions we use is essential, but personally I do not think I will use this in my career.

EGEN310 is a useless course and I do not understand why I was forced to pay money to take it. It is one of those things that look great on paper - teaching students how to work in a job setting and learn leadership - but it does NOT pan out that way. It is somewhat akin to high school courses: busy work, forced leadership roles that do nothing, and a skewed view of how industry operates. Now, the rover project was quite cool and I think a CS directed instance of this class would work well in the curriculum. Also, I think Ethics is a very important class but it was bungled the semester I took it - could be better.

College Writing and Technical Writing could very well be combined into a single course. It seems too much is focused on writing a paper, only a single semester seems necessary. The art of making essays is very different

from writing used for CS courses and still very different from research papers within the CS world (I have read a good bit of them).

To me discrete structures and computer science theory were valuable to an extent, but it felt like they were based so much in the theory that it was hard to see how the concepts would actually be applied to programming tasks. If there had been some more hands on examples with the theory explained I think it would have been more valuable.

EGEN 310: I did not feel like this class was relevant to a career in CS. To make it relevant, the project should be able to focus more on CS. WRIT 221: This was also extremely irrelevant to practically all aspects of a career in computer science. The projects, such as the creation of a trifold brochure or an advertisement, were not helpful to me. The only practical aspect of this course was the creation of a resume.

EGEN 310, CSCI 361, CSCI 246

Ethics for Computer Science

None Every class has its necessity and value

M 172. I really don't see how I would need to use any of the materials I learned in that class in my field given that I won't be going to grad school.

I do not think there is any course that were least valuable.

Personally I found the Computer Science Theory and Networks classes to be the least valuable in my experience. Theory was poorly taught even though I can tell that Binhai knows the material very well. It is an important topic to be sure for those who are pursuing an academic related path in computer science but I found myself struggling with relating the material to what I would actually be using computer science for in the field most days. Networks was a similar class where I went in expecting to work on projects and assignments that were related to what was being taught in lecture to help solidify my understanding of the material. Unfortunately this was not the case and most times I found myself having to spend most of my time researching what was asked of us in the programs as they had hardly anything to do with the lectures themselves.

In terms of CS learning value, EGEN 310 was the least valuable. I learned a lot about teamwork, and a fair amount about building a remote control system, but I could've learned significantly more if the class didn't have all of the annoying hoops to jump through. CSCI 246 Discrete Structures was quite frustrating for me, but that was because I didn't seem to mesh with Professor Fasy's teaching style.

None Every course has its necessity.

CSCI 351 or any of the EELE. The CSCI course felt out of place and the EELE classes kind of go into what we do throughout our CS courses are about, but its like comparing apples to oranges. The EELE courses serve no real use for what we do in coding.

CSCI 204: This multimedia dev methods course which is just a Unity Game Development course is poorly run. Often times coding in this class results in just following online tutorials and then modifying parameters. EGEN 310R: I thought this course was going to be amazing. Where I worked with other engineering major to design and develop a race car. I took this online due to covid and it was awful. The projects felt repeated and I don't feel like I actually learned anything in terms of working on a team of various engineers.

I cannot think of a subject that was not valuable to me because every course taught us very useful information and improved our way of thinking for how to approach other courses.

linear algebra, i think i only really used these things in that course

Software engineering does not seem valuable but I hear from people in the field that the class was one of the most valuable. I was expecting a class that went more into front or back-end development. I can see how all the classes that I took, whether I liked them or not, had valuable information.

305 was useless for me. Poorly taught and important parts of programming languages were approached indirectly or briefly. 366 wasn't that helpful either.

I think some of the least valuable were classes like Discrete Structures or Computer Architecture because I never saw the content brought up there help in other classes. They might be useful to people that want to do really low level coding, but I didn't find them as useful.

I think social and ethical issues in computer science were the least valuable while I do think that it is important to talk about but I do not think it has an entire semester worth of content.

CS Theory/Ethics felt less relevant.

I felt the CS Theory course wasn't very valuable, because a decent amount of programmers will be writing code, not studying theory. That is generally the Masters program.

EGEN 310. I felt like it didn't add anything to my degree. It had nothing to do with computer science, and even the design component of it wasn't applicable to software engineering practices.

EGEN 310. Complete waste of time, nothing I did had anything to do with my degree.

Any of the classes that didn't have coding as part of the work done. It's not that they weren't interesting, but I'd rather have the experience of coding.

I believe EGEN 310 was the least valuable. While it was lovely to work with students of other majors, forcing computer science students into an environment with other majors without a real purpose created an environment of frustration for everyone involved. Basically, the engineering students had a very clear vision of what we needed to do, but the computer science aspect of the class just felt tacked on. Success in the class also depends heavily on who you get in your group. I was lucky to get a really fantastic group and we worked well all together. Others were not as lucky and I've heard some horrible stories of students who suffered heavily because of unreliable group members. It's also very difficult to get into because of limited course offerings and the amount of students who need to take it. In conclusion, this class was not quite successful in making all CS and other engineering students work well together. It lead to far more frustration than growing experiences.

My opinion is colored by the semester I took it in (Spring 2020), but I did not understand the purpose of EGEN 310 and found the design principles interesting, but less applicable to computer scientists than the other engineering disciplines in the class. I also found Social & Ethical Issues in CS a bit "fluffy," though enjoyable. And finally, I don't understand the overarching thread of 305 Concepts/Programming Languages -- it just seemed like 2-week sprints through different languages, which was great fun! But I'm not sure what I took away from it.

Linear Algebra. Kind of useful for machine learning and I hear for graphics, but it really hasn't come up in my classes, my internship, nor my life. Software Engineering really disappointed me. My friend had to make monopoly with a group, so I was waiting excited to try the design patterns and team structures we were learning about, booksmart-like. It never came. I don't remember much at all from the class because I never got to use it or see it in action. All I remember is that everything seemed very, uh, corporate, and not fun and hard to care about. And I had a partner lined up that I work very well with who was my neighbor, and we did our homework together (as was allowed), but I never got to actually create something with him, and now he's all moved over to Billings and never talks to me anymore, really. :(

I think the least valuable courses were concepts/programming languages and EGEN 310. I feel like programming languages had to cover too many different areas, so the material didn't really stick, while I feel the main content of EGEN 310 repeated the concepts I learned throughout other courses.

CSCI 215 - I don't believe it is the job of a state funded educational system to teach young adults "morals and ethics" that is their parents job.

EGEN 310 was pointless. We already learned about Agile/Scrum/DevOps/CS Specific team work in ESOF 322 and Writ 221. It was a fun class, but i wrote maybe 12 lines of code in that class and all of the soft skills/experience it provided I got elsewhere. CSCI 338 was horrible. As I am not going into academia or research, 246 was sufficent theory for me, in my opinion. It was my least favorite class I have taken in college, and do not see how theory is that applicable to real world jobs. None of the SEs on my team at work have used anything more than basic theory and logic their entire career.

Certain courses weren't great, but as subjects I believe them all to be relevant.

Data Structures/Algorithms and Discrete Structures – While these concepts seemed valuable from an academic perspective, I, personally, cannot see myself applying what I learned in these courses in the real world. I have held

many different positions through internships/co-ops throughout my undergraduate career and never have I ever felt that I used any information from these courses. I felt that these courses were designed to see how well you can memorize lecture material and regurgitate it onto a written exam. I also had an unpleasant experience with the way Dr. Fasy taught the course which made me discouraged to learn the concepts.

Which subject(s) or course(s) were the least valuable? Why?

my technical writing class was a joke and I didn't even learn anything relevant.

None, all required courses are important for this industry

Programming languages, technical writing, and EGEN 310, and software engineering I thought were the least valuable. The biggest problem with all of the mentioned coursework is that it felt like it was just checking off a box in our educational journey, rather than something that was done strictly to benefit us as computer scientists. I think technical writing should be waived by another writing class if desired (eg. texts & critics), programming languages should focus a bit more on the actual structure of different languages (which is super interesting!), and EGEN310 and software engineering should be more dictated by projects that are student chosen and led, rather than making some meaningless arbitrary thing, if possible.

test

SysAdmin was pretty useless if you already spent the time to learn how to navigate Linux, but I guess that's with any other subject too. This one's pretty much just for me.

I think the CS theory class was the one that felt like the most waste of time because it felt way too academic and/or too historical. I also think architecture, while interesting, wasn't very useful to me as a high level developer.

I think classes focused on using tests as a measure of learning metrics were a big issue. They were very punishing for topics that in real work you would be looking up the niche concepts and question ideas as you go to make sure you are correct. I would not apply this to the strictly theory classes necessarily as you have to be able to have a measurable metric but i believe that any class based on being able to design or develop software should focus on your ability to develop it with the resources and tools you would be able to have in a work environment. Focusing on actual projects rather than odd and confusing quiz and test questions.

Proofs. Not applicable to what I do.

CSCI 338 It is a class that has merit in academia but not much applies to a desk job.

338 - CS Theory, because it does not seem applicable to many real world scenarios and I think the most important aspects were covered already in 246 - Discrete Structures

I don't feel that I got much out of my technical writing class, mostly because it was interrupted by Covid.

Just my extra curriculars. All of the cs classes were good.

EGEN310 was completely pointless in my opinion, though it may have been due to the covid structure of the course. Additionally, CSCI215 I felt I didn't learn anything in, as it was all what I would think to be pretty self explanatory content, though I understand why it would be required in a curriculum.

anything taught by hunter lloyd, daniel defrance, or millman

Artificial Intelligence: the heavy focus on group projects made an already difficult subject even more difficult. Students are not typically experts at working effectively with a collaborative tool like Github. Make it a requirement, or touch on it in the coursework.

CSCI 246: The professor I had (blanking on the name), was rude, sexist, and a complete failure as a professor. I still don't remember anything that I learned in that class or why discrete structures even matter. I have never been so embarrassed to be a MSU student than when I was in his class. I heard from someone recently that he was let go due to complaints which is awesome. If he hasn't, I would highly recommend he be removed from all classrooms ASAP, I don't remember his name but I took it in Spring of 2018. PLEASE GET RID OF HIM. CSCI 145RA: Had a lot of potential and I could see it being a very helpful class, but I don't feel as though I learned all that much about web design, it's supposed to be an intro class but felt much too easy compared to the other intro-level classes.

Calc II - I haven't found it applicable to other things I do

The course that I thought was least valuable was EGEN 310R. This class was extremely frustrating as there are not enough electrical engineers in it thus the computer science student is tasked with figuring out a lot of things that really aren't things that are taught to a CS major unless they have a strong personal passion for robotics or embedded systems. Furthermore, this class will be even more challenging for those that chose to take it as it will reduce the rare chance of having both a CS and EE in your group. The teachers are not helpful in getting the actual content needed out to students to complete the final project and it felt like a third grade class being taught how to analytically watch a movie again. To make matters worse while being tasked to watch a movie or read the script of a movie, the over arching rover project continues to creep up on students with no mention from professors until about 2/3 of the way done with the class. Overall, it felt as those I as a CS student was tasked with redesigning a product similar to what we worked with in CSCI-455. I know that this class is currently being transitioned out of the CS required program but I truly think it has the potential to be a great class but each team should be provided at least one of following engineers(EE, ME, CS, ChE or at least provide some sort of help sessions!

Discrete Structures was the least valuable class not because of the provided information but because of the way it was taught. Brittany Fasy had very little idea of what she was doing and the class was a mess. I ended up taking it again because none of the info from that class was retained. Computer Systems also had the same issues, though less extreme, again the information the class provided was very weak.

I think that the Concepts of Programming languages course was likely the least valuable because the integral topics about programming languages that I learned from the class were taught to me again in Compilers.

Much of the early classes (132, 232, 112) left me learning much about actual programming to myself. Many of these classes gave me the frustrating experience of learning the hardest way of doing things before finding out the best way.

I would say I was not a huge fan of theory courses, it seemed that those classes did not need a full class dedicated to them, as the things we were learning usually had been covered in other classes at some level. The same is true for real world application that if theory is what you were going into, the class was needed, but if not it seemed like information that was easy to forget.

The least valuable course to me is CSCI112, since I didn't use C language in programming after taking that class, and the concepts from that course are already covered by CSCI127 and CSCI132.

Computer Ethics 215, Programming languages 305. I did not take anything valuable away from these classes. There weren’t any new concepts introduced to me other than new syntax of languages.

Embedded Systems - focused entirely on writing python scripts for raspberry pi, didn't actually learn practical embedded software skills Compilers - I understand its a national requirement but really doesn't seem like practical knowledge ESOF 422 - Spent majority of time learning/navigating very niche proprietary software EGEN 310 - Spent most my time teaching myself poor electrical engineering. CSCI-338 - Only relevant for people getting into theory, not much that transferred to realworld.

Ethics in Computer Science *could* be a good class, but the current implementation has a lot of fluff and largely unnecessary sections that make the class really unpleasant and useless.

CSCI 460: Operating Systems - this class was super high level with a lot of theory and I had a hard time understanding how the programming assignments were reflective of what we learned. CSCI 455: Embedded Systems, Robotics - this class taught me nothing and was needlessly stressful with unclear objectives that did not come together in any sort of meaningful way that made me feel the class was worthwhile. CSCI 338: CS Theory -

yes, I understand it's a theory class. But I can't see why it's a required course seeing as how the content taught in it is almost never utilized throughout the rest of the CS program.

I don't think any weren't valuable.

The first course that comes to mind was Social & Ethics. This course was very straight forward and felt almost like a filler class. It could of easily been implemented into another course like Software Engineering or something. The other course, though not required, that I took that I felt wasn't very valuable was Systems Administration. This class was pretty much the Linux for Dummys manual and it felt more like a 100 level course than a 300 level course.

Concepts of Computer Programming - I was only able get a small taste of languages and did not give me enough to really be able to find any useful

Discrete structures - I did not learn anything from this class, could have been bad teaching but was completely useless to me. The way that Java was taught did not seem to be a good introduction. Many people struggled with this especially compared to my peers at other schools. I think there is a better way to go about introducing students to OO languages then trial by fire.

The courses I felt were least valuable were not necessarily courses I did not enjoy, but overall courses revolving around theory or algorithms felt as though they would not be applicable much in the future. They were some of my favorite classes and I definitely found them interesting, though it seems unlikely I will need to carry much from them forward into my future.

The lower level classes because the material covered in them can easily be learned and found elsewhere with little deviation in knowledge

CSCI 432 and CSCI 232 were the least valuable classes in my opinion. The curriculum in these classes was taught so quickly and not nearly in-depth enough. Fasy taught 432 and felt like I barely made it through the class. Every week a new algorithm topic was taught and the weekly homework assignments were often so in-depth that I felt like I didn't understand anything. CSCI232 was similar in the sense that a lot of complex data structures were taught very quickly. Granted I took CSCI232 Spring of 2022, so the pandemic really affecting the teaching. However, after only one semester worth of Java experience, I felt like the class was extremely fast paced and I didn't have the experience or skill set to complete assignments and understand the material.

The security course wasn't super valuable. I'm glad I took it but I left the class feeling like I didn't have much of an arsenal to defend against hackers. Also, programming languages wasn't very valuable.

Computer vision was a sub par course. We didn't go in-depth into content. I also had to self teach myself everything because the instructor was sub par. Technical writing I thought was just busy work and not thought provoking.

Any of the math classes that we had to take were a complete joke. That is Calc 1, Cal 2, and Linear Algebra. While its fun to sit there and just memorize stuff that we have built in functions for it really isn't all that fun. There is no reason we should be memorizing formulas in this day and age. Instead of learning to be thoughtless students that just memorize stuff they should teach us practical application of the mathematics. For example, a student can go through data mining without taking linear algebra. It's more important they understand what they are doing and how to use the mathematics to solve problems rather than just memorizing the equations. Generally, most computer scientists do not use mathematics in their field. I would rather have 2 more coding classes than learning how to compute derivatives. I would like to have another data mining class instead of linear algebra.

I thought Egen 310R multi disciplinary engineering was the least valuable course. I unfortunately was not paired up with a electrical engineer like every other group and the project partners did not have any way for me to contribute to the final.

Discrete structures, I didn't really see the application of that class as much in my other classes

Calculus, haven't used it yet. Also all my core classes, nothing is applicable. Although some classes were good for generic knowledge, the workload conflicted with CS courses that were more important. Data mining has been only confusing, nothing seems to translate to something actually applicable.

While I didn't have the theoretical CS classes, it was at times difficult to see the value of super theoretical subjects. I did find these interesting and helpful at times, so I wouldn't call for these to be removed from the program, but they could potentially be paired down.

Ethics - the course could be super interesting but the class fails to address current ethical issues in a meaningful way. It would also be better taught as a seminar type course similar to honors college courses. 112 -

EGEN 310 largely felt like a waste of time, as there were several other classes that had groups of students working together on a project. Additionally, it kind of felt like the only reason the CS students were there was to make everyone's project work, as opposed to being a contributive member. CSCI 305 also seems largely redundant as much of at least the conceptual stuff is covered in other classes such as theory and compilers.

I'd argue that I didn't take a class that was less valuable than another. They were all valuable in their own way. However, if I must choose CSCI204 was not particularly useful to my professional development.

I found EGEN 310 to be worthless. I was told to learn a new field and implement that. I didn't use CS at all, and only used what I learned in WRIT 221. The new curriculum removed 310 though. CS 145RA was too easy to be useful. We covered the content again in 331 in 2 weeks.

Unfortunately anything that had us working with C is of lesser value. C is not as commonly used now and I only see it being pushed further in the background in the years to come.

N/A

The least valuable course I took would probably be CSCI 130, Programming in C. I found that with the way the class was taught using extremely old technologies was not very relevant to any other course or subject I came across. It could simply mean that I haven't encountered a use case for ideas from the class and it does still have some value.

N/A

Least valuable courses were social and ethical issues, web design, and discrete structures because for the most part they didn't cover real world concrete skills that transfer over to like as a software developer.

I am torn on 338 it is very useful in learning what even is computable but I don't think that most CS students have the math background to fully appreciate it. 112 was also not that useful, it was just trivial and slow.

I didn't find much use in social and ethical issues. Nor did i find it very important to take calculus and statistics. Statistics should be replaced with data mining as a requirement. There are similarity's but data mining is more computer oriented.

CSCI 338

Administrative systems and some of the options for the 13 credits of math and science. Administrative systems offers great understanding of the command line general Linux based operating systems, however I don't see how this is class will be helpful to me in the future. It was very fun to learn how to use the command line, thought I feel a more practical application of the commands learned would make the class more useful. Multidisciplinary engineering design really only taught me the cycle of which things are designed, which is covered in the UI/UX class. Computer science students are also very often pushed to do electrical engineering student tasks when those students aren't present in the group. I did not feel I learned anything valuable from this class and it was a waste of my time. I think it should be replaced with a mobile development class.

The software engineering courses were not very valuable to me. I think much of that information can be learned in an easier format online, and I think the course was taught poorly (however this was during 2020 COVID times so I can't put too much blame on the course format). I struggle to remember the concepts in that course related to UML diagrams, etc.

I found more theory focused courses such as Computer Science Theory or AI to not have much application but they may play more of a role in graduate school.

There were a few classes in particular that I found to not be particularly helpful, namely CSCI215, EGEN310, and ESOF322. The social and ethical issues in CS class seemed unnecessary and not super useful. EGEN310 was pretty stressful and I did not learn much from the experience. The content in the software engineering class seemed

pretty irrelevant to the average graduate. I don't know how much ER diagrams and class diagrams are used in most companies these days.

310R - Which has just been removed from the curriculum... thank goodness. What a joke.

EGEN 310 wasn't very valuable for CS specifically due to the limited interaction and large amount of knowledge we aren't taught for the subject matter. I also think 215 wasn't terribly necessary as a full course. Maybe if it was integrated with another course, but on its own it felt like a waste of time.

Computer Science Theory I think was the least valuable, but that's probably because it had been a while since I had taken Discrete Structures so I was trying to relearn old concepts and learn new concepts at the same time

The only course that I don't think was very valuable was calculus 2 because I haven't needed to use nearly all of the topic covered in that class since the class. The only useful part of calculus 2 for me was the summations but all the other topics were not useful to me.

Egen 310, I took during covid semesters everything felt like busy work for the sake of busy work.

A full semester of computer ethics was a complete waste of time not because the course was boring or long drawn out but the fact that when out in the field the code of ethics comes down to business practice and personal morals. In other words, it was a pointless waste of time because of how blatantly obvious the content was the majority of the time.

The ethics course, because it turned into busy work, and a full semester on it was a waste of time and I learned nothing valuable after a week of lectures because it became redundant.

CSCI 215 because it became a political discussion rather than a discussion about todays current CS environment

CSCI 305 and EGEN 310R are the 2 courses I have always thought were not needed. 305 would be a substantially better course if it focused on relevant programming languages and how to pick them up in a short amount of time. I just felt that using the older and more obscure languages did nothing for me academically. 310R is a class that I felt was a joke. When I took it the 'group project' was decided by each group and it seemed that you could choose a project with very little work if you wanted to. My group chose a project with a relatively heavy amount of work and almost all of it fell on myself and the electrical engineer in the group. I think to fix this class there should be a very specific goal in mind that fully incorporates each discipline equally. I also think that professors from each discipline should be involved. How does a general engineering professor know what good code looks like or how to help a student in a discipline that is not their own?

CSCI 127 was nice for getting use to coding but there should have been more focus on data structure and how to use GitHub. Upper division course are difficult without having Git instruction in the earlier courses. Discrete structures also didn't seem extremely helpful in my later course, but this could be due to the fact that I took this course during Spring 2020 when the pandemic started. The C course was unhelpful, was able to complete CSCI 366 without it.

ESOF 322 - Topics touched on are rarely used in the professional world CSCI 215CS - Failed to touch on the current ethical issues in CS CSCI 305 - did not learn about how languages are structured, just learned about random languages that anyone could pick up on their own time

CSCI 338 didn't really seem all that applicable to anything that I would do. Learning about how to prove that something is NP-Complete just didn't seem all that helpful. Personally, I think CSCI 338 should be a recommended elective but not a required course. That said, the professor I had, Sean Yaw, did a really great job breaking down the more difficult aspects of the class and made them much easier to understand. He did a good job trying to make the course interesting and engaging.

CSCI 246 Discrete Structures - I don't remember anything I learned in that class and haven't needed to think about it again. If it weren't for complaints for underclassmen I think I would have forgotten the class existed.

Theory because it has little connection to applications in the modern world. It's more of a history class with obscure math

I strongly disliked calculous. I find it interesting that calc 1 and 2 are required for this major. I understand that knowing the math behind some of the functions we use is essential, but personally I do not think I will use this in my career.

EGEN310 is a useless course and I do not understand why I was forced to pay money to take it. It is one of those things that look great on paper - teaching students how to work in a job setting and learn leadership - but it does NOT pan out that way. It is somewhat akin to high school courses: busy work, forced leadership roles that do nothing, and a skewed view of how industry operates. Now, the rover project was quite cool and I think a CS directed instance of this class would work well in the curriculum. Also, I think Ethics is a very important class but it was bungled the semester I took it - could be better.

College Writing and Technical Writing could very well be combined into a single course. It seems too much is focused on writing a paper, only a single semester seems necessary. The art of making essays is very different from writing used for CS courses and still very different from research papers within the CS world (I have read a good bit of them).

To me discrete structures and computer science theory were valuable to an extent, but it felt like they were based so much in the theory that it was hard to see how the concepts would actually be applied to programming tasks. If there had been some more hands on examples with the theory explained I think it would have been more valuable.

EGEN 310: I did not feel like this class was relevant to a career in CS. To make it relevant, the project should be able to focus more on CS. WRIT 221: This was also extremely irrelevant to practically all aspects of a career in computer science. The projects, such as the creation of a trifold brochure or an advertisement, were not helpful to me. The only practical aspect of this course was the creation of a resume.

EGEN 310, CSCI 361, CSCI 246

Ethics for Computer Science

None Every class has its necessity and value

M 172. I really don't see how I would need to use any of the materials I learned in that class in my field given that I won't be going to grad school.

I do not think there is any course that were least valuable.

Personally I found the Computer Science Theory and Networks classes to be the least valuable in my experience. Theory was poorly taught even though I can tell that Binhai knows the material very well. It is an important topic to be sure for those who are pursuing an academic related path in computer science but I found myself struggling with relating the material to what I would actually be using computer science for in the field most days. Networks was a similar class where I went in expecting to work on projects and assignments that were related to what was being taught in lecture to help solidify my understanding of the material. Unfortunately this was not the case and most times I found myself having to spend most of my time researching what was asked of us in the programs as they had hardly anything to do with the lectures themselves.

In terms of CS learning value, EGEN 310 was the least valuable. I learned a lot about teamwork, and a fair amount about building a remote control system, but I could've learned significantly more if the class didn't have all of the annoying hoops to jump through. CSCI 246 Discrete Structures was quite frustrating for me, but that was because I didn't seem to mesh with Professor Fasy's teaching style.

None Every course has its necessity.

CSCI 351 or any of the EELE. The CSCI course felt out of place and the EELE classes kind of go into what we do throughout our CS courses are about, but its like comparing apples to oranges. The EELE courses serve no real use for what we do in coding.

CSCI 204: This multimedia dev methods course which is just a Unity Game Development course is poorly run. Often times coding in this class results in just following online tutorials and then modifying parameters. EGEN 310R: I thought this course was going to be amazing. Where I worked with other engineering major to design and develop a race car. I took this online due to covid and it was awful. The projects felt repeated and I don't feel like I actually learned anything in terms of working on a team of various engineers.

I cannot think of a subject that was not valuable to me because every course taught us very useful information and improved our way of thinking for how to approach other courses.

linear algebra, i think i only really used these things in that course

Software engineering does not seem valuable but I hear from people in the field that the class was one of the most valuable. I was expecting a class that went more into front or back-end development. I can see how all the classes that I took, whether I liked them or not, had valuable information.

305 was useless for me. Poorly taught and important parts of programming languages were approached indirectly or briefly. 366 wasn't that helpful either.

I think some of the least valuable were classes like Discrete Structures or Computer Architecture because I never saw the content brought up there help in other classes. They might be useful to people that want to do really low level coding, but I didn't find them as useful.

I think social and ethical issues in computer science were the least valuable while I do think that it is important to talk about but I do not think it has an entire semester worth of content.

CS Theory/Ethics felt less relevant.

I felt the CS Theory course wasn't very valuable, because a decent amount of programmers will be writing code, not studying theory. That is generally the Masters program.

EGEN 310. I felt like it didn't add anything to my degree. It had nothing to do with computer science, and even the design component of it wasn't applicable to software engineering practices.

EGEN 310. Complete waste of time, nothing I did had anything to do with my degree.

Any of the classes that didn't have coding as part of the work done. It's not that they weren't interesting, but I'd rather have the experience of coding.

I believe EGEN 310 was the least valuable. While it was lovely to work with students of other majors, forcing computer science students into an environment with other majors without a real purpose created an environment of frustration for everyone involved. Basically, the engineering students had a very clear vision of what we needed to do, but the computer science aspect of the class just felt tacked on. Success in the class also depends heavily on who you get in your group. I was lucky to get a really fantastic group and we worked well all together. Others were not as lucky and I've heard some horrible stories of students who suffered heavily because of unreliable group members. It's also very difficult to get into because of limited course offerings and the amount of students who need to take it. In conclusion, this class was not quite successful in making all CS and other engineering students work well together. It lead to far more frustration than growing experiences.

My opinion is colored by the semester I took it in (Spring 2020), but I did not understand the purpose of EGEN 310 and found the design principles interesting, but less applicable to computer scientists than the other engineering disciplines in the class. I also found Social & Ethical Issues in CS a bit "fluffy," though enjoyable. And finally, I don't understand the overarching thread of 305 Concepts/Programming Languages -- it just seemed like 2-week sprints through different languages, which was great fun! But I'm not sure what I took away from it.

Linear Algebra. Kind of useful for machine learning and I hear for graphics, but it really hasn't come up in my classes, my internship, nor my life. Software Engineering really disappointed me. My friend had to make monopoly with a group, so I was waiting excited to try the design patterns and team structures we were learning about, booksmart-like. It never came. I don't remember much at all from the class because I never got to use it or see it in action. All I remember is that everything seemed very, uh, corporate, and not fun and hard to care about. And I had a partner lined up that I work very well with who was my neighbor, and we did our homework together (as was allowed), but I never got to actually create something with him, and now he's all moved over to Billings and never talks to me anymore, really. :(

I think the least valuable courses were concepts/programming languages and EGEN 310. I feel like programming languages had to cover too many different areas, so the material didn't really stick, while I feel the main content of EGEN 310 repeated the concepts I learned throughout other courses.

CSCI 215 - I don't believe it is the job of a state funded educational system to teach young adults "morals and ethics" that is their parents job.

EGEN 310 was pointless. We already learned about Agile/Scrum/DevOps/CS Specific team work in ESOF 322 and Writ 221. It was a fun class, but i wrote maybe 12 lines of code in that class and all of the soft skills/experience it provided I got elsewhere. CSCI 338 was horrible. As I am not going into academia or research, 246 was sufficent

theory for me, in my opinion. It was my least favorite class I have taken in college, and do not see how theory is that applicable to real world jobs. None of the SEs on my team at work have used anything more than basic theory and logic their entire career.

Certain courses weren't great, but as subjects I believe them all to be relevant.

Data Structures/Algorithms and Discrete Structures – While these concepts seemed valuable from an academic perspective, I, personally, cannot see myself applying what I learned in these courses in the real world. I have held many different positions through internships/co-ops throughout my undergraduate career and never have I ever felt that I used any information from these courses. I felt that these courses were designed to see how well you can memorize lecture material and regurgitate it onto a written exam. I also had an unpleasant experience with the way Dr. Fasy taught the course which made me discouraged to learn the concepts.

Q6 - Are there any subjects or courses missing from the curriculum?

Are there any subjects or courses missing from the curriculum?
maybe some more modern programming.
Need C++ into the curriculum, we need to stop relying on scripting languages (python and js) as Moore's law is plateauing on us.
Quantum computing is the big one that sticks out to me. Otherwise all of the obvious stuff is either covered in the undergrad or graduate curriculum.
test
Not a course, just industry terms. What the hell is a full-stack developer
I wish there was more focus on how to use modern tools like Github and how to actually do code reviews and set up CI/CD pipelines. Also a class on good API design, the difference between GraphQL, REST, gRPC would have been cool - I learned all that entirely outside of school.
I can't think of anything specific missing
Writing tests
No.
Not really, but the courses listed in the catalog need to actually be taught more often
I think an applied data science and algorithms course would be useful, where you work on projects that use data structures and algorithms.
I wanted to take computer graphics for soooo long and it never was available to me.
I think M221 should be required for the major (and maybe M441 too would be quite valuable)
more game development/unity
Machine Learning: Limited availability (Spring Only?)
Based on my experience with my peers in computer science I think that some courses in public speaking or communications should be required for a computer science degree. Many if not most of my peers were terrible at public speaking, awful at communication, or just too terrified to speak in front of a group of people, THIS IS NECESSARY for life. The writing level of my peers was also atrocious. I am a published author so I have a great deal of experience in writing, but some of my classmates seemed to not understand punctuation. The tech writing class was awesome, maybe consider requiring some additional courses.
A course that teaches student's how to use an IDE.
I think that the current curriculum is solid and the only thing I wish I could have learned earlier was utilizing the tools at hand to software developers. Tools like git and debuggers should be taught within the freshman year curriculum for any CS major. They are inherently complex but when taught how to use them in a less stressful environment with someone that knows the inner workings they can be simple to understand and use. Along with this using a debugger greatly expanded my knowledge of how computers and thus computer programs work at the next level. In my opinion, these tools should be taught to you within the introduction to data structures class.
An intro class that discusses the software we use as computer scientists to standardize a common basis of programs (e.g. GitHub, JetBrains, etc)
Front end engineering

None

no

One course that is missing from the curriculum is the CSCI447, the machine learning course from our curriculum starts at a very high level, and I believe that a lower level machine learning course can help us better to get familiar with the topics.

Cyber Security.

Cloud technology was only touched on in 366, there should be a full, hands on course on this. Version control was never fully covered, was used for a couple projects but we were never explicitly taught full features. GUI and UI design, there is one class but it was scheduled for the same block as a required course so many of us didn't have the opportunity to take it. Modern web design with react, javascript, HTML5 etc... Same as gui/ui, tough to fit in schedule. Didn't learn anything about security. Devops course would be great: setting up CI, environments, cloud infrastructure/architecture, etc...

No.

More available courses on Cybersecurity at an undergraduate level would be nice. In addition, courses that could help prepare for coding interviews would be a great help to students.

Not really, but there were a few that I wish were offered more often. Like computer graphics and machine learning.

I feel like one thing I missed out on (which of course is now added to the curriculum) is some sort of Networking course. The closest thing I got to that was in Computer Systems where we tried to do networking in C which was awful and confusing.

I would have liked to see more unit testing and programming with design patterns in mind being taught. A class specifically designed 'around deployments and CI/CD would have been very helpful for anyone going into industry.

A class on computer security could certainly be added to the undergraduate curriculum (though I believe this is happening already).

Personally, I wish that there was more access to material from newer fields such as video game design. Classes covering material like this would be very relevant and enjoyable for many students such as myself.

I think the only thing missing from my college experience was learning newer technologies. The first two years of the degree were good building blocks for the latter half of the curriculum, but after that it seemed that the rest of the courses just went more in depth on basic languages. I would have enjoyed a course or two were we explored languages or technologies that are being used in the real world. CSCI305 would have been a very interesting class is we focused on newer languages. I did not benefit whatsoever from learning about Fortran and Lisp for a month.

More applicable cybersecurity courses. Especially web app security. I also wanted a DevOps course.

More in depth web development classes. A ReactJS class, A server class using Golang or nodeJS. And Web3/Blockchain class would be awesome!

Intro to Github, Mobile development with java, Coding with swift

I maybe would like to see some type of cyber security class.

A course on the CI/CD process and how to host deployments and everything that goes into that is missing. I only found experience in this from internships. Release pipelines, testing, kubernetes, etc.

Anytime hardware components were brought up in my classes I was very lost because I have very little experience with hardware concepts. I would love to see more emphasis on gaining a basic hardware understanding. I also feel, given trends in industry, cloud computing concepts and subjects about servers/networks could use more emphasis.

I think it would be great to have more of the upper division courses as two class series, I think the courses provide a good introduction but following that it would be great to dig deeper. Especially machine learning, AI and

security. I also think a natural language processing class would be interesting. I also think more theory classes would be great. I also think that courses should use a variety of languages, I think 80% of my classes used python.

Nothing in particular, but it was frustrating during registration that certain classes never seemed to be available. I would suggest offering a wider variety of electives in general, and ensuring that the electives that are on the catalog are offered more frequently/consistently.

Interesting, I think that a course similar to 494 (Industry Methods) that is extremely project-based would be an extremely valuable course.

Clem is teaching digital forensics in the 2nd half of ESOF 422. This should be maintained. It isn't a big enough topic for a semester, but it is super interesting and teaches a lot about computers. Something about producing finished products would be cool. Like, making applications that are useable. We write all this code to be run on specific systems, environments, or IDEs. A class on production would be pretty useful.

When I attended there were few if any courses relating to security and so I was unable to take any. I would recommend adding more to that area as that field will always be important.

A course entirely on Github that is a low level course so that students have to learn Git very early. More courses focused on IT type work. Blockchain technology.

I think there should be a front end design class as a part of the required curriculum. I was never able to take the optional web design or UI design electives and I think these are valuable skills that every programmer should have.

A career course that teaches you about how to write a resume, apply, and interview for internships and jobs in CS. It could also teach you what career paths are available and what electives you should take to explore these paths.

More options for cyber security courses.

More math fundamentals (M221) would be helpful, or something that discusses numerical error (M441). A better capstone would be good, something like the inter-disc capstone or other engineering capstones where students work year-long on a project. The advantage of this is a nice portfolio project and something to discuss in interviews.

I would have like to take a class using swift to develop ios apps. I understand the difficulty of that though as you must have apple to do so. I did take the computer vision class which we did a little bit of android development which was cool. I think there should be a class dedicated to mobile development.

Some more IT related courses

Mobile development, API calls, database queries, practical/industry programming.

No, I think everything is covered. However I would have liked more direct programming assignments where I am writing code. I don't feel like I wrote very much code to obtain my degree and I think that could be improved.

I would've liked to have learned a little about electrical engineering and the hardware side of things.

GIT!!! I've had teachers say that you won't get a job if you don't have a Github profile and git isn't even tough... does MSU not want their graduates to get jobs?? there should be a class dedicated to using Git bash and every freshman should be forced to take it. My first exposure to git was in 366, and I now use it for everything all day every day.

Cybersecurity and operating systems should be required courses. I don't really know how either of those two subjects work because I never took the courses.

I don't think so

having computer security required in the course curriculum

a Git class/more integration.

Courses on workflow and actually project development are sorely missing from this academic course. Teaching Github, how to Google, and work pipelines would not have been taught if it wasn't for Carson, and I want you as a learning institution to realize and reinforce the fact that many of these tools are so important to every single developer/computer science student.

A course to teach how to go about finding answers to difficult coding problems we may actually face. Because let's face it we will probably use languages that we didn't learn in our studies and the most important thing we can learn is how to go about finding information and tools to use. Also, teach GITHUB!!!! If Carson Gross didn't teach it, I would have had no idea how to use it and that is something that NEEDS to be taught.

No

I don't know if it would require a whole course, but something related to tools in the workplace. I know some professors use GitHub and do brief intros to it, but I think a course dedicated to tools like GitHub would be very beneficial.

I wish there was more courses offered at the same time. It often feels like we have to randomly select any class that is open rather than what we find interesting. It would be helpful if we included advanced versions of the UX design and security courses. There should also be courses that help students understand the licensing of softwares and open source code vs. free software.

Intro course detailing how to use the command line, github, IDEs, etc. It was very challenging as someone who had no prior cs experience to pick up these difficult and very necessary skills while trying to balance current course load.

A git class would be nice. Carson Gross does a decent job explaining things in CSCI 366 and CSCI 440, but I feel that that's way too late to be taking such a class. I think more courses in cyber security would be interesting.

I would love more Cyber Security type courses but I know that is in the works.

I did not feel this way

I think that Human Computer Interaction and design courses should be required for the curriculum. We are often only taught applications in classroom settings and are told exactly what is expected, though much of computer science is understanding our users and getting to know their unique needs.

React / Vue / JS frameworks - a high end web design / web app class for CS students. Making web apps is a huge job industry and is useful to know how to do regardless. Neural net frameworks (pytorch, tensorflow). AWS / working in the cloud. Although the upper division courses are great, not a lot of them teach actual skills - it is just more background.

More time needs to be spent with the fundamentals of coding. As a tutor, I can tell many students struggle with understanding what code does even in CSCI 232. Problem-solving is a big aspect lacking in the introductory courses. How to simplify code, what are different ways of writing the same code, create basic problems that incrementally require more while/for-loops throughout the course. CSCI 132 when introducing Java, more time is need to be spent on the language, then onto polymorphism, then complex and traditional data-structures may be explored.

I really wish there were some classes that focused on the intersection between hardware and software. I think it would be very valuable if some of the electrical engineering courses could count as professional electives or count in some way to a computer science degree and not just towards a minor if you already have a minor.

CSCI 476, in my opinion, should be a required class. There should be some early class, such as CSCI 112 or CSCI 127, which teaches how to use git. Later classes assume that a person knows it, and there is a 400-level seminar which teaches it, but it's a critical-enough skill that it should be taught early.

Mobile Development, Modern Technology Stacks (MERN, MEAN)

No

None

Yes, there should be a class that teaches students how to use linux systems that includes a designated lab and lab time.

I do not think there is any course that is missing.

I think they might be adding them soon but some more courses involving security practices might me nice. I also think that adding more courses in designing applications such as the UI design course would be beneficial as we

have a large selection of courses designed to teach coding practices but not so many in teaching how to make those applications appealing to a target audience and learning what makes those attractive qualities in applications.

Having more UX/UI courses would be really cool. More Robotics courses as well. Having some courses in video game development would be really fun and educational, and those courses would likely engage students who are struggling to maintain interest in their studies.

None

Intro to Java or any of the other big programming languages that are coming about.

We need more security, gaming, and design courses.

I think that a course focused on mobile app development would intrigue a lot of students because app development is so huge in our world.

i think CSCI 476 should be a requirement as opposed to an elective course

Possibly more classes with real world examples of something you might do for a job. For example CSCI 204 and designing a game or EGEN 310 and working with an arduino, app development and bluetooth capabilities. It's tough to say without having worked in the field. I think it would be a positive addition to have a class that focused more on computer hardware.

Cloud computing (using AWS), data science A 200 level 'tools for computer scientist class' or something like that could be helpful. Could cover terminal basics, git, Stack Overflow, open source

I think security classes would be really helpful to have as a more standard class we have to take, as well as a class dealing with UI in some way as that is what is so prevalent in software development today. It was tough taking EGEN 310R and having to write an app, when we had no experience with that before, and no teacher to help guide through that.

I think that the advanced web development course while in theory is great it does not do a deep enough dive into the technologies and spends too much time on what was taught in the prerequisite so I think a course going over more in-depth into creating a front end with React or Node and linking to a database is missed.

Not to my knowledge.

Not that I can think of from the top of my head.

It would be nice to have Distributed Systems offered more often.

It would be nice if courses like Distributed Systems were offered more often.

A whole class on properly using GitLab (not Hub). The commands are easy but learning the workflow with a group is vital and takes several weeks over the course of a project for everyone to see how much rope they need to hang themselves with bad group members.

I think it would be great to have an introductory Git course for all CS students. The way it's taught right now that I experienced is stuck into other classes, but students aren't able to understand it quickly enough and it leads to some horrific code mixups. If a student doesn't know someone who can help them with Git, they get discouraged very quickly and this would be completely fixed if there were a course to help students learn it ahead of time. Git is used so much and I think it would be great to give students a dedicated space and time to learn these tools without fear of breaking their important coursework.

I think there should be a class, potentially just 1 credit, toward the beginning of the curriculum going over version control and Git. I still don't fully understand it, I started to use it in my junior year, and at that point professors largely assumed that we understood how to use it. In this sense, any and all topics in https://missing.csail.mit.edu/ would have been useful.

I would have appreciated more electives to learn how to properly use a specific tool or programming language, like how the C class was set up. Yes, I can teach myself C# or Rust, but it's a slow process. Knowledge is freely

available, I just don't know where to start, what I need to learn, why I'd want to use the language over others, or any best practices!

Technical interview prep course

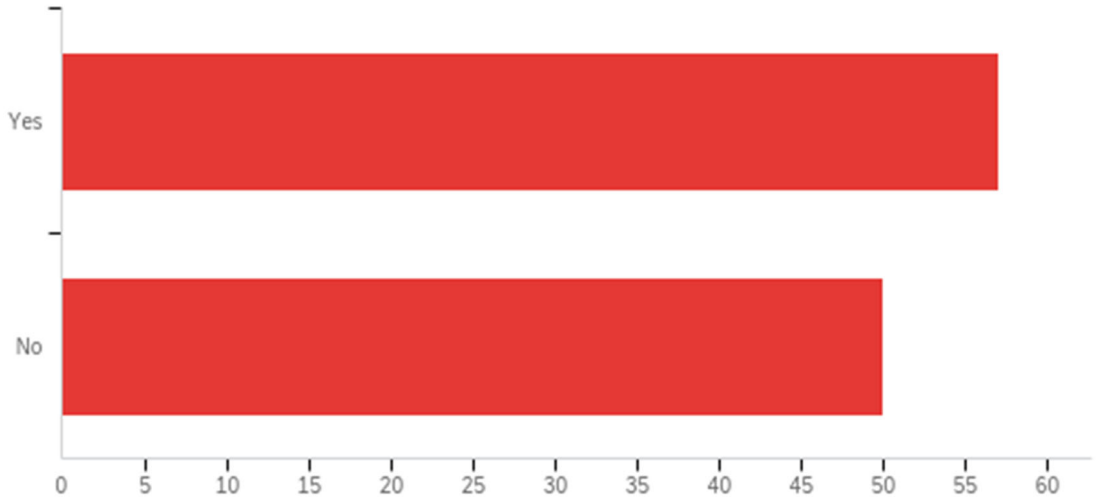
I have recently encountered several seniors who still have no idea how to use git. In my daily experience as a software engineer this is one of the most vital skills that is needed in the work place.

I think that Operating Systems should be a required class, as I am leaving college with nearly 0 experience writing code in C and have no idea how an OS works. I also think that networks should be touched on more at a lower level- 90% of my job is making API calls and HTTP requests and setting up endpoints, and I was so confused during networks I barely learned anything. Also, AWS is so widely used across the industry I dont know why we dont learn it at all

I have yet to apply for a job that didn't have AI or machine learning in the description. I think its absolutely absurd that there are only two AI/ML related courses that are only offered in either the spring or the fall. If the curriculum wants to stay industry relevant MSU needs to provide relevant topics.

I believe more courses should be offered like UX Design & HCI. I loved taking Dr. Stanley's courses and I think the curriculum could benefit from more courses with a similar subject. Co-op recognition: I participated in a 2-term co-op rotational program during my undergraduate degree (8 months total). Because I am an interdisciplinary student, I was told there wasn't any option to count my semesters away doing a co-op as credit towards my degree. This was also frustrating because I was considered an unenrolled student during my time away working in industry. This was hard on me because it meant that I had to start paying off my student loans even though I was only temporarily gone to get experience working in industry. I felt that I was being punished by the system for getting experience in the real world to apply what I had been learning in my courses. It was a big hassle trying to reapply to be a student and I fear that this lack of recognition for co-op programs will discourage students from partaking in the future. This is unfortunate because I truly feel like I learned far more by getting to work in industry during my time as an undergraduate. I hope that the CS department at MSU can accommodate students in the future that want to take a semester or two off to work in industry.

Q7 - Did you participate in an internship?



#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	Did you participate in an internship?	1.00	2.00	1.47	0.50	0.25	107

#	Answer	%	Count
1	Yes	53.27%	57
2	No	46.73%	50
	Total	100%	107

Q8 - If you did participate in an internship, what value did it provide? (If you did not, skip to the next question.)

If you did participate in an internship, what value did it provide? (If you did not, skip to the next question.)
Tons! I learned quite a lot about web programming, and gained confidence in my ability to construct complex, sizable software systems with real consequence. Also, the research I conducted was of immense value (it was the most valuable part of my undergrad career by far) and I think the department should further encourage undergrad research.
test
It helped me learn how to interact with other developers and the process of agile development in a way no class could. I also learned a lot of practical skills like working with a SQL database and tracking down weird edge-case bugs.
I did a Boeing internship doing data mining in documents and it not only helped my confidence but also allowed me an outlet to test my expertise and abilities with difficult projects that many of the lower and mid level courses pursuing my degree lacked.
Working with senior developers and writing tests
It showed me how software is applied in an industrial setting.
Front End developer.
It provided a chance to actually apply some of the things I learned from the curriculum to a real product
n/a
Taught me how to interact with people in a corporate setting. Presented an opportunity to develop my programming skills.
Teamwork, version control, debugging/troubleshooting skills, professional development.
It has taught me how to create automated web tests and how to work as part of a team.
Gave me a lot of experience with Adobe products. Also let me realize a corporate environment is not for me.
It gave some real world experience toward what it would take to get a job in the real world, and also gave some job experience that will make getting jobs much easier.
It was the most valuable "course" that I took. Learned about working on large codebase, writing clean and readable code, version control, working with others, issue planning / tracking, modern cloud technology, setting up environments, pros and cons of different technologies and practices, and much more. Was the main reason I was able to secure a job before graduation. Only possible because of the CSCI-498 course, would not have had time otherwise. CSCI-498 needs to be advertised better, I approached my advisor with the internship opportunity asking how I could receive credits for it and he said that wasn't possible. Only found out about CSCI-498 through word of mouth. Needs more support.
N/A
I had an internship with Teledyne, a tech conglomerate that owns major companies like FLIR. Although my internship utilized an outdated an seldom used language (Delphi - a proprietary version of Object Pascal), I learned much about what the day in the life of a software engineer was like as I was allowed to work very closely with the lead Software Engineer and I was given tasks that actually reflected major changes within the system.
It gave me some experience working on a real website and what is expected from me in a job.

The internship provided so much value for me. First off, it got me contacted with people in the industry. This was vital for me in looking for a full time position as I am graduating. The next thing was getting a good understanding of what industry work actually looks like versus academia. This reassured my decision to choose this major and decide to go the industry route. The last thing was getting to understand how the business side of things work.

I was able to learn more about User Experience Design. I was also able to learn about databases and many other key concepts in practice.

My internship taught me almost everything I currently know about programming. Classes gave me a small foundation to work off of and a good understanding of data structures and algorithms. Other than that most of my skills and abilities have been learned on the job.

I believe the internship provided excellent opportunity to develop my professional skills and understand the types of things I could do with the topics I had mastered regarding the general curriculum.

It learned a lot technically about schemas, development lifecycles, IT compliance, automated testing, and much more.

A ton of real world web development programming experience. How to work on a team, structures of large companies and small start ups. How to navigate the tech world.

I did not get to do any coding in my internship but learned to work as a team and communicate in an office setting.

I learned a lot about web development and what it means to be a full-stack developer. I also hadn't taken the web development class so I learned PHP, bootstrap, javascript very quickly

My internships provided more education than any of my courses. It really exposed how much of my degree was a waste for pursuing basic software engineering. Other courses can expose different careers, but that's it. Everything still pushes you to work meaningless labor for some big company. The CS department could build amazing entrepreneurs with something other than mainstream development, but lacks the ability to encourage students to go that way.

It gave me a feeling of what working in research is like as well as working in industry. It also prepared me to apply for a full time job, without them my resume would have been lacking.

It gave me a chance to interact with people in the industry and learn from them.

The internship that I completed provided a lot of experience in the field starting from scratch and knowing exactly what the process is and what it takes to develop a solution. I thought it was a great opportunity. I enjoyed it. I found myself to be a significantly better programmer than before and found I experienced less imposter syndrome.

My internship provided real-world working experience, which was very valuable. I learned how to work in an agile environment and how to adjust to learning completely new technologies on the job, as well as how to work in a multidisciplinary team.

My internship provided me immense value in how systems admin work alongside development teams. I learned a ton of new skills and protocols within a moving company which built my teamwork and communication skills.

I learned current machine learning methods and got to use large models. It was interesting to actually get to deploy models at scale. I used cloud computing. We also used a little bit of git.

My internship was a full time job that I've held since junior year. I started out as an IT support specialist. This helped me learn common issues that users experience with computer hardware, and software. A lot of things aren't intuitive for users and I believe this gave me valuable UI/UX experience before I realized it. Last fall I was promoted to Software Quality Assurance I, where I was responsible for testing software as it was released and enhanced. I was able to find some bugs that our developers hadn't considered while coding, which I will carry on into my development career. This will allow me to program and think ahead about potential issues I could be creating. Recently, I've started doing software development. This has taught me how to create API calls and do database queries in React webpages. These are things I never learned in the CS program at MSU.

It gave me a taste of what a career in computer science would be like. I learned a ton about software design and new technologies too.

showed what a job in the industry would look like, allowed for me to focus my skills in a particular area.

My internships biggest impact was that I learned how to work in a team based environment. The project I was assigned was mostly a solo project, but I got to experience working and an Agile development team. I think this was by far one of the best things I could have learned and I think Agile development needs to be taught in classes like software engineering. Not just the topic, but actually make the students participate in a mock version of it or something.

N/A

Helpful look into the professional world/ what skills are needed to get a job and do well. Learned how to use GitHub. Learned how to work in an agile environment.

It gave me experience in getting a company set up for government contract work. Although not necessarily CS related, it's still been very helpful as I learn different technologies available to companies.

It showed me what working in the real world was like and gave me applications of the concepts I had been learning.

It provided me an understanding of html and css

Both internships that I've participated in were very helpful in giving context and hands on experience to the topics I learned in class. It was also very helpful to learn a lot about tools like github or how the more business side of computer science projects since they aren't really taught in any classes. It was also really helpful to work in a company team since the structure is pretty different from just a group project team (having a supervisor and set/consistent schedule etc).

I have just now begun my internship, so I cannot yet speak to this. It has provided value to me in the form of wages, if that counts.

Industry experience and working with a team.

N/A

It taught me real, industry knowledge.

N/A

To be blunt, it provided me way more practical knowledge than any class did by a wide margin. When I came back to school this semester, it was a night and day difference. I was more efficient, effective, and capable at understanding new concepts and getting through coursework. Quite frankly, it feels like students who get an internship end up being miles ahead of students who don't get one.

Gained experience and guided my decision to choose to go to grad school.

I think it helped me figure out what I want to do in computer science. However, I think that in my case I did not learn much and it never really challenged me. It also did not lead to a full-time position because they do not have anyone in a full-time position on the programming

It was incredibly valuable. I gained hands on experience in the design process of software and also learnt great coding practices.

Incredible value. I learnt a lot of valuable skills around good design and coding practices.

The value it provided was how to use GitLab and work productively as a group.

My internship was very helpful in that it supplied supplementary skills that I was unable to get from coursework. It also helped me figure out what I wanted to do for a career and taught me skills that helped in future classes. I learned Git from my internship before it was necessary for my classes and I was then able to help some of my classmates understand Git. I also got to learn how to work with many different people over a period of time and help to train them for our work processes.

I interned at Zoot Enterprises in the summer of 2020 and it was valuable to see enterprises implementing sprints and using tickets to manage their projects. I came away with a better sense of project management. (This wasn't a hard coding position, so I can't say I improved my skills much, but I loved the work).

Ah, money. A reduction in imposter syndrome after spending hours with senior-level engineers trying to get things in a working state to run anything at all. Some further specialized training in the field I think I want to master (infrastructure & DevOps). A sense of self-worth after being crushed into the dirt while looking for jobs for over a year because I need money to survive.

My internship helped me learn how to navigate a remote workplace, as well as learn how to have a productive workflow.

I have participated in two internships so far. My first internship taught me about Android development in Java and was an excellent learning experience. My current internship with Oracle is teaching me C/C++ database schemas, advanced git and all sorts of other backend skills.

I got an internship at a company here in town. It provided a lot of real world experience that I feel like theres no possible way to teach in a CS degree. Also I learned a LOT about networks and endpoints, and AWS.

A real look into what working in the industry will look like, granted it was a very niche company. I returned for my last semester less hesitant to work in groups, and when I did work in groups I worked better. It also confirmed what kind of work I thought I wanted to go into.

Internships were the single greatest aspect of my undergraduate degree. The skills and knowledge I got from doing internships are invaluable. It allowed me to network, open up new doors and opportunities, and get experiences that wouldn't be available to me had I not done them. Getting to apply what I learned in school directly to an internship during the summer and vice versa was an amazing experience and really helped to solidify my confidence as an undergraduate CS student.

Q9 - What are the strengths in the current CS curriculum?

What are the strengths in the current CS curriculum?
marry ann, jhon paxton, carson gross. all teach important classes that I think helped me become a great programmer. also the Algorithms classes are good ones.
Well rounded in application and theory.
In general, the CS faculty are excellent, and this is especially true regarding the tenure track faculty who are each (for the most part) teaching courses related closely to their research. I think there is for the most part a good diversity in coursework, which is well suited for most computer scientists, and especially the upper division electives are almost without exception extremely engaging. Additionally, the professors who encourage undergraduate research do so extraordinarily well.
test
There are a lot of options that a student can choose from in the upper division, just a ton of freedom.
I love how MSU seems to focus on making students that know how to program and not just the academic theory behind it! In a similar vein, there seems to be a huge push for internships and software factory jobs, which I think is really phenomenal.
I think that the biggest strength of the current curriculum is leveraging the learning you get from projects combined with how people tend to do better and be more interested in projects they can choose and are truly interested in. This is really through the list of optional upper level major specific classes we are allowed to choose from.
Theory
Location.
The required courses are valuable and the progression is well paced
It has a wide breadth, students get exposed to lots of different languages.
The Professors
While I think the 127-232 sequence could be condensed into less classes (or at least a faster paced option could be provided as I feel it could done in just 2 semesters for a good amount of the students), the content is generally good & provides a decent baseline in just how to write code and begin to think like a computer scientist.
average compared to rest of US
Great Facilities. Good job keeping up with current trends. Allowing flexibility in technology stacks and focusing on programming principles.
Programming and teaching students how to teach themselves.
The projects in Carson's classes. Teachers are willing to help and make themselves available to students.
Overall the CS department provides an overall great curriculum that prepares students for the real world.
A collection of teachers who excite learning in students
Lots of programming early.
Professors are very knowledgeable about their respective fields and convey that information well.

Python and Java make it super easy with the projects we have done to get into the door at a job interview. I would say project oriented upper division is definitely a strength.

Our helping center helps a lot for students who are new to programming, as well as students who need help in advanced topics.

Carson has real world knowledge that most other professors do not seem to portray. He helps display the importance of concepts in class that will apply in the field.

Great coverage of theory, and low level aspects of computer science (ie data structures/algorithms, byte code, compilers, etc..)

There's a lot of opportunities to learn different languages and improve at them.

The current CS curriculum covers a broad range of topics and I believe that helps students become well rounded CS professionals.

I think it is pretty diverse and it covers a lot of different areas

Strong development of fundamental skills and techniques along with interesting courses that don't focus on the same things.

Heavy on backend, giving a strong foundation for development

Most professors really care about their students and organize the class in a way that is good for learning. The basics are taught, albeit could be better. Most classes provide meaning to CS as well.

The current curriculum prepares students very well for many positions in the realm of computer science, and I feel I am graduating with many skills that I will continue to use in my career.

In the current curriculum, the availability to branch out into other fields is a very valuable tool for exploring what people want to do.

One strength of the CS curriculum is repetition. I am confident in my skills with Python, Java, and web programming. Through the courses I took, these concept were taught several times and I learned many powerful skills using all of these languages.

Web development, databases, data structures and algorithms.

MSU's curriculum provides a great CS and programming base to build upon. MSU's professors are great at what they do. All the nitch field classes are valuable because of that.

I think the faculty and the staff is the greatest strength. All my professors were very knowledgeable, excited about the material, and willing to help with any issues I would experience.

Professors were amazing and willing to help out as often as needed. I felt like I gained a well rounded education.

I think that the build-up for the first two years did a good job setting up me. I also really loved the flexibility in upper-division electives that I was able to take what interested me rather than just following a stricter flowchart like other engineering disciplines. There is also a lot of intradepartmental support that I felt even if I didn't also utilize it, I never felt like I was alone.

The strengths are pushing students towards a basic path everyone is looking for. They have a copy paste system of creating the same result in students with differing backgrounds.

All of the CS faculty is extremely responsive and helpful. The staff and students provide a great community to find help/resources for anything you need. I was very pleased with the structure of most of my classes and though overall the department had excellent teaching abilities. Advising and planning my degree was smooth and painless!

Less so the curriculum but I think that courses taught by Carson Gross are great.

The openess in the order of completion is both a blessing and a curse. While it is nice that we can tackle any classes after 232 and 246 in any order we want, it also can easily lead to very unbalanced workloads semester to semester with students unintentionally taking only easy classes one semester and only hard ones the next.

The breadth of the curriculum is very strong.

Most of these professors at MSU are incredible. They teach really well and have challenging assignments that develop growth in the field. To the point, I don't recommend the CS curriculum to people who I don't think are capable of completing the program. It's hard and time consuming and it should be. I want people to drop out of the program. No free rides for anyone.

Definitely theory related topics. You will leave Bozeman with a strong understanding of some difficult concepts.

The current curriculum provides a strong base of theory for CS. It also provides a good understanding of data structures throughout.

The CS Curriculum provides a wide base of knowledge and experience of different languages and tools. Certain CS professors are also very good at exposing students to many different types of real word situations you may face in the industry.

The availability of professors and TAs for assistance, as well as the number of valuable electives and educational opportunities

Strengths would be intro classes, data structure, software engineering, and web dev content.

I like the number of electives. I get to choose my own path and what areas to focus on. Obviously in the last year with loss of faculty there were fewer options, but I still enjoyed it.

Lots of knowledgeable teachers, with good experience in the subjects.

Professors and Department Head.

The current CS curriculum does a great job of covering theory and offering a glimpse of different areas of computer science. There is a wide variety of classes that I feel allows students to dabble in different areas. I also love that there are special topics classes offered most semesters and that those topics change each time. The department does a great job of communicating deadlines for registration, graduation, and other important events. I also really enjoyed that emails were sent out each semester regarding which classes would be offered for the coming semester. All the general advisors do a phenomenal job of guiding students in the direction they need for their CS career at MSU. I never felt stressed about not knowing what classes I needed to take and when.

I think the early-level classes are done very well (the into to Python class particularly) and they really helped me get hooked on programming things. The higher-level classes get more segmented and did not feel as impactful as some of my early classes.

I found that heavily encouraging electives allowed to pick up a variety of skills and find what specific field of CS interested me.

I think we are given a solid foundation with the algorithm and data structure classes. I think there are some very interesting and useful elective classes too.

A broad overview of concepts.

Good at teaching programming and other essentials for a job.

I think some of the current strengths in the CS curriculum are the intro classes. I think they set me up nicely for the more complex classes later in the curriculum

the main strength in the CS curriculum is the verity of upper division electives that a student can take and getting a feel for all the different things that are out there in the world of computer science

the variety of upper divisional electives so you can take what you want.

Having hired Carson Gross considering that all four of the courses I have taken from him have proven to be the most enlightening and rich learning experiences I have ever had in a CS course.

We get a wide variety of exposure to different languages and topics.

Professors from all different backgrounds

I think the strengths are that it is relatively diverse when it comes to the course material. I feel like I have a basic knowledge of a lot of different areas, but not a lot of knowledge in one single area.

Professors available for office hours, focus on data structures.

Students come out of the program as strong coders.

Presently, the curriculum provides a wide range of topics within CS. I think that's really good as it allows students to really find their niche. We also cover widely used languages (C/C++, Java, Python), which is important in today's job market.

The variation of electives is a strength in our current curriculum. Having the option to see many different sides of CS is awesome!

Coding work environment

The strengths are that many of the professors are passionate about teaching and research. This makes a huge difference in the way that I learn. When professors demonstrate enthusiasm in what they are explaining I find it much easier to pay attention and retain information.

Most of the teachers are awesome. It seems like there are lots of opportunities for jobs / events / inclusion. Some of the courses are outstanding and most are pretty good. Sharlyn is awesome. The CS success center is quite nice.

A very fair balance between Learning the field, Theory, and Practical courses

I think the courses build off of themselves really well and provide a very full understanding of computer science. I also think professor put emphasis on bigger projects instead of busy work which was really helpful for truly learning the material.

The current CS curriculum has a very good learning curve and a wide array of interesting 300- and 400-level electives.

Professors

Most courses require students to work in groups. This is likely accurate to industry and is good preparation for the future.

Large market demand

Currently CSCI 232, when taught by Mary Ann Cummings. She really cares about the students and has her class designed in a way that everything that she teaches is something that we will need in the industry.

The flexibility to allow the students to choose what they would like to complete.

The strengths in the CS curriculum are the many varieties of classes that teach various coding practices/languages that give us a wide variety of experience to learn from.

I feel the current curriculum teaches the basics and theory of CS well, and builds a decent foundation to build off of. It also provides decent opportunities to get introduced to other sub fields within CS.

It is in great demand.

Logical based thinking and problem solving. Given a problem and enough time if I fully understand the concept, I can come up with a solution to it.

The variety of coding languages we use. The amount of coding we do. Assignments that can be used and applied to real world software development.

The current CS curriculum does a great job building a strong foundation in programming and that the next class uses information that we learned in previous classes.

its very well rounded in the topics it covers

Lots of options for upper division classes, you could learn about nearly any specific type of CS field you wanted to. I also really enjoyed having an online option for most classes, lots of my professors would record their lectures and I could watch it later if I missed class due to work or something else coming up. In certain formats I felt like I learned a lot more being able to watch and pause videos. The advisors were always very helpful and easy to get in contact with.

Great overall organization and some really great professors

It is very good at teaching you Java, how to debug and think through problems, and giving you the ability to study different areas of CS if you want with electives.

I think it does a good job of dabbling in a wide variety of concepts. It also does a great job of focusing on the project and applying what you learned via the projects over the course of the semester

Preparing students for real-world systems.

Carson Gross, Maryann Cummings, and John Paxton and what and how they teach is crucial to this department. These three professors have taught me most of the most valuable things I have learned attending MSU Bozeman.

Teachers are really helpful and there are courses that span both theoretical and more practical sides of CS - both of which are incredibly important in a career.

Professors are mostly very helpful and the curriculum has a strong focus on developing real-world/industry skills.

Many of the courses give important overviews of what is to come as a developer. Any of the classes with well-thought-out projects.

Our current CS curriculum is pretty good at teaching us theory so we can fairly quickly understand different programming languages and their structures. I would also say one of the greatest strengths for the CS curriculum are the professors who care about their students' success and have done so much for us in terms of accessibility for help and patience in answering questions and stepping us through problems.

I really appreciate the rigor of theory in this curriculum, but it isn't overwhelming. The breadth of electives is exciting and wonderful! The interdisciplinary option allows for a lot of space to try out different courses and gain some practical coding experience, too. Though this isn't directly related, there are plenty of data science electives that made it easy to graduate with that extra credential.

Clearly you want to stay language-and-vendor-neutral, and I think you do a good job of that. It isn't the practical, vocational sort of degree I thought it would be, and you know? I'm glad I know more theory. Most everything can be self-taught, but I have something that's harder to learn on my own, and hopefully can write prettier, shorter, and more-optimized code than my self-taught equal because of it. I just have to learn the language first... I think you have some good faculty who are passionate about their area of specialty. Keep the upper-division electives coming! There are so many paths to follow as the only computer degree that isn't CE, so it's great to try some of the ones that interest you on for size to see where your own passions lie.

The current curriculum provides many different areas of CS (i.e. the data structures algorithms courses vs ethics vs sw engineering vs computer systems), which makes me feel like I have a solid understanding of the basics that make up computer science.

I believe that the curriculum as a whole is very strong, I have very few complaints from actual CSCI courses.

The content- I am walking away from this program with a lot of confidence that I will be properly equipped for any job I have. I feel like I have a wide knowledge of topics, and the capability to dive deeper into any more specific field that I choose. Also, it is very easy to get help and resources. Also, most of the faculty is stellar. (Give Carson a raise)

Professors that bring their professional non-instruction related work into the classroom. For instance HTMX by Carson Gross, or Dr. Sheppard's research. I also loved interacting with graduate students, going to those seminars hosted by Dr. Mumey were a game changer and so were courses in which graduate students and undergraduate students learned in parallel like CSCI 451 or Comp Security. Of course there are faculty members who are strengths all on their own like before mentioned Carson Gross, Daniel DeFrance, Dr. Sean Yaw, Dr. Cummings, and many more.

Well-structured, plenty of opportunities, and great support from CS faculty. I really feel like I was supported by all of the faculty throughout my undergrad. Whether it was getting to go to conferences, getting help with résumés and career advice, or learning of new opportunities, I received plenty of support from the faculty, and I will always be thankful for that.

Q10 - What are the weaknesses in the current CS curriculum?

What are the weaknesses in the current CS curriculum?
the focus on outdated information from time to time.
As a tutor for CSCI 132, the students I see (and depressingly some of my peers) are coming out of that course without any true understanding of how to write in Java well, they came out and did NOT understand polymorphism. CSCI 132 needs to be a primer to ESOF 322, since it is critical for developing any good program.
I think the early coursework was the least valuable for me, generally speaking. The upper division stuff is excellent, but it takes some patience to get there and the lower division stuff in particular I thought moves pretty slowly. It'd be cool if there were more accelerated/theory based options.
test
Some classes are definitely easier than others, but this could just be because some concepts are easy once you grasp the requisites.
I think any weaknesses stem from an evolving CS field. I've been a part-time student for 7 years and I've seen it try to be more practical and also more focus on data science. That is great, but sometimes classes suffer when they're brand new or have brand new professors.
I believe that the program has two big weaknesses. Firstly there is a large amount of rampant cheating unfortunately which means that certain individuals find the work very quick and easy in classes by using past code from other people as models while other individuals find the deadlines brutal as they are building everything from scratch creating a gap between the learning and difficulty level of the courses incentivizing individuals to share their code in order to make classes easier. Secondly I would say that though it is not a fault of the instructors, most individuals in Computer Science very much struggle with the conceptual classes specifically the courses on the topic of determining algorithmic efficiency and similar topics.
Not enough time spent programming like we do in compilers
Course availability.
Not enough CS electives offered each semester
I felt that, in general, the courses were too heavy on theory and I didn't get enough experience with programming or working on applications.
Outdated ideas in part of the curriculum
I found in my experience with internships that coding for actual projects felt like a completely different task than anything I encountered in my coursework, and often course content didn't match well with actually implementing anything useful. Additionally, there is a lot of arbitrary degree requirements that I don't think really exist in other degrees, such as the odd science course requirements, with the 6 credits needed from physics or bio type classes. It makes no sense, and forced me to take extra classes because some that I already had weren't arbitrarily added to the list.
too many incompetent professors
Limited availability of classes depending on semester.
Many of the CS professors are borderline too smart to be teaching. A fantastic example: Hunter Lloyd. Hunter Lloyd has to be one of my favorite professors, I have never been to or heard of better office hours. If I did not understand something, an hour with him would result in me being able to teach it. However, he is not great in front of a classroom. He is hilarious, but it's clear that he has a hard time understanding that someone might not

understand what he is talking about. To me it was clear that he would frequently skip over things because they were simple to him, these things were not simple to the entire class.

In some of the freshman/sophomore classes there's a learning curve and students are expected to have pre existing knowledge.

There seems to be a lack of overall consistency with professors. I know that each professor teaches in their way but the lack of consistency leads to more stress and less opportunity for learning. Implementing a standard, even at a single course level, that created a concrete plan such that all assignments are due on "x-day" at x-time each week greatly reduces the confusion felt by students.

A few teachers were not great at the class management portion. A few teachers I had especially at the end would leave homework ungraded for a very long time and would fail to tell students crucial info.

Not a lot of system design topics.

By the time I got to upper division classes I didn't feel like a strong programmer at all. I never felt like I was shown an abundance of actual coding or the right way of programming something.

Nothing in the curriculum that i can think of right now.

The weakness is that, for some high level courses, such as machine learning, advanced algorithm etc, if we have a low-medium level course that gives us intro for those topics, it will help us a lot to make that jump from 100-200 level algorithms to the upper level courses.

It seems almost impossible to receive any help from professors (at least during covid) for more than 10 minutes. I think professors should dedicate an entire class period once every 2 weeks to just allowing students to work on assignments and be available to answer questions.

Not much coverage of practical skills, or modern technologies (both of which are needed to get a job). Advisors and auditors aren't knowledgeable on degree or department logistics, can be difficult to get in contact with.

It doesn't prepare you very well for making large, long term projects.

The current CS curriculum can also be too broad, in some sense. Especially when the push seems to be for students to come out and pursue careers as Software Engineers.

It would have been cool to use some more specific languages and frameworks for certain classes.

Some courses feel not as developed or not necessary on top of a couple course topics missing.

Cybersecurity being missing, Data structures and algorithms classes struggling with teachers, and discrete structures class also struggling with teachers. The current structure of the degrees really tends people towards backend. Meaning if people want to be more front end, you have to REALLY pay attention to what classes you are taking in order to make sure you get the information you need before you graduate.

There is a very large disconnect between industry and MSU. Many classes are taught in a way that only academics would tackle a problem. If you were to ask a Senior Developer in the industry they would go about almost every task differently than how we are taught.

I believe that more crossover with hardware could certainly improve student's understanding of computers in general, though these types of classes are likely more related to another degree or area of study.

The main weakness in my opinion is the focus on learning various programming languages without focusing as much about gaining experience coding in different scenarios for different projects. This is only really covered in the upper division classes. Another big weakness is the variance of upper division classes offered.

One weakness in the curriculum that impacted my college experience were teachers. I felt that the foundational classes I took like CSCI132 and CSCI232 weren't taught effectively and affected my ability to succeed in later classes. CSCI132 and 232 were both taught by Hunter Lloyd when I took them and I feel like I didn't learn enough about Java and OOP as I would have liked.

Cybersecurity and networks

Some classes should modernize what technology they teach like webdev

The CS department needs to be consistent with literally anything. There is no consistent way of finding class information, some professors use thier own website, some use d2l, some use GitHub. There is no consistent way of turning in assignments, some use grade scope, some use GitHub, sometimes you uploaded it via ssh, sometimes you use d2l. Every professor uses different IDE's even if they use the same language. Github needs a class or more explanation before being implemented in classes. Basically I would spend more time with stuff other than my code rather than the code itself for issues because nothing is consistent. I have a class where I have to do assignments on Jupiter notebooks. We were never instructed on how to convert the notebook to a pdf, you need to use LATEX. We do not have a LATEX class or way of learning. Several students were confused on how to convert this Jupiter notebook to a pdf so they could submit to grade scope. If a professor wants something submitted in a special way on a special website not sponsored by the school, why do we just have to figure it out, the professor should show the students how it's done, even if it is easy. Moral of the story, BE CONSISTENT. I realize d2l sucks but if it sucks that doesn't mean professors should just go around using whatever service they feel is best. We should improve d2l or get a better service. Sure, code is harder to submit but that really doesn't matter. Create a standard for the CS department on where to submit things. For instance, just make everything due to gradescope or something else if you guys don't like d2l. Don't make it more confusing on the students because faculty is too lazy to make the service that the university pays for work. It is literally just laziness from a students prospective.

Needs more hands on coding. I felt like a lot of the classes skipped over more hands on learning and got to theoretical.

I wasn't always able to take the classes that I wanted due to the fact that they aren't all offered every year

The CS curriculum lacks an environment that gets students thinking outside the box. Also mental health is ignored. Especially with covid, students were forced to a lifestyle with screens and no human interaction. Depression and anxiety has gone through the roof and CS neglected to promote healthy lifestyles.

I would like more emphasis on industry standards/practices in course materials, or possibly more emphasis on job preparation. More opportunities for specialized field training would be good too. I don't feel like I had enough chances to try my hat in a wide range of specified CS topics, which makes navigating the career market a bit overwhelming.

I think subjects that are based in math could include more of that mathematic background if students were required to take more math courses.

The focus on electives only works well when there is a wide variety of electives that are consistently available.

I think if the courses were more connected and they built on each other, it might allow students to understand how to implement and apply cs concepts better

A substantial number of the students in the CS curriculum are worthless. They cheated their entire way through the program and are sometimes even graduating with honors, despite being entirely clueless in the field. I spent an incredible amount of time doing someone else's work in a CS class just for them to ride on my hard work and success. Nothing is more aggravating than spending 30 hours on a project in 2 weeks just for some worthless student to submit a solution from the internet or get a free ride from your hard work. In my junior and senior years, I worked with 2 students who were up to my standard, and 1 student who was phenomenal. Other than that, I can confidently say that I was continually let down by morons. The professors' input was ridiculous aswell in summary: This is what the real world can be like. Not submitting a solution to the problem will negatively impact your grade. Additionally, some of our 3 credit classes take a lot more than 9 hours a week. CSCI 446 and 466 especially should be considered 4 credit classes. A major weakness of our current curriculum is that we kind of are off topic compared to employment opportunities. It sometimes feels like were too late for entry level java jobs, and too early for python.

Some of the early class seem to be more difficult than is needed. I fear that it scares off students who would enjoy Computer Science if they would just tough it out.

There might be too much theory based curriculum and not enough raw coding done. Limited software engineering type courses.

I think there should be more reviewing of code and feedback. I never really recieved any feedback on the code I wrote and submitted throughout all of my classes.

Some courses aren't offered every semester, students can miss out on key electives if they don't have time to take them during the semester they are offered. Lack of GitHub training in lower-division courses, which is expected knowledge in upper-level courses.

Weaknesses would be auxiliary math courses including linear algebra, discrete structures, cs theory, etc.

Lack of math focus. I understand that there is only so much math that can be required, and that most CS students would strongly dislike added math. But i do think that it is very important and the theoretical background is what makes it computer SCIENCE not just programming.

I think that a lot more computer science classes should be taught more hands on. Like doing more in class problems and going over them instead of just getting talked at for the whole class. Labs were kinda the bridge between the two, but it seemed like after 232, labs didn't exist anymore.

Some of the classes are very similar

The current CS curriculum fails to offer students the option to specialize in certain areas of computer science. I feel that I am leaving MSU with a very general understanding of computer science and I'm unsure of the direction I want to take my career. I also feel as though the curriculum is aiming to create software engineers. There should be more in depth classes that allow students to deep dive into different areas of the subject. It felt as though all my programming classes didn't really challenge me. I am leaving not knowing how to call APIs, create database queries, and other important backend programming techniques. I have also received minimal education on mobile development and frontend development. I know that many of these changes can only happen if the department grows in students and staff, however I think that with these changes in the future, this program could be one of the best in the nation.

Too many group projects. Group projects in college (or any level of education, really) are often extremely different than any group work done in a professional environment and the level of learning they provide in regards to communication and collaboration skills does not outweigh the annoyance and struggle of working with other students who may or may not be motivated to work together.

I would have liked the opportunity to fully learn a variety of more popular programming languages used today, such as Rust and Kotlin. But I understand this may be hard to provide in the curriculum and Concepts of Programming Languages provides enough to learn the basics of most languages.

Some of the classes I took were not very well structured and some of the class content was fairly outdated.

Outdated requirements. Literally, no one ever uses Java SWING library to make GUI.

Very dependent on professors, some important detail is lost depending on who teaches a course.

I think a weakness in the current CS curriculum was went we went online. I know it was hard to do online for the instructors and some did it well, but others didn't have enough engagement in their online courses that it was difficult to learn for that year or so.

One weakness that I think there is, is in the lower level classes depending on the teacher the experience of the class could be completely different.

lack of integration to git. I was always confused when a teacher introduced it. making resources more known such as the help center.

Limited course variety in electives, I felt as though I was forced to take a bunch of courses that I had no interest in to fill out all the electives I was supposed to have. Also, there should be more professors like Carson Gross.

That there are not a lot of options to fill out cs electives/directed electives I felt shoehorned into many topics that I had no interest in.

Poor student support

I think the CS curriculum is in desperate need of professors that can teach industry methods and practices. Like I said above I have a decent grasp on a lot of topics, but at times I feel very incapable of applying them to a workplace. Sure I can write code and decide what algorithms to use, but what does that look like in an actual workplace? How do I work with a team and write code so that it makes sense to the people I am working with/for.

Git not being taught, lack of relevant languages in program, c seems a little outdated when entering the real world.

Not enough diversity in computer science electives. MSU is not on the same playing field as other universities computer science departments.

Although the department provides a large range of courses, usually only one or two classes exist in the related field. For example, we only have one class in computer security. There's only one class on operating systems, one class for AI, etc. etc. Also, I would honestly just say the CS department needs more staff, period. I think more staffing would drastically help improve the CS curriculum at MSU.

EGEN 310 could have been a really cool class but was not. Working in a multi-discipline group is important but that class is a weakness in our current curriculum.

Additional help. There are few resources for students to use to improve themselves

Computer science is such a broad umbrella term. I feel like many students would benefit from a more specialized course curriculum according to their desired career. I have ambitions to become a UX researcher following graduation, though there are very few courses even available to me for this field. I find many of the classes I took were extremely challenging, time consuming, and not useful for my future career plans.

Although I have no other CS curriculums to compare to, I think MSU's is quite good. However, this industry is changing very quickly and some tools are going out of use while others go mainstream. I think it is very important to keep up with the times and not get bogged down in the way it WAS done.

I do not think there are many. I personally like the ordering myself.

You can't really focus your interests on specific topics until the three semesters with professional electives and I think it would be cool to be able to do that sooner in the program and take a couple electives that build off of each other.

The current CS curriculum includes a lot of material which is irrelevant or not useful; for example, WRIT 221, EGEN 310 include material which I do not believe is useful in a CS career. There is also material which is not covered but should be, such as the basics/essentials of using git.

Breadth of topics

I would like to see more courses offered over the summer and winter mini-semesters.

Easy to get started but difficult to specialize

Certain Professors to be completely honest. There are certain professors who have an attitude of "I am too smart to be teaching a lower level CS class and would much rather do my fancy research instead". It is extremely demoralizing to be on the receiving end of that. It also doesn't give students the strong base that they need.

I did not acknowledge any weakness in the current program.

Going off the strengths in the curriculum, I would say that some of the upper division classes that are newer a can sometimes be a little lackluster and not exactly meet my expectations of what I would be getting out of the class. This is all hinging on the fact that these classes are new to the college so everyone is still trying to learn how to get the most out of those lessons, so I would say that it is a weakness that we are trying to work on currently.

Actual practical experience seems to be limited. Courses like EGEN 310 try to give students industry experience, but that course did essentially nothing to prepare me for actual industry work once I hit the workforce.

Easy to get started but difficult to specialize.

Coding without having the best language to code in. What I mean by that is there are the different languages I know like Java, C++, Python, but it's like a Jack of all trades and a master of none. It's difficult for me to switch languages multiple times.

Lack of security. Need more courses about designing software. Lack of game development.

I do not think there are any weaknesses in the current curriculum. I thought each class had as much importance as every other class had.

it probably focus's too much on some of the math and the theory and not enough on more practical applications

It would be nicer if there were more 200 or 300 level courses that aren't as difficult but allow someone to get an intro into a particular subject.

Course descriptions are not as helpful as they could be. The nature of computer science is that there is a lot of variety in what a class is like, ranging from theory-based to more implementation-based. These differences are further exacerbated by teaching styles and approaches. For example, Carson's classes don't deep dive into underlying concepts, whereas Dr. Fasy or Dr. Sheppards classes do. I think it is great to have this variety, but students could be made much more aware of these differences in course descriptions, which might allow students to better pick their classes and improve their experience.

I don't know if I know of a lot of weaknesses in the curriculum. A potential weakness is not a lot of real world experience and practicality, as I have heard a lot of people say when you step into a job afterwards, you just have to real learn a lot of things.

I think that its weakness is that it does not do a great job of preparing for professional experience and getting the first job.

Preparing students for the workplaces/workflow in a CS job.

It can sometimes be hard to find assistance for programs or assignments. Some of the smarty cats tutors can't even access the material for the classes they are tutoring.

Not as much breadth of subjects to choose from.

Lack of breadth in course options. Capstone options are also limited (eg. professional option is Compilers which seems like a very specific area that only few students may be interested in specializing)

A clear definition between classes that teach you how to code and classes for those wanting to go on to theoretical work.

I would say expected use of Git in higher level courses without sufficient training using git tools in lower division classes.

I attempted data structures 232 once and withdrew; when I attempted for a second time it was much more comprehensible. I think Data Structures needs to have a rock-solid professor teaching it each semester. I also struggled with 338. Both of these were Spring 2020 and that definitely contributed, though.

It forces people to hate Java! People learn python, which is a fine language and makes sense to newcomers, then are expected to use Java to learn data structures & algorithms? No, of course that's not going to work. I spent a whole semester just becoming competent with Java first, and that's after a year of learning it in high school. Now Basic D&A has to become Java 1 and some things have to be dropped or rushed in regular D&A. Everyone is confused and rushed, and all my peers in upper-division group work only wanted to use python for everything, which I barely even knew! They had such a bad experience speedrunning learning a whole new way of thinking that they just want to go back to procedural programming for every possible task instead of considering object-oriented or any other type of programming that would really shine in that particular application.

I think some courses fail to focus on the fact that most students are going to become programmers after their four years, so trying to incorporate that more into all curriculum would be beneficial.

I would recommend adding more project based classes to the lower level CS classes which result in the student making an actual product for their resume.

Lack of instruction with AWS, too much theory, and also- the projects ive had since 132 have given WAY too much starter code.

There are a few courses that float along being temporarily fostered by one instructor or another resulting in low quality courses.

I felt like the class sizes for some of the courses were way too high. It removes that personal connection with the professor and makes it hard to individually grow. Not having the support from the department to do a co-op made things stressful and difficult as well when I shouldv'e been supported instead.

Q11 - What changes would you suggest in the CS curriculum?

What changes would you suggest in the CS curriculum?
teach more useful languages other than Java. Java is average.
Stated previously.
I think we should have a course on quantum computing, I think we should consider integrating some early coursework with the honors college (eg. 132/232 as is done with the calc series in the math department, or in the physics dept, or chemistry, etc.), and generally hire more research active faculty, especially in systems/quantum related fields where we are more limited. Otherwise, I think we are generally doing things very well, and I can't wait to see how bright the future of the school of computing will be, especially given the substantial funding the department has been receiving recently. It is certainly in good hands, and I am extremely thankful for the opportunities I've had because of you all :)
test
Not really that I can think of. Perhaps more opportunities to mingle (but you are already doing a lot to address this, especially recently)
I would make the CS theory class optional, as well as the EGEN 310 class. I would also more strictly require linear algebra, maybe in place of some calculus. I would also add more to the software engineering side of things than the two classes that are there. Maybe an entire cloud computing class? Or an entire class on agile development with CI/CD?
I would recommend keeping in place the very hybrid class structure that professors have put into place due to covid restrictions. I believe the online resources provided massively helped people in the classes as it enables people to go over old lecture material. Also I was only a student at MSU for 2 years of my bachelors degree so I can't speak on the lower level classes but I would like to see some kind of difficulty measure for classes. Not to avoid hard classes but intended difficulty of a course can have a huge affect on student success. Unknowingly signing up for some of the hardest classes all at once can be quit punishing and while certainly there is somewhat of a gauge in the course numbers, there is little use of telling difficulty of courses in the upper division classes based on these. I have had very very difficult 300 and lower 400 level courses while on the other hand having really easy higher numbered 400 level courses that were combo'd with grad classes.
None
More classes taught by Carson Gross.
Offering the classes listed in the catalog more consistently
More preparation for working in the industry and possibly different paths involving data science, security, AI, and interview prep.
Computer Science Theory earlier
Get rid of, or at least have more general elective options available, make M221 required, and allow a fast-paced option for the 127-232 sequence for more advanced students. And in general have the upper level courses be more useful (and just some more options, it does feel like there aren't many options for upper level electives in CS).
Pay your professors more so you get good ones
More reliance on online resources, to be discussed during lectures. Classes should be available throughout the year. I don't care if they are taught by a TA.
Like I said above, maybe an additional writing course and definitely a public speaking or communications course.

More foundational classes that ease students into the program.

Ive addressed my changes within previous question when addressing weaknesses and courses missing.

Better communication

Change the capstone course for the professional option. I really don't understand how writing a compiler is a culmination of all that I've learned.

After a program is due, please show students the correct answer. It wasn't until Computer Systems 366 that I actually stepped through code.

Maybe a few more data mining classes for those wanting to go more in that direction in their career.

One potential change that can be made is to add some 200-300 level courses relating to the 400+ level courses, so that it is easier for us to know and have a better understanding of the advanced topics.

Every class should be recorded. If a professor is worried that students won't come to class then they can make the recordings private. Then the students can request for past lectures if they want to review.

Put more focus on practical skills and modern technologies. Hire dedicated staff to take care of advising and degree auditing (more people like Sharlyn).

Have a class before Capstone that involves a long term project.

Offer certain focuses within the CS program. For example, offer focuses within Software Engineering, IT Technology, Cybersecurity as broad topics. Software Engineering can focus on topics like web, application, or enterprise level development. IT Technology can focus on topics like systems administration, networks, etc. Having some kind of focus that is up for the students to declare within the program would help those students develop skills within the specific fields they want to enter and they would feel better prepared upon graduation going into the field they studied for.

Maybe offer more classes each semester

Honestly, I think the newest change of the curriculum is a really good start of what it should look like. Adding Networking is good, but I do think you could eliminate/condense Social & Ethics.

I would like to see more industry standards brought into classes such as code readability, simplicity, and unit tests being taught. I think there should be greater emphasis on what is necessary to understand in the business world. I would also like more preparation for internships and a greater push from the faculty to get students to apply.

I cannot think of any particular changes I would make to the general curriculum, though many changes I have heard (specifically with a computer security course) would make an improvement overall.

I suggest looking more into what would offer students more options to practice and explore within the real life fields because CS is such a dynamic market. Focus less on specific languages and move more towards increasing what students can do with their programming knowledge and skills

Get more professors and break up the work amongst them better. I don't doubt that Hunter is a great professor, but I feel like he would be better if he only taught his upper-division robotics classes. Two more professors like Carson Gross or Laura Stanley would greatly benefit the curriculum.

I would suggest adding more APPLICABLE cybersecurity courses. Less theory and detail, more answers to how I can protect my software and follow secure development principles. Also, I'd love to see a DevOps course added.

Get rid of Hunter Lloyd, Add blockchain classes, add a more intense full stack web dev class.

I would like if a mobile app development class. I think that is a huge field that students would be interested in. I asked about creating a swift application for the independent studies class and basically got laughs from a few professors because no professors at our school are into that language. iPhones are all over the world and swift is a powerful language. The department should be embarrassed that a student want to learn a top of the line language, used by Apple, and has no foundation or support at a university. How behind are we with the times? It also doesn't have to be swift, we learn java and that is also used in mobile app development.

More hands on coding classes. Or a class to get me better prepared for a job as I feel like I have no clue what a software engineer will do in the workplace still.

I would suggest the CS program to promote a CS degree with the purpose of entrepreneurship. Any kind of self driven development that means you can do WHAT you want instead of laboring for someone else's dream you couldn't care less for.

Cirriculum changes mentioned above.

If there was a way to get through 232 faster so that you are not stuck behind prerequisites for the first 2 years would make the program more engaging, or offer other technical entry level courses. Provide a better intro to linux and git. Even out the courses between the fall and spring, the fall is stacked and the springs courses are less interesting. I also think OS should be a required course in place of security.

Widen the variety of electives and make the offerings more consistent semester to semester.

See other answers for my suggestions

At the end of each set of credits: 60, 90, 120 I firmly believe that students should have to take an exam. It should be a timed exam where they are required to produce solutions to problems exactly like how job applications work in the current market. Not even pseudocode. I want a 2-4 hour long exam where they have to write working code. This test should be conducted at the end of the spring and if they fail, give them a class during the summer where they are either taught to be computer scientists or they drop out. The way that Mike teaches CSCI 466 or Clem teaches 422 or Sheppard teaches 446(if it was 4 credits or make the projects take less time and leave it at 3) should be the model for teaching any class. Some professors do a decent job aswell but those 3 professors specifically do a phenomenal job. When a professor respects my time, by expecting me to do the reading outside of class, do the project where I mostly only have to write the stuff that pertains to the core concepts of the class, and then just teaches concepts while providing examples and proofs in class, I feel that the class is worthwhile. This University now has a lot of money invested in the CS department. If students cannot be self-motivated to achieve excellence and understanding in the CS field, do not punish others by making us drag the dead weight.

Consider requiring more classes that actually relate to CS and less topics in other fields. I understand this is college and for some reason everyone is required to have credits that don't relate to their topic in any way but it tends to add to the fatigue that students feel especially in their final year.

It seems like the courses that focus hard on coding are towards the beginning of the curriculum and then you go into a lot of theory based stuff without much coding. Then you go back to large project coding. It feels like there should be some in between classes that are for intermediate level coders that are focused a lot more on the writing code side of things.

I would suggest that certain professors try to teach using more of a real-word industry approach as opposed to a scrictly academic approach

Require CSCI 440

I would suggest diversification of classes sooner towards someone's junior year. Having more options for cybersecurity would be my biggest one. I don't know if I see a reason for that many extra math courses required for students. I understand if someone if looking to go into research, but for the vast majority I think would say that something like linear algebra provided no aid in their progression as a cs student.

I think a more flexible capstone project like what is in the interdisciplinary capstone or other Engineering capstones. Where it is a year-long project. I think that this would help students gain more experience on large projects, using latest technology and libraries, better teamwork, and a nice portfolio

Not that i have really any outside experience in computer science, but I think that there should be more stuff taught like building web apps / mobile apps, more robotics (maybe thats more EE) and classes like that. Maybe less classes theory related.

More preparation for the workplace.

When I first started as a freshman, I started in 127. I had a strong programming background from coding classes in high school, however I noticed a lot of my peers struggled to grasp the basics, and I noticed this even more so when I later became a UCA and TA for the class. I think it would be very beneficial to require 107 as a prerequisite for 127 or the option to test out of it. This will help create a stronger foundation for future CS students. I really

wanted to get into cyber security and I felt that the current security class touched the basis and very outdated exploits, I would have loved more classes focused on security that provided a more deep dive into writing secure code and finding exploits in modern code. I also think a course that focused on math based programming could be very beneficial for students. I'm never sure when I should use math vs programming tricks when working with code at work. Planting the seed for this type of thinking could create very strong programmers at this school.

I would suggest some way of requiring students to take an internship or at least gain 3rd party experience. I didn't do this (as it wasn't required, and I worked my way through school to pay for it so I was often busy working) and now that I am job hunting the lack of extra curricular experience is hurting. I think a course that requires students to apply for and gain an internship or some other type of on-campus extracurricular experience would help those students who don't realize the value of internships early on.

I would've enjoyed more hands on programming assignments in the lower level CS classes.

Change the introductory course back to Java. Learning python first sets you up to fail. CS is NOT about coding, it's about solving problems with code. I feel like they switched this intro to Python to keep higher retention. But now sophomores think they can apply the loose syntax requirements for python to Java. They need to learn to solve the problem of learning Java and they will better understand codings. Also, make 305 a requirement at lower level of year. Like sophomores should take it

Remove EGEN 310, combine a few of the less intensive classes, require some electives that teach widely used tools.

Changes I would suggest in the CS curriculum are to have advisors actually advise for courses. I had an advisor at one point tell me not to take a class because it was a 400-level course and I was only a junior, but I wish I had taken that course instead of listening to them. Also, I took CSCI 338 wayy after I took CSCI 246 which was difficult

the only thing that I would change to the curriculum is having more consistency in classes that are taught by more than one professor and adding a wider verity of topics and classes to the curriculum allowing students to take an even wider verity of topics and learn more skills at the university.

I think the new requirement of Computer Security is a good thing, continue to move with the times, I also don't think there is a need for both 132 and 232. I think 1 semester of java is more then enough, majority of upper divisional classes used C, Python or HTML/Javascript

I had some very negative experiences with some of the CS professors, some of which were very condescending and bordering on misogynistic. I would highly recommend looking for more professors like Carson Gross because it cannot be overstated how impactful he was in my education. And, more course variety.

Be more focused on teaching things we will actually need to use in industry, again with the GitHub thing, theory is nice, I guess, but we need more practical applications.

Reevaluate the curriculums for 215 and 246 as well as regularly auditing professors

I would suggest that students are introduced to topics such as software engineering earlier on in their career. I don't see the point in hiding it behind 132. In fact I think it would be a great class to take at the same time as 132 to help students understand what they are doing and why.

Remove c and add GO to the the curriculum. Add Git into CSCI 127 Add a UX design focus in the degree- similar to the data science minor

Make an intro class that teaches students how to use github, the command line, IDEs, what a virtual machine is etc.

Remove EGEN310R from the required courses in the CS curriculum. Although I'm presently taking the course, it just does not seem all that applicable to us. I would make systems administration a required course for all three degree programs.

AP CS relates much more closely to 132 than 127, both in content and language. I never learned python through a class but aced 132 without going to class. I think it would be valuable to change which class AP CS relates to in the curriculum. Additionally, I think it would be cool to show different paths for different careers. If you are interested in Software Engineering look at this worksheet or if you are interested in AI/ML look at this worksheet. Since sometimes only one elective CLearly relates to a path but many other ones are equally as helpful.

Just tell people that there is a lot of non-coding classes

Take the EGEN 310 course OUT of the curriculum. This course doesn't make sense for computer science majors.

Add the courses suggested above. I heard there was a 5 year master's program (in my 3rd year) which I probably would have done had I known (so maybe put that out there more). Give CS students access to the supercomputing cluster on campus (duh), maybe only upper level students - I was doing a super rad data mining project last semester and could not get access to enough compute to really make it amazing. Maybe more student research outreach. The E-commerce class was pretty cool - maybe more 1 credit industry people classes.

I think having an introduction class at the very beginning of the program that goes over the basics of computer science and shares tools that you can use would be super helpful. I felt like I had to backtrack and learn some tools and techniques for coding as I went

I would suggest that WRIT 221 and EGEN 310 are made to be optional, not required classes. I would also recommend adding lessons on git to some early CS class.

Student and faculty led courses. Introduce a course that teaches modern technology stack(s) and the course can be taught by students (paid, similar to TA) and professors.

More focus on Computer Security and possibly white hat hacking.

None

I believe that professors teaching lower level classes(only some of them needs this, not all) needs to show more enthusiasm than what they currently do. It discourages students from asking for help. Also, Professors should be encouraged to respond to students emails.

I would not change any in the current program.

Looking back I would suggest to maybe replace the software engineering class with something that might be a little more geared toward teaching computer science students specific software engineering practices they would be able to work on. When I took the class I honestly didn't have much if any coding or software experience gained from the class and it all seemed to be filler for the other engineering students in the course. Maybe pairing this class with more computer science opportunities could be another change that might make that experience more beneficial to the students.

If there's a way to help get students field/industry experience, that would be fantastic. The career fair is a thing, and there are some other ways to get connected to employers, but even if one goes through all of those, they still might get passed over. Enabling students easy access to industry work would probably be the best way to get students industry ready and would result in a massive quality spike in graduates.

None

A focus on java coding for beginners and an explanation of how credits are distributed throughout someone's plan.

More courses on security. Need more courses about designing software. More courses on game development.

I would suggest having some of the electives available more frequently than they are, I know for me it was a struggle finding electives to take the past 2 semesters because only a select few electives were offered and some that interested me the most were not available.

i think CSCI 476 should be a requirement as opposed to an elective course

More 200 or 300 level courses, more courses with real world job examples or curriculum, and a class that was more focused on computer hardware.

The increase in version control use has been great.

I would say the only changes I would give are to have more classes like the ones that I mentioned above.

I think a career development class would be very helpful. I think this course could replace ethics and have a single section on ethics instead of an entire semester. This class could help you create cover letters and resumes.

Practice coding interviews finding internships and help you figure out the type of work you want to do with your CS degree.

More emphasis on real world coding, possibly larger individual projects that we can add to our portfolio.

The focus of more programming. Maybe it is because I am a kinesthetic learner, but books can only take me so far.

Include more subjects to choose from and offer subjects on a more regular basis. Also perhaps change the capstone options so there is a more diverse range of options of what a capstone course is. Eg. the professional option has to take Compilers as its capstone, and while Compilers was a great course, there should be other capstone options to take that could fulfill this requirement, as probably only a few students (if any) will end up following a career path in the compilers area.

More availability of courses.

Make courses for those that want to be a developer that might take a project from nearly the first class all the way to the last class.

I would suggest the removal of EGEN 310 from the CS curriculum. The best parts of the class seem to be group communication and work but we get plenty of that in other classes. I believe this would remove a lot of unnecessary stress from CS students. If possible, I would also say to have ESOF 423 count as a capstone for the Professional degree option as well as Compilers. Both classes are fantastic and teach us many valuable skills and the professors of each class did an amazing job of guiding students through the course. However, sometimes students' interests align more with the ideas of ESOF 423 as opposed to Compilers so it would be beneficial I think to be able to take Compilers as an elective and ESOF 423 as the capstone, or vice versa if that is possible. Again, both classes are fantastic, I just think it would be great if a student could use either as the capstone.

I would not require EGEN310 and I would include more practical skills up-front, to habituate students to enterprise-like environments.

I think Databases needs an overhaul. It seems like one of the most important things to learn, but I, personally, didn't get much out of it. Computer Security also seemed quite outdated and not what I was hoping it would be (although I still loved it!), but I hear that did get some sort of rework sometime after I took it, so maybe that's already taken care of! Ethics seems like it could cover so, sooo many interesting things, but it passes over some of them and is filled with people who don't care and don't want to be there. I don't know the solution. Making it optional seems like a good way to create a robot uprising someday. But I can't help but feel like it could have been so much more, like a really good seminar class.

I would add an optional 1 credit interview prep course that can be taken any time after 232 is completed.

Delete 215. Add git to ESOF or wherever else it fits in. Stop teaching UML diagrams and design patterns; they are not actually used in industry.

Have a more consistent CSCI 232 curriculum. Each semester is very different depending on who teaches it, and I feel like we go into upper division classes with different skills and tools

Courses like computer vision, or advanced ESOF expose students to industry specific opportunities and fields. I know dozens of soon to be graduates that have gone through all of the courses but have yet to take a course in their area of interest and even more that don't know where they want to go after graduation. They have no idea where they want to apply their skills. I'd also give professors more flexibility if they've found a successful niche, such as Carson Gross and his remote teaching approach, or Dr. Laura M. Stanley's amazing 15 minute lectures that allow an insane amount of focus. Lastly I'd make attending at least one of those seminars that Dr. Mumey hosts mandatory for undergraduate students perhaps for a seminar related to a course. I think it would be a great way to expose students to industry opportunities as well as graduate school opportunities.

Add support for co-ops, reduce class sizes perhaps by implementing more sections, and offer more courses like those I suggested above.

Q13 - Please comment on your advising experience. How could advising be improved?

Please comment on your advising experience. How could advising be improved?
I think it was very nice.
An advisor should also have the initiative to communicate with their advisees, not just the students.
test
Cant think of anything. My advisor has helped a lot.
I found my advisor to be extremely helpful and my advising experience was about as good as I could possibly imagine it being. He was extremely reasonable in providing availability around the time for registration, and always had good feedback for me at that juncture in my career. My only suggestion would be to perhaps have more faculty, so that advising is more dispersed.
I really enjoyed having Dr. Sheppard as my advisor. When I was younger he was a bit intimidating, but as we got to know each other he was very encouraging and honest with me on the best way to achieve my goals.
I had a very positive advising experience. I had difficulties at times but it was no fault of my advisor.
He was hard to get a hold of most days.
I don't even know who my advisor is.
My advising experience was minimal but I did not seek out more than what was required
I think more emphasis on discussing career plans post-grad and what to take in order to be more employable would be good.
no comment
I didn't really utilize advising much, but I think potentially more of a focus on finding a personal fit, both in terms of designing a schedule as well as finding an advisor in an area of interest would be good.
It's horrible and has no communication, puts everything on the student, largely unnecessary
Good! limited availability sometimes was annoying. There were times were no advisor was available for longer periods of time. Requiring a PIN for registration is ridiculous.
I always just went to Sharlyn for everything. She was phenomenal. Always felt like she cared about the students, and she clearly works very very hard to ensure that the students have what they need.
Since I was not assigned a class to advise for I helped with whatever was needed in the lab. If I could have accessed the assignments for the classes ahead of time and had a chance to read them before student's asked me for help it would help me save time when I'm assisted them.
My advising was ok. I think that my advising experience was greatly diminished by covid shutting down campus for a large portion of my time at MSU and hope that the futures students have a stronger connection with their advisor than me.
I had a freshman advisor for 2 1/2 years as it was never changed and no one notified me that this was the case. Other than that it was excellent.
Encourage advisors to talk about real world jobs and applications of the class topics. I feel like I don't really understand how the topics I learned align with industry positions that use those topics. It would be nice to told when registering for classes what kind of jobs they will prepare me for.
My advising experience was bad. Early in my studies I received very little help when I was struggling, even being told that school is not for me. I had to search out someone who wasn't even my advisor (shout out to Sharlyn!) for

any structure. More recently my advisor told me I should be contacting my advisor with questions about my senior year and was fully approved to graduate before being informed the day before the semester started I actually needed to take an additional class.

My advisor was good, it would have been nice to have a little more direction. My schooling was essentially extended two more semester's than it should have been. Maybe a little more willingness to look into things and make sure that this wouldnt have effected me so much trying to graduate when i wanted to.

According to my advising experience, I got a lot of help from my advisor. At this point, I believe our advising is good enough. It could be improved if we can choose our advisor instead of assign an advisor to student.

I wasn’t impressed by my advising experience. I would have to go to Sharlyn to get any valuable and detailed information about classes, even though she was not my assigned advisor.

I have received incorrect information from every advisor I've ever had (not just the CS department). In certain cases if I had not caught the error I would have had to push back graduation. Oftentimes it is difficult to contact and secure a meeting with advisors. MSU needs dedicated advisors (as in staff that are not also professors/researchers) that fully understand the degree requirements and timelines. Sharlyn was a great advisor, we need more people like her.

Advising with John Paxton has been easy and great.

I am a non traditional student, so I did a majority of my semester planning on my own and only spoke to an advisor to receive my PIN. Sharlyn, however, is the most helpful advisor I've ever had - although she was not my officially listed advisor.

I thought the advising was really good. I never had any issues with registration or scheduling a meeting. And, my advisors were helpful.

My advising experience was really interesting. I started with Sharlyn as my advisor and after one semester had Sean become my advisor. The first semester I had Sean, I was not able to get in contact with him at all to register, so I had to refer back to Sharlyn. Though the following semesters, I was able to be in contact with Sean and that was a really good experience. He had good insight on classes and was quick to responding. Overall a good experience with just one hiccup.

Loved having Sharlyn the first year, she was a blessing after having my first advisor in the University Studies system. When I was assigned to my next advisor, it was a bit hard, they had no idea about my field of study and could not really give advice on classes to take.

My advising experience went very well. Whenever I needed a question asked my advisor responded quickly and clearly. He was always there to help.

I feel that my advising went above and beyond the responsibilities it entailed. I have no complaints or changes I would make.

My original advisor was not very helpful to me, and I do not know why. It could have been that advising was something he had to do to fulfill his role as a professor and was not interested in doing much on that front. I suggest looking more at the assigned advisors and making sure they are invested in helping the students.

My advising experience was great. DeFrance was my advisor for most of my time at MSU and he was great. I never had an issue with him, and if he couldn't answer a question, Sharlyn could always help me out.

Overall my advising was pretty good but I do wish my advisor was more clear about what CS elective count as upper-division elective credits and what don't.

I have no complaints about advising.

I switched from physics after my first semester. I had a hard time figuring out which classes to take, I didn't realize there was a flowchart until my senior year.

My advising was horrible when I had Veronika Neeley. I had no guidance really and had lots of things cancelled resulting in me not getting into classes. Once my advisor was changed everything was fine.

I really enjoyed my advising experience. My advisor was Sean Yaw and he was always very helpful when I had any questions.

My advisors messed up and let me enroll for my capstone my junior year, which I didn't know until finals week. There is a severe lack of paying attention even when course lists are subitted for review. Advising focuses on

completing the degree, not fulfilling the goals of the individual in terms of what they want to do and learn. I'm sure some students aren't meant to be CS students but are not aware of other options. I hate Computer Science but by time I got an internship and realized it was not what I wanted I was already too deep into my degree.

Overall, I had a very good advising experience. Only issue is that my advisor changed 4-5 times throughout college. It would have been nice to have a consistent advisor to get to know and help guide my post-college plans (on top of University advising).

It was fine, I don't feel like I was able to get advising that would get me where I wanted to be, instead it felt like I would have a basic idea of what to take and that would be approved without a conversation about what classes would fit or help with my technical interests. I think having suggestions of classes for students interested in certain topics would help as well as requiring courses that other colleges require so that we get the same technical background as other students.

The advising I received was serviceable.

My advising experience has been great! However, I do wish it was easier to develop a professional relationship with my advisor.

My advisor was great. He and I had a mutual respect for each other's time. I am a busy student; he has research to do. If I understood it and had a plan, he respected that and provided minor advice. If I needed advice, we emailed, or video chatted to discuss it. My previous major didn't accommodate quite so well. I have no advice for improvement. My advisor was fantastic.

I mean no disrespect to my advisor but they did not seem very interested in the process. They struggled with the process itself which suggested to me that they haven't advised very many students. It mostly felt like just getting a rubber stamp so I could sign up rather than getting much advice.

My advising experience was very good. It could be hard to find what was being offered when in the case of electives, and maybe there could be an easier way to access a list of courses somewhere.

I had a good experience with advising. I usually would build my own schedules and audit them and then have an advisor approve them. I did not spend much time working with any advisor on building my schedule.

Advising was good

Advising was phenomenal. No improvements needed, I always got my questions answered promptly.

I thought it was good.

Advising was fine. I didn't really have much 1 on 1 with my advisor, so i don't have much comment here.

John Sheppard refused to advise me so that was a negative, however I then got to choose an advisor so that was nice.

I had an amazing advising experience. I typically went to Sharlynn for advising and she always helped me understand what classes I needed to take and when. My actual advisor wasn't super helpful when I visited them. They did not check to see if the classes I wanted to take were being offered and didn't have much advise for me on what classes to take and when to take their prereqs. I was approved to take multiple classes that weren't being offered the next semester, so I think more adequate training for new advisors would be beneficial.

I have had two main advisors during my time as a CS student and it never felt like they were helping me stay on track to graduate. I took a non-traditional degree route (working while in school, took ~7 cumulative years to graduate) and I had to plan my semesters almost entirely by myself. Having more support from advisors in planning non-traditional degree paths and semesters would be really helpful.

I had a decent advising experience and can't think of any way it can be improved.

Sharlyn is the best. I didn't run into any issues.

Advisors are great!

Advising is hit or miss. Some faculty does a very good job with it, while others just have no clue what classes are offered. Mine was more of the latter in my last two years of college. One improvement might just be to keep advisors to people who either have training in advising or on a volunteer basis

I had an advisor recommend I not take a course because it was 400-level and I was only a junior, and like a fool I listened to them and didn't take it. I wish I had taken it, and I wish my advisor at the time hadn't steered me away from it. Also, I wish my advisor had recommended taking 338 sooner because I took it way after I took 246 and it was really hard to do that.

The advising was overall good but when dealing with teacher advisors wasn't always the best because they would try to convince you that the classes that were taught by them should be taken by everyone. A way that I think that this could be improved is if there was more general advisors that mainly did advising because they are a lot more enthusiastic about what the students wants and is more helpful when making a schedule that the student will like.

I think letting students pick what advisor they want based on their interests would be helpful. not all advisors give the same insight to what students think about classes.

My advisor (DeFrance) was constantly slow to respond and when he did respond it was just him forwarding the question to someone else. It felt as though he was not really advising but just doing the bare minimum to stop wasting his time with me. Get better advisors like Hunter Lloyd.

It was ok but only when I had Hunter Lloyd. He was the only advisor I had that actually got anything done, all the others didn't do as great.

Sharlyn was the best person to help me in every sitiuation. Ensure other advisors have the same vigor and knowledge

My first 2 years of advising were great with Sharlyn. She was amazing as an advisor and went out of her way to make sure that I and a lot of other CS students got into the right classes. After my sophomore year my advisor was changed to Dr. Zhu and I have a less than desirable experience with him. He always seemed uninterested in helping me and discussing my college career. Each time I visited him in his office it seemed as if he was trying to get the meeting done as soon as possible. At one point he even told me not to come to his office again regarding class registration, but to rather email him for the registration code. For my final semester I decided to talk with Dr. Paxton rather than Dr. Zhu because I wanted an advisor that was willing to help me figure things out. He actually ended up noticing 2 classes that I had taken that were not being counted in my degree works that Dr. Zhu had missed while being my advisor. I think advising could be improved by having advisors that actively want to help students. I don't want to sound too harsh, but Dr. Zhu did not provide me with that at all. Being an advisor is a job where you are always willing to help and go the extra mile to help a student succeed.

It was a little frustrating, I was uninformed that my advisor was no longer at MSU and I had to find my own but Sharlyn helped the process a ton! Sharlyn is what made my experience at MSU so successful, she was also able to help me build schedules and understand how to add on other majors and what opportunities to pursue. She was also able to point me to the correct people if she was unsure on how to help me. Easily the best advisor I have ever had at MSU!

I had a great advising experience with Mary Ann. No improvements.

I think you get assigned an advisor in your second year at MSU. I think students should be able to potentially meet with any potential advisors prior to getting assigned. Or, at the very least, the professors that have openings send out a little blurb about who they are, what they do/did, etc. etc. Just sort of introduce them to the students, this would help students narrow down who best matches what they would like to do when they go into the job market. For example, if you really want to go into UX/UI, you shouldn't be just blindly given to the professor who teaches AI.

I went through four advisors since professors kept leaving (no problem for me as I had a 4 year plan already created). My first two told me to refer back to Sharlyn for all the questions I had. I would suggest giving all the advisors a FAQ sheet or a run-down of how to advise. Sharlyn is awesome though and deserves all the support - I don't think most students would be ok without her.

It was fantastic, sharlyn was amazing. Nobody says anything about your actual advisor.

Sharlyn is the best advisor I have ever met. She is phenomenal and goes above and beyond her duties in every possible way. However, I feel like she is so good at her job that everyone goes to her regardless of if she is their advisor or not. Other advisors need to do their part and be available for advising.

Advising was good - I just went to Shar although I am not sure she was my advisor.

My advising experience was acceptable. Binhai was very understanding and helpful in helping me register for courses.

My advising experience was awesome! I started with Sharlyn and then my faculty advisor ended up leaving the program so I stayed with Sharlyn for pretty much the whole time. She was so encouraging and helpful and gave me a lot of career and internship advice in addition to just advice about classes. When I had a faculty advisor I felt like she didn't know enough about the specifics of classes or the best order to take classes in, etc.

My advising experience has been minimal. I have only spoken to my advisor when seeking the PIN required to register for classes. I have never felt the need to speak to my advisor, and he has never reached out to me. Also, my advisor, Binhai Zhu, has been my advisor for all 4 years that I've attended MSU, and I understand that I should have had Sharlyn Izurieta-Gundersen as an advisor at first; I believe this to have worsened my advising experienec.

Average. Did a lot of self-advising and learning what was needed to graduate so did not rely on advising as much.

With three majors, advising has been a nightmare. I would like to see advisors offer a smoother experience for multi-major students.

Already good for me

I personally think I had gotten lucky with my advisors. The advisors I have had really cared about me and showed me that they cared about me. However, I know that all of my peers didn't have the same fate as myself. If Sharlyn could be put in a position where she could train advisors then I believe the advising system would improve a lot.

I do not think there is anything we would change for advising. I did not find any major errors.

My advisor was Sean Yaw and honestly he did a pretty good job of advising. I didn't really go to him often asking advisor related questions but he was able to answer most questions when I would email him about them. I think advising might be more improved by giving the advisors full permission from the start to access and edit a student's Degreeworks info if it is needed, as that was the only issue we ran into trying to move my classes around to fulfill the Degreeworks requirements

Advising went well for me. As far as I can see, any issues I had with advising were my own doing, or were easily resolved by contacting my advisor.

None

I had no real problems with advising. My advisor got back to me in an orderly and quick paced time. I was able to ask questions with no issues.

My advisor, after Shannon, didn't really help me. I would just show up with a plan I thought would work and would fill my requirements. They did not help me or direct me to pick different courses. There were times I saw that the courses wouldn't work together due to scheduling or even a course not even being offered that semester.

I had great experience with my advisor and she was very helpful and answered every question I had. I did not have any issues that stuck out to me that needed improvement.

i liked that i could go to whatever advisor i wanted to and not feel pressured into going to the assigned advisor

Paxton and Sharlyn were a huge help multiple times to me, it was also nice to be able to talk to teachers from my various classes and get their input. No improvement needed.

Sharlyn was my advisor the whole time. She is amazing obviously

My advisor was Brendan Mumey, and I thought he was very helpful and available anytime I asked for help, and was always quick to respond to emails.

I think it was fine. Sometimes it was difficult to get a response from my advisor so I would have to contact another. They did a good job of signing you up for classes but that was about the extent of it.

First year advisor was fantastic, after that the advisors seemed less knowledgeable and helpful. Maybe help train professors to be better advisors.

Finding the sources was somewhat hard, so I had to contact my advisor somewhat frequently for help. I used Sharlyn the entire time as my advisor, and am very satisfied about how helpful she was. DegreeWorks is outdated, and needs to be updated so advisors don't have to manually override so many classes.

Advising was great. Sharlyn is the best!

Advising was great, Shar is awesome!

I'm a not traditional student so I didn't really use advising. I knew the courses I need to take and just signed up for them when I could.

All my advising experiences have been wonderful. I don't have any comments for improvement at this time.

Sharlyn was an amazing advisor! She had the answers to all of my questions and came in clutch when it came time to certify me for graduation. My advisor, once I got out of Sharlyn's bubble, was Prof. Millman and he was great too. I maybe could have had a better pairing for economics, and we never talked too much about it, but he was accommodating and let me steer my education here with some cautions against overloading. :)

I hated it. I had advisors in CS, music, and (briefly) physics. None of them knew anything about the other programs. It was a chore to have to run around to get things signed off on and to be told that my plan was fine after getting my advisors caught up on how the other program works. Really made me feel like a kid. DegreeWorks is a great program, can't I just be in charge of my own life? Still, somehow after being babied every step of the way, it completely slipped through the cracks that my plan for years to drop my music minor if I couldn't finish it in time would not be possible until I went to do it. No, you can't be interdisciplinary then, even though DegreeWorks says you can as long as you get at least 12 credits in your minor! You've gotta switch to professional, take another full semester of math and science electives because music no longer counts towards anything, and you can't pick your capstone project, just take compilers like everyone else. Now I have a minor, which I'm grateful for, but was it worth it? I don't know. I've been here forever. I'm poor, and I'm over it.

Advising was good, just wish we used something other than DegreeWorks...

I have no complaints for advising, Shar and Sean are both amazing.

It was horrible. There were several times I got my degree works plan approved, only to realize that 2 or more of my classes werent being offered that semester. Shar is fantastic, so usually i just skipped my normal advisor and went to Shar.

My advising experience for the most part was non-existent. My advisor was absolutely unreachable and if it weren't for Sharlyn I'm not sure I'd be graduating. Either insure advisors like Professor Fasy don't disappear on their students, or simply Clone Sharlyn a couple dozen times.

Sharlyn is a god-send. I can't thank her enough for all the advising that she provided me with throughout my undergrad. She helped me to discover new opportunities and was always available for support and writing recommendations. My actual advisor (no longer faculty at MSU) was less than stellar. I did not feel like she wanted to help me at all and there seemed to be a bit of a language barrier. She provided me with incorrect information on occasion as well.

Q14 - Are CS facilities, labs in particular, adequate to support the classes you take? What improvements would you suggest?

Are CS facilities, labs in particular, adequate to support the classes you take? What improvements would you suggest?
I loved the labs its the only thing that made me feel like my education was worth it.
I never got to use the lab facilities within Barnard sadly. It would be cool if there was a CS center like mathematics' and writing centers.
test
No improvements. All you really need is accessible computers with the proper software.
Yes, in general. I think the most important thing would just be to have more faculty to accommodate more specialized coursework (eg. quantum computing)
I honestly never used the labs or really anything of the CS facilities, so I'm a bad person to ask
I think that they are. I will say I have never used a computer lab on campus but my experience with the robotics lab has been good. It is packed in there this semester but there is always somewhere to go to work on the stuff.
Yeah
The lab in first floor Roberts smells like elk musk. Students need to shower once a month minimum.
Too many CS classes in reid hall which is not very technologically advanced
I never needed to use the CS labs.
I never really needed them.
No, I didn't utilize any cs spaces in my courses apart from roberts 111 for the labs in the 127-232 sequence. I think that room could be set up much better, and there's no reason to have all the desktops in the rooms/labs, as I honestly have never seen them used in my time at MSU. (everyone has a laptop, and having the computers just takes up a lot of space)
No, you really need more powerful local machines that dont run on VMs, bare metal performance is essential, and more student control
Yes. Perfect. Department seems flexible to facilitate instructors when need be. This is appreciated by the whole community.
Yes, more or less. I don't have any suggestions.
I think they are helpful. Sometimes it's hard to tell who is a tutor.
The CS success center is the greatest resource I have found at MSU. While unfortunately I was not able to use if for a year due to covid it is by far one of my favorite things that exist on campus. Furthermore, I would suggest that this continues to receive the funding and attention that is required to keep growing. This center alone has shaped my college experience.
I made very little use of most of the labs as they were just banks of computers. The only ones I used were the maker space briefly and the robotics lab
I found the facilities to be adequate, just hard to find.
The facilities were great.

Yes.

Our lab section helps us a lot in programming. I believe it is good enough.

Yes, I don’t think any improvements are necessary.

Yes, we don't need much in CS. The lab was great.

The library and lab computers need to have stuff like PyCharm available. The CS Help Center can't be the only place to be able to use actually good IDEs.

Yes, they are adequate. No comment on improvements.

Yes, they have everything i needed

I definitely think that even the current CS facilities/labs are more than adequate enough to support the classes I took. With the new building that's coming in, I can only imagine that this will improve which is amazing.

yes Ioved the CS lab and how if I had questions I was able to ask. Maybe help people to understand that in higher classes the TA's may not be able to help with set up of environments that they have not used because they have not taken the class.

Labs and facilities are fine. Most people have computers/laptops to code on.

The facilities are already excellent, and I feel that I have access to more resources than I could even need.

Most of the computers on campus are old and slow likely because they are virtual machines run off of the school servers and I have not wanted to use them since i first came here. The software that they can use is fine and it works but i suggest looking into how you can improve the computers and how they can be made to not be virtual machines.

I never used the CS labs to try and work on a project until my last semester. My capstone project required Mac computers and the only public Mac lab in Reid was extremely outdated. I couldn't believe that the lab didn't have xCode which is the primary Swift IDE.

The labs are pretty good. I really appreciated the CS help center. The only improvement that would be nice is if every CS student got a certain number of cloud computing credits (i.e. GCP, AWS, Azure).

Yes, I always used my own laptop so can't really suggest what to improve.

They are good. The CS learning center/Success center is kinda weird. Once you get past like freshman year classes you really don't get much help from that area unless your TA is weird. In like mathematics and stuff the help center has a ton of people and you can get support all the way through senior level classes. I feel that that area is very understaffed and I tried using it for help several times but the people can't really help unless you are in data structures or a intro coding class.

Yes they are. Everyone has a laptop and can pretty much do this degree from home if needed so the classes were fine.

I thought that labs were usually pretty good, I can't think of any that stood out as poor.

TAs are severly under educated to help students. In my years I have had two good TAs who were able to actually help and communicate with me well enough that I learned instead of just getting some hint that isn't useful or thrown an answer

All of the Barnard and Norm As Classrooms/labs were beyond adequate. Reid hall is terrible, the temperature and airflow control is spotty, and every room is too small/stuffy and uncomfortable. I was very disappointed that nearly all of my classes were in Reid. It often demotivated me from going to class. Please move our department or vouch for improvements.

I used the lab my freshman year and following that did not.

Unless required by the class, I did all of my lab work on my own desktop.

Yes, the CS facilities are adequate. However, I wish that there was more space in the lab (or just more labs), they were taken advantage of more by other students. Finally, I hardly use the equipment in the labs because they're not the most useful at all times. I think if they had intellij IDE's installed on them as well as some other cs geared software. Finally, for the equipment specifically, I think dual monitor setups would be a nice touch. And also if there was an easy way for people to connect their own computers directly to them that would awesome.

No. CS students should never be expected to develop code in rows seated forward. That is unrealistic. I firmly believe that all labs for CS should feature round tables, or the tables like in the Barnard area. We are collaborative individuals.

A laptop is essentially required for this major, which can be difficult for those of us who chose to invest in a tritonal pc. Having pcs on campus is nice and all but that doesn't change the fact that you may not be able to take adequate notes or follow along in class if required. The labs are nice though.

All the labs I used seemed fine.

Yes, however I only used the labs and lab periods in my earlier classes and have rarely used them in my upper division classes

N/A

For the most part they are, I would suggest having Macs that have Xcode capabilities on them.

Yes

I find there was no reason to have computers in the lab room, as every CS student has their own computer. They were just in the way. More TA's in the lab would be helpful as well. I spent lots of time in lab waiting to get help.

Yes

I think it would be beneficial to offer lab times for higher level classes, however I do understand that resourced are much more limited in this area. The help center and lab times are great for lower level classes, however I think it should be required that students visit the help center at least once for a class so that they are more comfortable to use it later on.

Yes. I have never really needed lab access aside from a few select times to use some equipment. 99% of my assignments etc. were completable using only my personal laptop. I don't have any lab suggestions aside from more comfortable chairs.

I didn't use that labs as I could preform everything on my own devices.

I didn't have any issues.

I have my own equipment, but from my experience, it was all good.

No. In all my time doing CS labs I've never used on campus computers. I don't think CS labs are really well suited for communal computers. I'd recommend changing lab facilities so they're equipped to use personal computers (things like external graphics cards would be nice)

I think the labs were adequate for the most part. For some of the larger intro classes, I think it was difficult for the amount of TAs to help the amount of students in their classes.

Most labs are small assignments that are using topics in class todo a small project and i feel that this an adequate way to support the class cause it puts the topics into practice. A way that I would improve this is having optional lab that students could do for extra practice.

I think updating some of the computer labs would be nice, at least the desks and work area. I think the labs should be more collaborative and open discussion.

No, get faster computers.

Not really, I usually avoided the labs in favor of my own computer, because the computers are not good on campus.

No improvements

All of the facilities are good enough for me. A few of the classrooms (Reid Hall) are not very fun to sit in for an hour, but other than that I have no complaints.

The labs are nice it would be nice if there was more spaces as the department grows

Facilities were great.

I think the CS facilities are adequate to support the classes that I take. I'm not sure what could be done to improve.

I think the labs are amazing. Having TAs and Peer Tutors is super helpful and I like the space we have. Maybe having a spot in the lab for club information or announcements could be nice to know what's going on in the CS department.

Absolutely not. The facilities are only useful to first-semester freshmen.

I work in the human interaction lab on campus and I find that working there has helped me immensely. It is certainly adequate to support the classes I take, however, the space is extremely small and it makes it difficult to conduct experiments properly (especially when they involve a lot of space e.g. VR/AR studies).

Did not use them often - the labs in Roberts are quite run down.

There has not been much advertising to CS help. Like the math center, I believe there should be a computer science help center. I believe there is one, but I never hear it advertised, or that it is manned.

Using a nicer computer lab than the one in roberts for labs. Would have been nice to have better tools for TA's to instruct with.

CS facilities are sufficient, in my opinion. The CS lab located on the second floor of Barnard is an excellent work environment. Of course, I wouldn't mind if the CS department were given a building.

Yes, I enjoy working in a lab and with TAs in introductory coures.

I believe facilities are adequate for students.

Already good for me

They are for the most part, it could improve if there were more computers though. Sometimes all of the computers are taken.

It became less useful as I progress to upper division courses. If we could have some graduate or professors in charge of the lab for the whole day, it would be much more helpful.

I would say that the CS labs were adequate for the lower division classes I took in 127,132, 232, etc. One improvement I would ask for, and which I started to see toward the end of my lab experiences, was having more than 1 lab TA to go around and help answer student questions in lab. In a classroom of 30+ students and a set amount of time, a single TA can't adequately get around to each student efficiently before time runs out.

Everything was adequate for me. I didn't have any issues or feet like I was lacking anything during my time here that I can recall.

None

The labs were fine. No real issue with having a computer to do the lab with.

I believe so. I always found the computer labs we use to be sufficient. Maybe add some oculus rifts or other hardware that we could develop software on.

I thought labs were very adequate to support the classes I took, all of the resources we needed were available to us and it was just up to us to take advantage of the resources.

yes, none

I did not use the computer labs throughout my time at MSU. As far as facilities go having AC in some of the rooms would have been a nice addition especially during several weeks it was very hot.

yes

I thought they were just fine. I always brought my own computer, as did most of the people that I studied and worked on stuff with. I think if there were an improvement, I would get rid of a lot of the computers oddly enough in the labs and have monitors there with easy hdmi hook ups. Almost everyone just works on their laptop anyway, and we just pushed aside the mouse and keyboard, but I would always be down to use extra screens along with my laptop.

I think the facilities are adequate. There are enough computer labs with space to work on projects individually or as a group.

Facilities and labs were perfectly adequate.

A decent number of the labs didn't have enough help available for the size of the class. It would have been a bit better to have more people to ask for questions.

Not sure, I didn't really use them.

I think, I didn't use many of the facilities.

I have my own computer so I didn't really use the labs much. However, there's always a need for more lab space.

I think it would be great if there were more spaces with tables, chairs, and outlets. The facilities themselves are great but most students bring their own laptops for development and coursework and there's not a lot of space to sit down with one's laptop to do work

I relied almost exclusively on my own laptop, and I believe the Roberts labs were adequate for early-level CS classes. I like the study lounge in Barnard, and those computers had everything I needed. It would be convenient if there were a printer up there, though!

Sure. It's quite a nice place, that area in EPS with all the computers and whiteboards. It would be nice to have more working space for people who have their own laptops and do not need to use school computers! The library is great if you're alone, but in a group, I found myself pushing keyboards and lab computer out of the way, as next to nobody was working off of them.

Not too many suggestions here, I always used my own personal laptop, but the spaces were great for working with classmates. Community chargers could be nice to have for when students forget theirs, which could be a checkout process from whatever TA is in the lab.

Yes, however, I would suggest we stop holding CS classes in rooms that don't have power outlets.

Yes- I would say more labs for upper division classes. For example, the labs that are in 127-232 should be continued through most of the coding heavy upper division classes- 366, 466, etc

I think as CS majors we're lucky. All we need is a computer and a keyboard. However I found the CS success center and lab classrooms to be constantly filthy. Only the trash cans are empties, its up to the students to wipe down their spaces.

Yes, I felt like the labs were quite adequate. To be honest, I did not use them often as I found working in a study space elsewhere on campus or from home, was better for me.

Q15 - Are you satisfied with the department web site? What suggestions do you have for improving it?

Are you satisfied with the department web site? What suggestions do you have for improving it?
make it easier to find staff for each department
One big table of contents would be awesome as a page.
test
I use it so little that I have no idea.
I am satisfied with the department web site, and I particularly appreciate that people's office hours are posted there. My only suggestion would be to consider adding sections corresponding to particular research groups, so interested people can see what active research is going on in the department (eg. have a link to the AI group, networks group, computational geometry group, etc.)
I think it’s very good honestly. I mostly use it for looking up office hours and it was pretty quick to find that.
I am very happy with the department website. I think it presents needed information in an easy to access manner which is all I can ask of it to do.
Yes and none
I didnt know there was one.
The site functions well enough. I probably visited it five times in four years
I have no complaints about the web site.
I mean its not the best from a design standpoint put none of the universities sites are.
It's not my favorite, but truthfully I'm not sure how I would actually improve it
it's fine
Satisfied.
Again, no suggestions.
I think the CS Department website is good, my only suggestion would be making the link to the help center more apparent on the first page.
It is overall great. I enjoy that each of the Professors and TA's office hours are easy to find which I didn't know was that complicated but I guess it is(Looking at the electrical engineering school here). I honestly can't think of much I would have liked to see that isn't already there.
It works, but it could look a little cleaner
I am satisfied with the department website.
Website was also good.
Yes,
The website looks good to me.

I am satisfied.

A flowchart that maps out the courses needed for graduation would be great, see the mechanical engineering flowchart.

Yes.

Yes, I'm satisfied with the department website. No comment on improvements.

Yes, I don't think it needs any major improvements.

I think the department web site is solid and gets the job done. No suggestions for improvement.

Yes it was good

The internship page should be more visible and broadcast to students.

The department web site is navigable and clean, I am already satisfied with it.

The web site is just fine, i have no suggestions for it.

The department website is perfect!

I am satisfied with the department's website. I especially appreciated the "Office Hours" section where I could easily find professor's office hours.

Yes

I am satisfied. For being a computer science department you would think we would have a great website that looks and works really well, especially since we have web development classes. But I guess the super plain website is cool.

Yes it was fine. I did not feel like I needed to use it in my 4 years.

Sometimes it was a bit of a struggle to find some of the documents that I was looking for, I didn't know what general student resource to look for it under

montana.edu is a poor website overall in my opinion. I can only access what I want with the search bar. also not knowing if something was up to date makes the site unreliable. An updated on section would be helpful

Department website was passable.

I rarely used the department webpage.

The department site is satisfactory and works well for what was intended.

I'm happy with the website.

The current students tab should be more obvious. From there, the content in it should look less like a directory and more like a webpage. Also, there are more pictures of with females in them, than females in the average class. I want women to be more included in this fantastic field. The pictures are definitely misleading though. Also, a student who graduated a few years ago is in 3 pictures on the website. Updating the pictures more regularly would be cool so we can see our peers who we know being recognized.

Yes the website is very useful, its much better than the main college site that's for sure.

I do not have any problems with it, but I rarely ever used the sites.

I am satisfied with the website. It provides a good amount of useful information

N/A

Department website looks great.

The office hours page is sometimes not up to date. Also it would be nice if there was a page for clubs and current activities. Also I am not sure why, but finding the required courses / curriculum is non-trivial for me.

I think the website is fine. Its simple and easy to find information.

MyInfo is a mess, the dept website is well designed but it is hard to find stuff

The navigation for the website is a bit confusing to me. I think a drop down with key pages under each item on the main page would be beneficial. Otherwise I find the department website quite useful.

It's fine, I suppose. I don't really use it much aside from looking up information, which I usually use Google for and may or may not end up on the MSU CS website. I don't have any suggestions.

I'm satisfied with the department web site and can easily find what I'm looking for.

I am satisfied with it. I don't have any suggestions.

Ya, it looks good.

Web site is fine. No suggestions.

Yeah I'm satisfied

I feel that the website could be improved by updating the styling of the site.

yes

It's okay, you guys have professors with UX and UI experience how is it even allowed to exist the way it is, not to mention the plethora of talented design students available. Please utilize your resources.

No, it needs to be easier to navigate and the information needs to be easier to find, such as contact info since that is usually what I would need to find when I went there.

No improvements

The department website is fine.

Part of the website images and style makes it seem outdated but all the information it up to date. I think incoming students would feel more encouraged to enter computer science if we had a more modern look to the website.

Yes. None.

I am satisfied with the department web site. The only improvement that I think could be done is to better highlight the student success center.

I honestly don't go on it at all.

I have never used it.

I dislike the department website. I find it very difficult to find what I am looking for -- I would suggest usability studies with actual students to determine the usability, utility, and user satisfaction.

Good not great.

Yes, within a few clicks I can find most stuff. There should be a database that lists the works of the professors.

Yes, none

I am satisfied with the website.

Yes.

I am satisfied with the website. No suggestions.

Already good for me

The website is alright. I have no complaints there.

I do not use the department web page a lot. Therefore, I do not have comments on it.

I honestly didn't have to access the department web site often so I would say that it was adequate for any needs I might have had. I do like that all faculty contact info for the department is easily accessible for getting in touch with the right professor for questions.

Yes. The only thing I can think of is maybe making the jobs section more apparent.

None

The website is fine. Having the information of the people in charge is the most important thing.

I am satisfied with the department web site. I do find most of the school website confusing so maybe changes in layout.

I am very satisfied with the department site. It has all of the information we need on there and I cannot think of anything that would improve the site.

yes, none

Yes, the only reason I used the department website was to find contact info for a teacher or to look at class requirements.

Website is good

It can be hard to find what advisor you have and how to contact them, I would make that information more readily available, as well as ways to sign up to meet with them. Past that, I didn't even really know that we had a department web site.

I think its fine

I am perfectly satisfied with the department website.

It is pretty decent. Not too many gripes about it.

I think so, I haven't explored the website too much but I have no complaints.

Yes, I haven't accessed it much but when I do use it it seems to have what I'm looking for.

I don't feel like there are any problems with the website.

Overall the department website is great. However, it would be great if there were a section of the site that displayed degree requirements and course descriptions (like how Degreeworks displays course information and descriptions). I say this because once a class has a checkmark or is in progress on Degreeworks, a student can't click on the class to see information or descriptions which makes looking for alternative classes much more difficult. So a section of the site that listed the courses, what section they fulfill (CS Electives, math and science electives, etc.) would be really helpful.

The school of computing website is great! There are so many resources (especially when I'm contrasting my experience to the Econ department). There's a typo in here ("Probablility"): http://catalog.montana.edu/undergraduate/engineering/computer-science/interdisciplinary-option/

I haven't really thought about it. Ah, it does its job, it has never stood out to me in a positive or negative way, and I don't know how I'd improve it.

Making it easier to find the CS clubs from the website; I don't know if they're even listed on there.

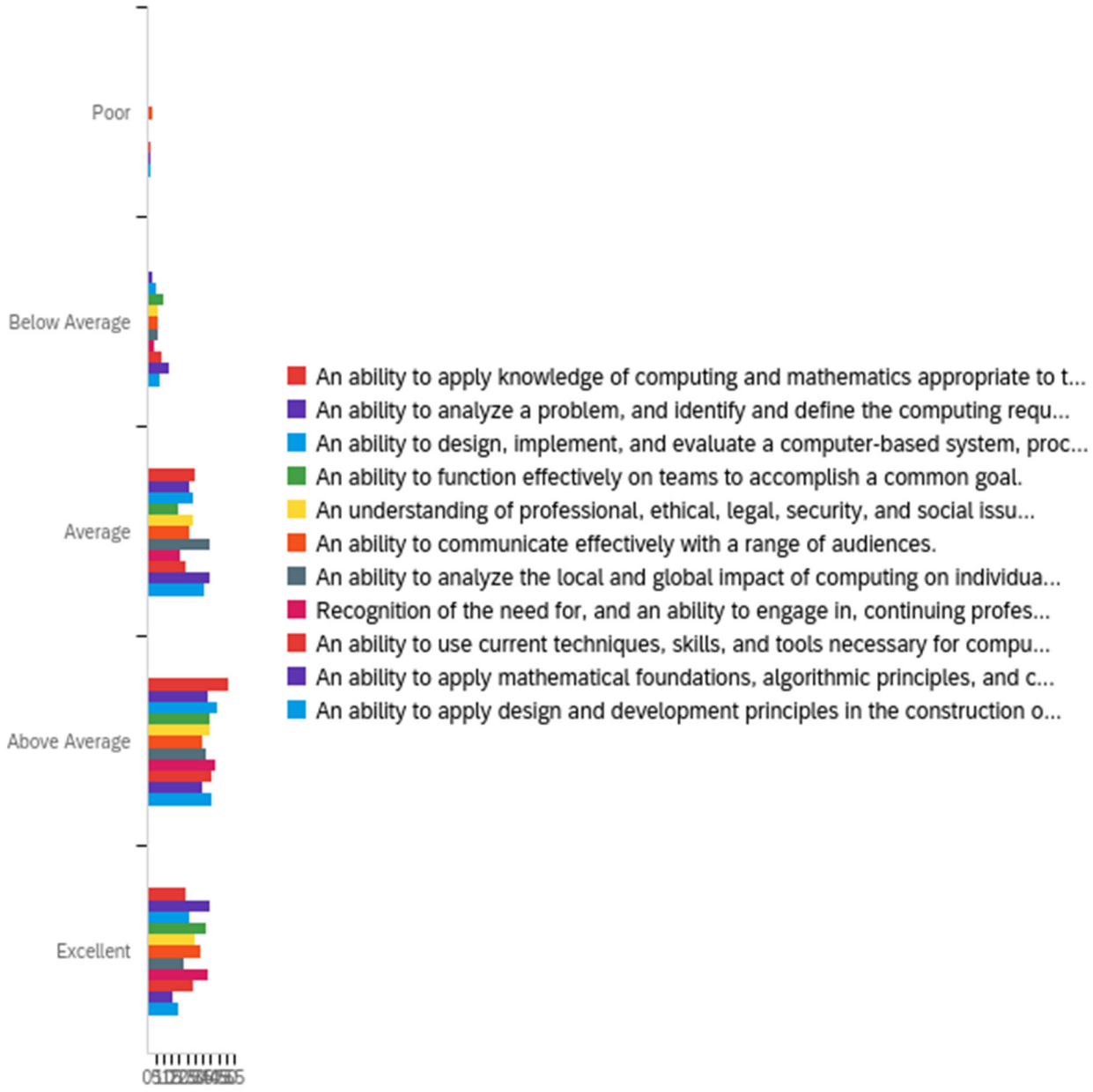
Yes I am satisfied, I did not find much reason to ever go look at it however.

Yes- the team does a fantastic job of keeping things up to date, and it is easy to find help, and find professors websites.

When visiting https://www.cs.montana.edu/ the only way to find degree opportunities is that tiny little blue hyperlink "the degrees we offer". I'm no Dr. Laura Stanley but I'm pretty sure that's just bad UX design. Considering that's MSU's product, I'd think MSU would like to make finding their money makers a little bit easier.

I don't believe I've ever visited the department website, so I cannot comment.

Q17 - Please indicate your level of preparedness in regard to the following CS program outcomes.



#	Field	Minimum	Maximum	Mean	Std Deviation	Variance	Count
1	An ability to apply knowledge of computing and mathematics appropriate to the discipline.	2.00	5.00	3.92	0.74	0.54	106
2	An ability to analyze a problem, and identify and define the computing requirements appropriate to the solution.	2.00	5.00	4.07	0.85	0.72	106
3	An ability to design, implement, and evaluate a computer-based system,	1.00	5.00	3.86	0.88	0.78	106

	process, or component, or program to meet desired needs.						
4	An ability to function effectively on teams to accomplish a common goal.	2.00	5.00	3.97	0.96	0.91	106
5	An understanding of professional, ethical, legal, security, and social issues and responsibilities.	2.00	5.00	3.88	0.90	0.81	106
6	An ability to communicate effectively with a range of audiences.	1.00	5.00	3.85	1.04	1.07	106
7	An ability to analyze the local and global impact of computing on individuals, organizations, and society.	2.00	5.00	3.73	0.86	0.75	106
8	Recognition of the need for, and an ability to engage in, continuing professional development.	2.00	5.00	4.08	0.84	0.70	106
9	An ability to use current techniques, skills, and tools necessary for computing practices.	1.00	5.00	3.82	0.99	0.99	105
10	An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the trade-offs involved in design choices.	1.00	5.00	3.48	0.96	0.92	105
11	An ability to apply design and development principles in the construction of software systems of varying complexity.	1.00	5.00	3.63	0.92	0.86	106

#	Question	Poor		Below Average		Average		Above Average		Excellent		Total
1	An ability to apply knowledge of computing and mathematics appropriate to the discipline.	0.00%	0	0.94%	1	28.30%	30	48.11%	51	22.64%	24	106
2	An ability to analyze a problem, and identify and define the computing requirements appropriate to the solution.	0.00%	0	2.83%	3	24.53%	26	35.85%	38	36.79%	39	106
3	An ability to design, implement, and evaluate a computer-based system, process, or component, or program to meet desired needs.	0.94%	1	4.72%	5	27.36%	29	41.51%	44	25.47%	27	106

4	An ability to function effectively on teams to accomplish a common goal.	0.00%	0	9.43%	10	18.87%	20	36.79%	39	34.91%	37	106
5	An understanding of professional, ethical, legal, security, and social issues and responsibilities.	0.00%	0	6.60%	7	27.36%	29	37.74%	40	28.30%	30	106
6	An ability to communicate effectively with a range of audiences.	2.83%	3	6.60%	7	25.47%	27	33.02%	35	32.08%	34	106
7	An ability to analyze the local and global impact of computing on individuals, organizations, and society.	0.00%	0	5.66%	6	37.74%	40	34.91%	37	21.70%	23	106
8	Recognition of the need for, and an ability to engage in, continuing professional development.	0.00%	0	3.77%	4	19.81%	21	40.57%	43	35.85%	38	106
9	An ability to use current techniques, skills, and tools necessary for computing practices.	1.90%	2	8.57%	9	22.86%	24	39.05%	41	27.62%	29	105
10	An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the trade-offs involved in design choices.	1.90%	2	12.38%	13	37.14%	39	33.33%	35	15.24%	16	105
11	An ability to apply design and development principles in the construction of software systems of varying complexity.	1.89%	2	7.55%	8	33.96%	36	38.68%	41	17.92%	19	106

Q19 - Is there anything else we should know?

Is there anything else we should know?
Nope
I didn't realize we could go back and retake the survey and I accidentally skipped the first page earlier, so I filled in the first page now.
thanks
Make the university pay for Hunter's robotics router.
Thank you for a wonderful 4 years!! I accidentally clicked the advance button before I'd written anything on the first page, (and I don't see a way to go back) so that's blank. But in general, I've been incredibly happy with the department, and enjoyed taking computer graphics, computer science theory, and computational topology the most. I think the weakest point of the department is just that some specialties are more limited, and my biggest suggestion for a course to add would be quantum computing. Otherwise, I don't have too much else for you. THANK YOU, this program has prepared me better than any other CS program could've I think in the entire country.
Every single CS student I've talked to has a horror story about a Hunter Lloyd class. I think he's a great and talented guy, but often his 100 and 200 classes are really unprofessional compared to the other professors. I had him in Social and Ethical issues and we watched the same Big Buck Bunny film twice to demonstrate what open source software is? That was 5 years ago and I still remember being dumbfounded I was paying for the course.
Nope
have fun teaching
no
Thank you!
I found the programming assignments in Carson's classes very beneficial to my skills in programming. They were the perfect amount of challenge and let me gain coding skills while also learn about the concepts of whatever the class was for.
Nothing comes to mind.
no
N/A
No.
No comment.
Please don't let Hunter teach entry-level CS classes. He is a decent enough professor when he is interested in the subject. Not so good when it is trivial to him.
Nothing in particular, thank you for an excellent undergraduate experience!
N/A
It has been such an empowering experience going through the MSU computer science program. Thank you so much!

No, thank you

The professors of this department are absolutely amazing, do what you can to keep them. Just hire one or two more to teach more recent topics. Also just redo the whole CS curriculum. It has already changed this year to take out that Egen which is great. If the course offerings get revised and expanded this school will have a great program.

I am so happy to be done with school I am 32...

I just want to thank you all for the last four years, they have been so much fun learning so much and meeting so many people! I'm really proud to be a part of this department!

Professors using discord for their courses is an excellent tool

Thank you for a wonderful Four Years! I am immensely thankful for this department.

Git should be taught earlier, Carson taught debugging for JetBrains studio in 366 and my mind was blown, and I felt like I struggled way harder than necessary not knowing how to use it. Submitting zips from a git repo provides the basis to prosecute students who plagiarize.

Nope, goodluck with all the other students you receive.

N/A

N/A

Not that i can think of.

Dr. Paxton is a wonderful department head

Sharlyn is the best and Hunter should always teach 305 and 246.

Go Cats!

Start the students learning Git!!!!!!!

Nope

NA

To sum up my experience, too many pointless courses and not enough on actually developing software solutions. Also, hire more professors like Carson Gross because the real-world perspective and enthusiasm should not be underestimated.

No

Should have said this before but EGEN 310 feels very disconnected from the major as a whole. Was not very applicable. Also would have loved to been able to take more math or cs courses rather than needing to take required science classes.

N/A

NA

Drop EGEN310 out of the curriculum. Thanks!

I believe Dr. Paxton should write less emails, OR there needs to be a general department email to recognize emails of high importance. Or a D2L page every CS major is in to be apart of for resources, announcement, etc.

I forgot to put this earlier in the suggestions section, but having lab times with the professional electives would be incredibly beneficial!!!!

No.

No

Some CS male students should be told to wear proper pants and have proper hygiene.

No

I know how hard you guys are trying to make the department the best it can possibly be, and for that I just wanted to thank you guys for putting in all the effort for our education. All of you guys deserve a pure gold nomination!

Nothing comes to mind. Thank you to everyone who helped me succeed!

None

N/A

I can't think of anything.

I think it would be good if more professors would allow students to retake quizzes or redo assignments after they had a chance to fix their mistakes. I also believe it would be great if professors would be able to accommodate students who suffer from burnout, especially students who have been in school for the entirety of the pandemic. It's been hard to keep up motivation and I know students who are doing their best, but just really need a break to rebuild their motivation. If possible, it would be great if professors were allowed to accept late work (within reason, of course) and allow students to learn from mistakes on assignments and quizzes and to fix them, so the focus is on improved learning of the course material and not on punishing students who don't understand the material in time

I loved my time here! I will carry forward so many useful data science skills that I can apply to economic research.

Um, thanks for everything. I think I've found my place here.

Very happy with my time in the MSU CS program! Thank you all!

Give Shar a raise.

thank you for providing a complete and a very fun college experience!

Thank for everything.

Thank you for everything. I had an amazing experience and I would choose MSU again in a heartbeat. Any criticism I left on this survey is only for you to take as advice to improve. I, overall, had an amazing experience here and can't thank the faculty enough for their support. MSU will always have a piece of my heart.