CSE6010, Fall 2026 Syllabus

- Instructor: Spencer Bryngelson
  - **Office hours:** Virtual, Friday, 5-530PM, Link: [see Canvas]
    - You can contact Spencer for additional office hours

Course credit (must read!)

GT Policy: Students cannot receive credit for both CSE 6010 and CX 4010. No credit for graduate students or undergraduates in Computer Science or Computational Media.

Course website

Canvas! And Piazza [see Canvas for link].

Contact

Email is the worst method of contacting the teaching team. Instead, use Piazza, which you can also reach through the course Canvas site.

Course description

This course is designed for students with a limited computer science background to prepare for upper-division and graduate coursework in computational science and engineering or computer science.

Course topics will focus on

- Data structures and algorithms
- Algorithm analysis
- Computer systems
  - Hardware, computer architecture, and operating systems including parallel computing
- Software design and development using C.

Some topics, such as selected algorithms and data structures, will be covered in relatively high detail, to the extent that you can code solutions in C or another appropriate programming language. Other topics, e.g., computer systems concepts, will be covered more at an introductory level in order to introduce you to key concepts. Our intention here is for you to

"know what you don't know" regarding these topics rather than to provide in-depth treatment. Because many topics are covered and programming is required, you should expect this course to require a significant amount of time, *especially if you have limited programming experience*.

Pre-requisites

You are expected to have a base level of programming experience (e.g., Python, Java, Matlab). In particular, this course assumes you already have knowledge of basic programming concepts (variables, loops, functions, arrays, I/O etc.) and have some programming experience (at least one course in programming). If you have limited experience in programming (e.g., a little exposure to Matlab), you are welcome to take the class, but you should expect to devote a significant amount of time and effort to this aspect of the course. Programming assignments will require software development. If you know one language, picking up another is relatively easy. The initial classes and programming assignments are designed to get you up to speed in programming using C; later assignments and workshops will integrate some C concepts that are more advanced.

Classes and recordings

You are *not* required to attend any classes, though historically students who come to class **do better**. So, I encourage you to come to in-person classes. All classes will be presented from my iPad to the projector, recorded, and disseminated on Canvas. Live Zoom links for each lecture will also be posted, though I cannot promise to monitor Zoom chat for questions during class (I do my best).

Material and scheduling

Course textbooks

Primary Texts

- Cormen, Thomas H., Leiserson, Charles E., Rivest, Ronald L., and Stein, Clifford. *Introduction to Algorithms*. 3rd ed. Cambridge, Mass.: MIT, 2009.
    - Standard widely used reference on data structures and algorithms.
- Patt, Yale N., and Sanjay J. Patel. *Introduction to Computing Systems: From Bits and Gates to C and Beyond*. McGraw-Hill, 2003. ISBN: 978-0072467505 (or international student edition).

    - Book on hardware, machine architecture, and low-level software; includes coverage of the C programming language.
    - *Note:* No ebook is available for the library to obtain, but a reserve copy has been requested (possibly a later edition, but similar in content).

Secondary Texts: Only a limited amount of material from these will be used.

- Hennessy, John L., and David A. Patterson. *Computer Architecture: A Quantitative Approach*. 5th ed., 2012. Morgan Kaufmann Series in Computer Architecture and Design.
    - Standard text on computer architecture covering more advanced architecture concepts; only a small portion will be used in this class.
- Silberschatz, Abraham, Peter B. Galvin, and Greg Gagne. *Operating System Concepts*. 8th ed. Hoboken, N.J.: J. Wiley & Sons, 2008.
    - Standard operating system text, aka the "dinosaur book."
- Sterling, Thomas, Matthew Anderson, Maciej Brodowicz, and C. Gordon Bell. *High Performance Computing: Modern Systems and Practices*. 2018.
    - Recent text/reference for high-performance computing.

Schedule

The course material will (tentatively) be presented in this order. We may not get to all material!

Lecture Topic	Reference Reading
Lists. Stacks, queues, priority queues	P 1; C 10.1–10.3
Introduction to C	P 11–14
Priority queues, binary trees, heaps	C 6.5; 10.4; 6.1–6.3
Pointers and arrays, structures	P 16; 19
Algorithm analysis	C 3.1–3.2
Data representation	P 2.1–2.7
Graph representation	C 22.1
BFS, DFS, topological sorting; recursion	C 22.2–22.4; P 17

Lecture Topic	Reference Reading
Shortest paths	C 24 intro; 24.1; 24.3
Combinational logic; Sequential logic	P 3.1–3.7
More shortest paths, C preprocessor	C 24.2; 25 intro; 25.1
Sorting 1; file I/O	C 6.4; 2.1; 2.2; P 18
Von Neumann model; ISA; data paths	P 4; P 5
Machine organization; stack; I/O; interrupts	P 5 + Appendix C.1–C.4; P 9.2; 12.5; 14.3; 8.1; 8.5; 10.2; 9.1.1–9.1.2
More sorting	C 2.3; 7; 8.1–8.4
Concurrency, synchronization; OpenMP	St 2.3, 2.7.1, 2.8, 9.1–9.6; 7, Si 3.1, 3.3, 3.4, 4.1, 6.1, 6.2, 7.1, 7.3
Search, binary search trees	C 12.1–12.3
Hashing	C 11.1–11.4
Caches, virtual memory, etc.	H Appendix B.1, B.3; 5.2, 5.3; Appendix C.1–C.2
Dynamic programming	C 15.1, 15.3, 15.4

Project

The project is a team-based activity on a topic of your choosing. The goal is for you to try your own hand at identifying a scientific problem of your interesting and implementing it in a sophisticated program that entails the materials you learned in class.

Programs that do not solve a known problem are not acceptable. You must come up with your own novel problem and solve it using the appropriate algorithmic techniques and programming strategy. Choose a scientific problem that is of interest to you and your group!

You will pick your own teams, which should have 3-5 students each. The final deliverable will consist of a project report, a short (5 minute) project video, and the code you've written, including the GitHub repository. You must provide the GitHub repository to your work during checkpoint 1!

Your project will be graded based on appropriate use of algorithms and data structures, demonstration of knowledge from the course material (command of how computer programs work), translation of the former into a fully documented (via your own choice of documentation deployment, e.g., Readthedocs) codebase, and use of multi-processor (serial programs are not acceptable) systems. All projects must include continuous integration (CI) and an open GitHub repository that shows their program outputs the results one expects, including rationalization about why the result one expects is, well, indeed what one expects!

Quizzes

We drop your *lowest* quiz score. Quizzes are online (on Canvas), multiple-choice, and timed (60 minutes). You have three days to start and complete each quiz. Quizzes will focus on all components on the course. One of the early quizzes will focus on syllabus components to ensure you understand what you are getting yourself into!

Grading

Breakdown

- 50% Quizzes (throughout the course)
- 50% Final project (semester-long)

Letters

Your final grade will be assigned as a letter grade according to the following scale.

- A: 90-100%
- B: 80-89%
- C: 70-79%
- D: 60-69%
- F: 0-59%

Extra credit

There are no extra credit opportunities in this course, with the exception of a 1% grade boost if better than 50% of the class fills the CIOS evaluation. I will remind you of this as the CIOS deadline approaches.

Course expectations and guidelines

Late assignments

There are ***no extensions*** for late quizzes! So, be sure to take them in the time slots given! The purpose of dropping the lowest quiz grade is to allow such things to happen without undue penalty: if you miss a quiz, just make sure you are taking all the rest!

Student use of mobile devices in the classroom

No policy here, but **do not** disrupt or distract other students.

Academic integrity

Georgia Tech aims to cultivate a community based on trust, academic integrity, and honor. Students are expected to act according to the highest ethical standards. For information on Georgia Tech's Academic Honor Code, please visit http://www.catalog.gatech.edu/policies/honor-code/ or http://www.catalog.gatech.edu/rules/18/. Any student suspected of cheating or plagiarizing on an exam,or assignment will be reported to the Office of Student Integrity, who will investigate the incident and identify the appropriate penalty for violations.

Collaboration policy and rules

- Quizzes are intended to be individual assignments. They will generally be open-book and open-internet, so you don't need to memorize everything. However, you should not directly collaborate with others. Such examples include asking for help or posting questions on or copying solutions from sites like StackOverflow.
- Projects will involve team efforts and are expected to entail close collaboration within each team. However, each student within the team must have a clearly identifiable contribution, and each student is required to develop software and contribute holistically. Individuals or teams may discuss the project with other teams; however, no software may be disseminated between teams except when explicit permission is granted by the instructor.
- In addition, you will need to use the Web or other sources for information as you work on your projects. You must cite these sources in your reports. For your code, you can use libraries but are expected to develop a substantial amount of code in constructing your simulator. The course staff can provide some guidance here on what will or won't be allowed. The spirit of this requirement is that you build a simulator and methodology yourself, rather than using existing simulation software simply to do some experiments.

AI Policy

This course recognizes that generative AI tools (e.g., ChatGPT, GitHub Copilot, etc.) can be powerful aids, but also introduce new academic integrity challenges. To ensure fair and transparent use, the following rules apply:

1. **Permitted Uses**

- **Brainstorming and Conceptual Exploration** You may use AI tools to explore high-level ideas, clarify programming concepts, or generate pseudocode outlines.
- **Syntax & Documentation Suggestions** AI may assist with boilerplate code or inline documentation (e.g., docstrings), provided you review and understand all generated content.

2. **Prohibited Uses**

- **Unaided Code Generation** You may *not* submit AI-generated code (in whole or in part) as your own work for quizzes or the project without substantial manual modification and comprehension.
- **Direct Copying of Solutions** Relying on AI to solve entire assignments, writing project modules, or answering quiz questions is prohibited.
- **Excessive Dependence** If an AI tool produces more than a few lines of code or complex algorithm logic, you must (a) explicitly acknowledge its use and (b) rewrite the solution in your own style, ensuring you fully grasp each step. All code you write must be commented, describing how the algorithm works and connecting it with the class to demonstrate this!

3. **Citation & Disclosure**

- **Written Assignments & Reports** If you consult AI for text, you must cite the tool in your bibliography or footnotes (e.g., "Used ChatGPT to draft algorithm description, corrected and expanded by the author").

- **Code Comments** For any substantial snippet inspired by AI, include a comment block (as below) and describe in that code blocks what you did and what the AI generated.

```
/*
 * Generated with ChatGPT (OpenAI), June 2025.
 * Reviewed and modified by [Your Name].
 */
```

- **Piazza Posts** If you paste AI-generated text/code into Piazza, clearly label it as such.

4. **Collaboration vs. AI**

- Using AI does *not* substitute for discussion with your teammates or the teaching staff. If you're stuck, ask on Piazza under the appropriate thread—avoid posting AI dumps.

5. **Integrity Violations**

- Submissions that appear to be predominantly AI-generated, without evidence of student understanding or original work, will be treated as honor-code violations.
- Instructors reserve the right to require oral "viva" or code walk-throughs if AI misuse is suspected.

6. **Best Practices**

- Treat AI outputs as you would a web-search result: verify correctness, test edge cases, and adapt style.
- When in doubt, ask on Piazza (privately, if needed) whether a specific use is acceptable.

Accommodations for students with disabilities

If you are a student with learning needs that require special accommodation, contact the Office of Disability Services at (404) 894-2563 or http://disabilityservices.gatech.edu/, as soon as possible, to make an appointment to discuss your special needs and to obtain an accommodations letter. Please also e-mail me as soon as possible in order to set up a time to discuss your learning needs.

Acknowledgements

I acknowledge the help of Professor Elizabeth Cherry, who has re-imagined and recreated this course several times over the years. We are using some Spencer-altered slides from Prof. Cherry where appropriate.