

CSC 2720 - Data Structures

Course Staff

Instructor: Jun Yi

Email: jyi39@gsu.edu

Webex: <https://gsumeetings.webex.com/meet/jyi39>

Teaching Assistant: Nikhil Koditala nkoditala1@student.gsu.edu

Teaching Assistant: Sarwan Ali sali85@student.gsu.edu

Teaching Assistant: Bikram Sahoo bsahoo1@student.gsu.edu

Teaching Assistant: Akshay Juyal ajuyal1@student.gsu.edu

Make sure you add CSC2720 in the email subject and CC all Teaching Assistants so that you're able to receive a response within a day or two. Also, do NOT email over on iCollege.

Course Schedule

Lectures

CRN 84994, CRN 84995, CRN 84993

M,W 11:00 - 12:15 PM (Classroom South 101)

CRN 84998, CRN 84996, CRN 84997

M,W 03:30 - 04:15 PM (Classroom South 101)

Office Hours

Jun's Office Hours

F 2 PM - 4 PM. Webex.

TA Office Hours

Bikram Sahoo M: 11:00~1:00 PM

<https://gsumeetings.webex.com/j.php?MTID=m3a2dc7824bff4631adb90598ff2357c1>

Sarwan Ali F: 10:00~12:00PM

<https://gsumeetings.webex.com/meet/sali85>

Nikhil Koditala T: 9:00~11:00 AM

<https://gsumeetings.webex.com/meet/nkoditala1>

Akshay Juyal M: 12:00~2:00 PM

<https://gsumeetings.webex.com/meet/ajuyal1>

Lab Sessions

CRN-84994, F 9.00 – 9.50 am, Langdale Hall 405

CRN-84995, F 10.00 – 10.50 am, Langdale Hall 417

CRN-84993, F 11.00 – 11.50 am, Langdale Hall 417

CRN-84998, F 12.00 – 12.50 pm, Langdale Hall 405

CRN-84996, F 1.00 – 1.50 pm, Langdale Hall 415

CRN-84997, F 2.00 – 2.50 pm, Langdale Hall

About this Course

This course will focus on the study of **data structures** and some corresponding **algorithms**. The study of data structures is essential to being a strong software engineer or computer science researcher, and is highly important in a lot of other fields as computers and technology become more important in society. In this course, we will expand our previous knowledge of coding and programming to include the ability to think about computational problems (algorithms) and how to organize data in optimal ways (data structures).

Data structures are crucial to the study of algorithms and in modern software engineering. The vast quantity of data available today demands efficient structures for storing, reading, updating, and manipulating this data. This course will teach you the basic tools to handle large amounts of data efficiently, for use in algorithms or otherwise.

Of course, students tell me that more than anything else the reason they want to take the course, in a more practical sense is that nearly every modern software engineering interview involves questions based on data structures and algorithms. If you end up becoming a software engineer, you can expect all of the datastructures studied in this class to appear in at least one of your future interviews!

But, interview prep in and by itself is not the purpose of this course and I would strongly advise students against such a perception. The goal is and remains to help you become a better Computer Scientist and or a Software Engineer.

Text Book

Frank M. Carrano Janet J. Prichard, "Data Abstraction and Problem Solving with Java: Walls Mirrors". 3rd edition. Pearson - Addison Wesley Longman, Inc. 2010.

Text Book is Optional for this course. I would argue that the lecture materials I provide absolutely suffices to do well in this course.

Prerequisites

I expect that all students will come to this course with a basic knowledge of computer science (CSC 1301 - 1302 or equivalent) and basic knowledge of computer programming in Java. A strong mathematical background is also important (in particular, knowledge of single-variable calculus is helpful but not strictly necessary). Most importantly - this course will concentrate on Data Structures and NOT on Java review

Attendance

Attendance in lecture and in lab is required for all students, except in the case of illness or family emergency. Please let me know as far in advance as possible if you will be missing a lecture. Unexcused absences (for all or part of a lecture/lab) which amount to more than 10% absences can result in your withdrawal from the class with a resulting grade of "PW" and eventually "W" if you have less than 6 withdrawals OR "F" if you have more than 6 withdrawals over your degree program.

Attendance will be taken at the beginning of (either or both) lecture and or lab , showing up more than 10 minutes late will result in an absence.

Additionally, laptops will be required in lecture. Please reach out to me privately if this presents a problem so that we can figure out a solution.

Course Topics

The below represents the schedule of our course. The course is divided into four major units, with a fifth flexible unit depending on the pace of the class. Midterm will be given after the second, and a comprehensive (cumulative) final exam will be given at the end of the course.

This schedule provided below is a general plan for the course and deviations most likely may be necessary. Any deviation is at the sole discretion of the instructor.

Unit 1: Programming Review & Basic Algorithms (08/23-09/06)

In this unit, we will briefly review key concepts from mathematics (mainly logarithms) and computer science (namely, basic coding and the fundamentals programming). We will then explore some basic algorithms, like finding the mean/max/min/range of a list and searching for an item in a list. The lecture plan is as follows:

- (1) **Divide-and-Conquer (Part 1)** In this lecture, we will explore a “divide-and-conquer” algorithm for playing a number-guessing game.
- (2) **Divide-and-Conquer (Part 2)** In this lecture, we will learn how to write code for a program that uses the strategies we developed in Lecture 1.
- (3) **Searching & Mathematical Algorithms** In this lecture, we will explore linear search, binary search, algorithms for finding statistics of a list (max, min, range, mean, mode), and algorithms for finding square roots and logs using a divide-and-conquer approach.
- (4) **Proofs** *I may decide to skip Proofs altogether. I will keep you informed.* In this lecture, we will explore mathematical proofs, which will be used in our analysis of the speed and space usage of algorithms. In particular, we will explore standard proofs (also known as direct or deductive proofs), as well as the specialized techniques of inductive proof and proof by contradiction.
- (5) **Coding Style & Object-Oriented Programming** In this lecture, we will deep-dive into what it means to write code with good style, and explore industry standards for clean code. We will also review basic object-oriented programming principles and how one can apply them to real-world situations.

Unit 2: Asymptotic Analysis (09/08-09/15)

In this unit, we will learn how to analyze how fast an algorithm is and how much space it uses up. We will discuss how computer scientists formally analyze performance, and use this to develop a number of classes of more performant and less performant algorithms. The lecture plan is as follows:

- (6) **Big-O Notation: Basics** In this lecture, we will explore the motivation for Big-O notation, learn how to count the approximate number of operations in a program, and learn about

classifying algorithms into the sets $O(n)$, $O(\log(n))$, and $O(1)$ based on their speed in the worst case.

- (7) **Big-O Notation: Formal Definitions** In this lecture, we will give a mathematically rigorous definition of big- O and the related concepts of big- Ω and big- Θ and go through some more examples of these concepts. As an aside, we will also explore the problem of coding a divide-and-conquer quadratic equation solver, which is a simple version of our first longer programming homework question.
- (8) **Big-O Notation: Space Complexity** In this lecture, we will explore how to analyze the space usage of an algorithm in a similar way to the way we analyze time complexity.

Unit 3: Sorting (09/20-10/04)

In this unit, we will explore the well-studied problem of sorting a list. We will delve into some more intuitive but less performant sorting algorithms first, then work our way into more performant algorithms. As part of this unit, we will deep-dive into recursion. Our lecture plan is as follows:

- (9) **Basic Sorting Algorithms** In this class, we will explore Selection Sort, Insertion Sort, and BubbleSort, which are intuitive but slow sorting algorithms.
- (10) **Recursion (Part 1)** In this class, we will deep-dive into recursion through some examples.
- (11) **Recursion (Part 2) & Arithmetic** In this class, we will continue exploring recursion through some more complex examples. We will also explore how computers do large-number arithmetic, which will tie into a tough homework question.
- (12) **MergeSort** In this class, we will explore the MergeSort algorithm both conceptually and in code.
- (13) **QuickSort & Medians** In this class, we will explore the QuickSort algorithm and the auxiliary concept of median-finding.

Unit 4: Data Structures (10/11 - 11/17)

In this unit, we will explore the most common ways to store data, including stacks, queues, heaps, and binary search trees (BSTs). Our lecture plan is as follows:

- (14) **CRUD & Lists** In this class, we will briefly note the common operations a data structure offers, and deep-dive into the different versions of the dynamic-length list data structure.

-
- (15) **Stacks & Queues** In this class, we will explore the stack, queue, and deque data structures and review some examples of when they are useful.
 - (16) **Graphs & Trees** In this class, we will explore graphs and trees and what they mean/how they are used in computer science.
 - (17) **Graph Traversal** In this lecture, we will explore the breadth-first-search and depth-first-search algorithms for traversing a graph.
 - (18) **Priority Queues & Heaps** In this class, we will explore the concept of a priority queue and use our knowledge of trees to explore the heap, a structure that can be used to implement a priority queue.
 - (19) **Maps & Hashing** In this class, we will deep-dive into sets and maps, and explore how the concept of hashing makes sets and maps run quickly.

Unit 5: Advanced Topics (If we have time)

In this unit, we will explore more advanced topics from computer science and software engineering. Our lecture plan is as follows:

- (20) **Software Engineering Basics** In this lecture, we will explore some basic skills a software engineer must have that are not commonly taught in school.
- (21) **Solving an Interview Problem** In this lecture, we will explore a particularly difficult example software engineering interview question and walk through the process of solving it.

Last Day of Lecture December 2

Final Exam

For Lecture F 11:00 - 12:15 PM :

Final - Wed December 8 from 10:45 AM-13:15 PM

For Lecture F 02:15 - 3.30 PM :

Final - Wed December 8 from 2:15 - 4:45 PM

The final exam will cover the entire course.

The date and time of the exam is based on which section of the course you are enrolled in. This information can be found on the University website.

[<https://registrar.gsu.edu/document/atlanta-campus-final-exams/?wpdmdl=5216>](https://registrar.gsu.edu/document/atlanta-campus-final-exams/?wpdmdl=5216)

Anyone who misses the final for an unexcused reason or without notifying me in advance will receive a zero on the final, which will in most cases result in a course grade of D or F.

1. No make up exam will be given. If you will be absent for an exam due to sickness, your case may be considered (i.e. you may or may not be given a make up exam) based upon a letter from a medical doctor or a legal office written on that doctor's letter head or the legal office's letterhead, stating that you were unable to attend school (and hence take the exam)

on the given day. Absolutely no make up or excuse for the final exam. If for some medical or legal reason you cannot take the final exam, then you should consider applying for an Incomplete.

2. I will give neither extra assignment nor exam at any time during the semester to boost your grade.

Grades

Your grade in this class will be weighted based on five components:

- Weekly Labs: 20%
- Attendance: 10%
- Programming Assignments: 25%
- Midterm: 20%
- Final: 25%

At the end of the course, your weighted numerical average will be converted into letter grades based on the following scale. If you are in the range below you are guaranteed that letter grade, but your grade might be improved with a +/- depending on the distribution. For example, a 79% is guaranteed a C, but could be upgraded to a C+ or a B-.

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

0.1 Labs (20%) of final grade

Every week students will meet for a Lab where the idea is to solve Data Structure problems via programming. Topics will be directly related and correspond to the topics covered in the lectures. Lab problems are generally expected to be solved by the next day depending on how challenging the problems provided are.

Except for documented medical or legal emergencies any absence in a lab will result in a score of zero for that particular-lab assignment. **Late submissions to labs will not be accepted. Absolutely no exceptions, please. If you disagree please feel free to drop the course**

Attendance (10%) of final grade

Please use the link (<https://ousp.cs.gsu.edu/seat/>) to record your seat in the classroom. The record will also work as the attendance sheet. NOTE: DON'T USE ANY PROXY TO DO IT. Users can only open the website using the campus network.

Programming Assignments (25%) of final grade

There will be programming assignments. You will generally have one week to complete each programming assignment. It will be due at 11:00 PM EST. Your grade will decrease by 25 points each day it is late, with the lowest reduction being 50 points. After this point you will not be able to submit the assignment.

All submissions will be graded on correctness, code design, and code style. Answers must work correctly on all of the staff's test cases, be clearly and elegantly designed, and be written in a clear and consistent style to receive full credit.

Midterm (20%) and Final (25%) of final grade respectively

This course will have one midterm and a cumulative final exam. All exams will be closed-book, closed-notes. Exams are designed to challenge every student in the class; as such, the staff may curve grades upward for particularly difficult exams.

Your final exam grade will NOT replace your lowest midterm grade for whatever reasons you may choose to bring.

Grade Disputes

You have one week from when a grade is posted to dispute it. In the case of a written assignment being returned, it will be a week from when it is passed out. After the week has passed, disputes will not be honored. To dispute a grade, talk to me during office hours.

Academic Integrity (please read**)**

Don't cheat. Cheating is defined to be submitting others' work as your own, or allowing another student to submit your work as theirs. You may discuss programming assignments verbally with other students in this class, but you must write the code yourself. You may not share code with other students.

When in doubt, please reach out to me directly or refer to <https://deanofstudents.gsu.edu/faculty-staff-resources/academic-honesty/>. Keep in mind that the course staff is easily able to detect instances of plagiarism, particularly with the tools available to us, such as MOSS. Any instances of cheating will be dealt with harshly.

Access and Accommodation

Students with disabilities needing academic accommodation should: (1) register with and provide documentation to the Office of Access and Accommodation Services; (2) bring to a letter to the instructor indicating the need for accommodation and what type. This should be done during the first week of class. For more information about services available to GSU students with disabilities, contact:

The Margaret A Staton Office of Disability Services
Suite 230 Student Center

Georgia State University
Voice and TDD
Telephone: (404) 463-9044
Web Address: www.gsu.edu/disability

Covid-19 Statement

Attendance Policies

Students who want to do well in this course will attend class following the class attendance policy. You will need an excused absence due to illness. GSU has a new process for students seeking excused absences through the Dean of Students Office. Please submit documentation to <https://deanofstudents.gsu.edu/student-assistance/professor-absence-notification/>.

I will then be notified by the Dean of Students of any excused absences. Should a student test COVID positive, any accommodations to the class attendance policy will be informed by evolving guidance from the CDC on quarantine. In most cases there will be no major change to mode of course delivery, so students will be responsible for collecting notes for missed in-person classes and making up any work they miss during quarantine. Anyone who has a positive COVID test is encouraged to alert the university so that appropriate contact tracing can be conducted.

Wearing Masks in Class

You probably have an opinion on the effectiveness and use of masks to limit the spread of COVID-19 but wearing a face mask is not required in Georgia State classrooms. I will be wearing my face mask, and you are encouraged to wear yours. If you choose not to wear a face mask there is no penalty, and students should not engage in any type of disruptive behavior towards those who have made a different choice about wearing a mask.

Course Resources

Office Hours

The staff will have ample office hours this semester as noted in the first page of this syllabus. Please use them! We are happy to answer any questions you may have about the course, careers in software engineering, interview prep, or anything computer science/software engineering related.

Course Documents (iCollege)

I will post all lecture materials, assignments, and other relevant documents on iCollege.

Day	Topics Covered	Remarks
08/23/21, M	Chapter-1 Divide and Conquer.	
08/25/21, W	Chapter-1 Divide and Conquer	Lab1 Release (F) : DiagnosticExam (08/27)
08/30/21, M	Searching & Mathematical Algorithms	Assignment 1 Release. Due: 09/08/21 ,11.00pm ET.
09/01/21, W	Proofs	Lab2 Release (F): 09/03
09/06/21, M	Coding Style & Object-Oriented Programming	
09/08/21, W	Big-O Notation	Lab3 Release (F): 09/10
09/13/21, M	Big-O Notation	
09/15/21, W	Big-O Notation	Lab4 Release (F) : 09/17
09/20/21, M	Basic Sorting Algorithms	
09/22/21, W	Recursion (Part 1)	Lab5 Release (F) : 09/24
09/27/21, M	Recursion (Part 2) & Arithmetic	
09/29/21, W	MergeSort	Lab6 Release (F) : 10/01. Review the questions that you are not understand in the lab, no extra material
10/04/21, M	QuickSort & Medians	Assignment 2 Release. Due: 10/11/20 11.00pm.
10/06/21, W	Mid-term exam	Lab7 Release (F) : 10/08
10/11/21, M	CRUD & Lists	Last Day to Withdraw : October 12.
10/13/21, W	CRUD & Lists	Lab8 Release (F) : 10/15
10/18/21, M	Stacks & Queues	
10/20/21, W	Stacks & Queues	Lab9 Release (F) : 10/22
10/25/21, M	Graphs & Trees	
10/27/21, W	Graphs & Trees	Lab10 Release (F) : 10/29

11/01/21, M	Graph Traversal	
11/03/21, W	Graph Traversal	Lab11 Release (F) : 11/05
11/08/21, M	Priority Queues & Heaps	
11/10/21, W	Priority Queues & Heaps	Lab12 Release (F) : 11/12
11/15/21, M	Maps & Hashing	
11/17/21, W	Maps & Hashing	Lab13 Release (F) : 11/19
11/22 - 11/26	THANKSGIVING BREAK	
11/29/21, M	Software Engineering Basics	Assignment 3 Release.
12/01/21, W	Solving an Interview Problem	No Lab. Students complete Assignment 3 due 12/06 11pm ET 12/03
12/06/21, M	Selected Data Structures Review	CLASSES END
12/08/21, T	FINAL EXAM	Final Exam released at 10am ET. Due 12/09/20 : 11pm ET

Expectations

The expectations for software engineers and computer science students are very high, as are the expectations of this class. Understanding data structures and algorithms is critical to a career in software engineering or any computer-science related fields, and is important in any quantitative field (and, I would argue, in life in general). Accordingly, it is important that you devote the necessary time (up to 8-10 hours/week) outside of class to develop a strong understanding of the material. Please come to office hours as much as you need; the TAs and I are here to help you succeed.

Throughout this class, we will focus on comprehensive and clear **communication**. Communication is an extremely important skill to have in any career or higher education program, but is particularly important in computer science since people like us communicate both with words and with code. We will place a high value on simplicity and clarity in our code, written explanations, and verbal explanations.

To Reiterate :

- This schedule provides a general plan for the course; deviations may most likely be necessary. Any deviation is at the sole discretion of the instructor.
- Students are obligated to get acquainted with my class guidelines outlined in this syllabus. I consider registration in my class as acceptance of all my teaching rules and class policies.

With that all said, welcome to CSC 2720! I'm very excited for us to dive into our study of data structures this semester. I really enjoy this part of computer science, and hope you will too.