CSC 2720 - Data Structures

Shiraj Pokharel, PhD
Lecturer, Department of Computer Science
Georgia State University, Atlanta

Spring 2022

Course Staff

Instructor: Shiraj Pokharel
Email: spokharel3@gsu.edu
Webex: <https://gsumeetings.webex.com/meet/spokharel313>
Office: 25 Park Place #716

Teaching Assistant: Sarwan Ali sali85@student.gsu.edu
Teaching Assistant: Hari Priya Vuppu hvuppu1@student.gsu.edu
Teaching Assistant: Harsha Duddu hduddu1@student.gsu.edu
Teaching Assistant: Yueyang Liu yliu114@student.gsu.edu
Teaching Assistant: Naveen Kumar Kalaga nkalaga1@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 14979 Sec 002 , CRN 17841 Sec 006 , CRN 14980 Sec 008. CRN 21756 Sec 004
M, W 12:30 pm - 01:45 pm (Petit Science Center 169)

CRN 14986 Sec 020, CRN 14984 Sec 018, CRN 14985 Sec 022, CRN 21767 Sec 024
M, W 03:30 pm - 04:45 pm (Langdale 600)

Office Hours

Shiraj's Office Hours
T, R 14.30 - 15.30 PM . In person or Webex.

TA Office Hours

Will be Posted by the TAs in iCollege

About this Course

This course will focus on the study of **data structures** and some corresponding **algorithms** in the Java programming language. This is a programming heavy course However, this is NOT a course which teaches you how to write computer programs in Java !

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.

Ofcourse, 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 data structures 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.

Prerequisites

Prerequisites: CSC 1302, MATH 2211, and CSC 2510 or MATH 2420 with grades of “C” or higher.

Learning Outcomes

- Understanding of fundamental Data Structures including Arrays, linked-lists,stacks, queues, trees, binary search trees, graphs, priority queues, and hash-tables.
- Understanding of fundamental abstract data types which can include: Maps and Sets.
- Ability to program data structures and use them in implementations of abstract data types.

- Ability to devise novel solutions to small scale programming problems involving data structures.
- Understanding of basic algorithmic complexity and the ability to estimate the algorithmic complexity of programs.
- Ability to sensibly select appropriate data structures for programming problems and to justify that choice.

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 during (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.

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: Basic Algorithms and Programming practice with Strings & Arrays

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. This unit also would have rigorous in class programming sessions with Strings and Arrays, as part of lectures. The lecture plan is as follows:

(1) **Divide-and-Conquer - 2 Lectures** In this lecture, we will explore a “divide-and-conquer" algorithm for playing a number-guessing game and learn how to write code for a program that uses the strategies we developed.

(2) **Searching & Mathematical Algorithms - 1 Lecture** 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.

Unit 2: Asymptotic Analysis

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. This unit also would have rigorous in class programming sessions with Strings and Arrays, as part of lectures. The lecture plan is as follows:

(3) **Big-O Notation: Basics - 1 Lecture** 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.

(4) **Big-O Notation: Space Complexity - 1 Lecture** 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: Recursion and Sorting

As part of this unit, at first, we will deep-dive into recursion both theoretically and with hands on in class programming. 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. Our lecture plan is as follows:

(5) **Recursion (Part 1)- 1 Lecture** In this class, we will deep-dive into recursion through some examples.

(6) **Recursion (Part 2) - 1 Lecture** In this class, we will continue exploring recursion through some more complex examples.

(7) **Basic Sorting Algorithms - 2 Lectures** In this class, we will explore Selection Sort, Insertion Sort, and BubbleSort, which are intuitive but slow sorting algorithms.

(8) **QuickSort - 1 Lecture** In this class, we will explore the QuickSort algorithm both conceptually and in code.

(9) **MergeSort- 1 Lecture** In this class, we will explore the MergeSort algorithm both conceptually and in code.

Midterm - 02/21/2022

This midterm will cover topics up to the class meeting before the midterm.
Semester Midpoint is March 1. Last day to withdraw and possibly receive a 'W'
Ref : GSU Calendar for authoritative information.

<https://registrar.gsu.edu/registration/semester-calendars-exam-schedules/>

Unit 4: Data Structures

In this unit, we will explore the most common ways to store data, including Linked lists, stacks, queues, heaps, Trees, Maps and Graphs - in both concept and code. Our lecture plan is as follows:

(10) **CRUD & Linked Lists - 3 Lectures** In these lectures, 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.

(11) **Stacks & Queues - 2 Lectures** In these lectures, we will explore the stack, queue, and deque data structures and review some examples of when they are useful.

(12) **Maps & Hashing - 1 Lecture** In this class, we will deep-dive into sets and maps, and explore how the concept of hashing makes sets and maps run quickly.

(13) **Binary Trees - 3 Lectures** In these lectures, we will explore trees and what they mean/how they are used in computer science.

(14) **Graphs & Graph Traversal- 3 Lectures** In these lectures, we will introduce the graph data structure and explore the breadth-first-search and depth-first-search algorithms for traversing a graph.

(15) **Priority Queues & Heaps - 2 Lectures** 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.

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:

(16) **Software Engineering Basics** In this lecture, we will explore some basic skills a software engineer must have that are not commonly taught in school.

(17) **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 04/25/2022

Final Exam

Date, Time : TBD
The final exam will cover the entire course.

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 four components:

- Weekly Labs: 25%
- Programming Assignments: 25%
- Midterm: 20%
- Final: 30%

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

Labs (25%) 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 in 2 days 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 be penalised by 25% for turning in the lab a day after submission deadline. After that no submissions will be accepted.Absolutely no exceptions.

Programming Assignments (25%) of final grade

There will be programming assignments. You will generally have one week to complete each programming assignment. Late submissions to assignments will be penalised by 25% for turning

in the assignment a day after submission deadline. After that no submissions will be accepted. Absolutely no exceptions.

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 (30%) 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 Accomodation

Students with disabilities needing academic accommodation should: (1) register with and provide documentation to the Office of Access and Accomodation 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 an 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.

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.