The University of Texas at El Paso
Google Tech Exchange – Spring 2023
CS 4390 – Applied Data Structures

Applied Data Structures Syllabus

Course Website	Content is hosted on classroom.google.com and repl.it
Class Times & Instructors	See below in Section 4 (Staff & Course Times)
Class Location	Will be hosted virtually via Google Meet
# of Credits	3 credits

1. General Course Description

1.1 Course Description

This course will teach students the skills needed for technical coding interviews at companies like Google. It will focus on understanding how to choose optimal algorithms and data structures for different problems, how to apply them, and how to explain their

reasoning. Topics covered will include hash tables, recursion, linked lists, trees, and graphs.

1.2 Lecture Structure

Students in the course will be split into different sections. Lectures for each section will occur twice a week (either Monday/Wednesday or Tuesday/Thursday) for 95 minutes. Lectures focus on introducing new concepts and having students practice coding exercises. Some lectures will include short quizzes.

1.2 Prerequisites

Students interested in taking the course must be able to:

1. Describe, implement (using basic object-oriented programming constructs), and use the following data structures:
    a. Arrays
    b. Singly Linked Lists
    c. Stacks
    d. Queues
    e. Binary Trees / Binary Search Trees (basic understanding)

2. Use (as a black box) the following data structures when solving coding problems (no need to know how they are implemented):
    a. Hash Maps / Dictionaries
    b. Hash Sets / Sets

3. Describe and implement the following searching and sorting algorithms:
    a. Linear Search
    b. Binary Search
    c. Merge Sort
    d. Bubble Sort (or another quadratic sorting algorithm)

4. Use the Big-O notation to describe the running time of non-recursive algorithms and basic data structure operations (1a-e).

5. Use basic data structures (1a-e) and algorithms (2a-d) when solving programming problems that explicitly ask for their use.

6. Write and trace algorithms (to ensure solution correctness and understanding) that use the following:
    a. Primitive variable types (booleans, integers, floating-point numbers, and characters)

b. Non-primitive (reference-based) variable types, including arrays, strings, and other user-defined classes / data structures (1a-e)
c. Arithmetic (+,-,*, /), relational (>,>=,<,<=,==,!=), and boolean (and, or, not) operators
d. Functions / methods (primitive vs. non-primitive parameter passing)
e. If-statements
f. Loops
g. Recursion (basic)

7. Apply (at a basic level) the following problem-solving constructs/strategies: problem decomposition, pattern recognition, abstraction, algorithmic thinking, solution evaluation, and reflection.

1.3 Course aims and learning objectives

This course aims to teach students to:

- Improve their understanding of and comfort applying data structures and algorithms to solve real-world problems
- Identify optimal data structures or algorithms when solving problems and explain their reasoning
- Learn and practice skills required for technical interviews
- Improve their ability to solve programming tasks within a time constraint

Level 3: Synthesis and Evaluation:
Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to:

1. Use the Big-O notation to analyze runtime and space efficiency of (recursive and non-recursive) algorithms and data structure operations.

2. Identify and evaluate algorithm design and data structure tradeoffs when solving real-world problems.

3. Articulate and defend decisions when problem-solving using metrics concerning efficiency, correctness, coverage, robustness, and adequacy in fulfilling problem requirements.

4. Apply, modify, and extend fundamental data structures when required to build better solutions.

5. Design, apply, and communicate rigorous testing strategies that demonstrate problem understanding, edge-case consideration, correctness, and strong problem-solving processes.

6. Apply (at a competitive level) problem-solving skills required for solving technical problems, including problem decomposition, question formulation, pattern recognition, abstraction, solution analysis, whiteboarding, tracing/testing, data structure and algorithm selection, and decision making (articulation and defense).

Level 2: Application and Analysis:
Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work out a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able to:

1. Assess, compare, and use the following data structures when solving real-world and technical interview problems:
    a. Array Lists
    b. Dictionaries
    c. Sets
    d. Linked Lists
    e. Stacks
    f. Queues
    g. Binary, Binary Search, and Balanced Binary Trees
    h. Graphs

2. Design and implement non-trivial recursive algorithms and identify contexts where recursive solutions are commonly preferred.

3. Trace non-recursive and recursive algorithms (call stack understanding and simulation).

4. Use common algorithm design techniques and identify when they need to be applied

Level 1: Knowledge and Comprehension
Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to:

1. Describe how technical coding interviews are conducted at companies like Google.
2. List the skills required to succeed during a technical interview.
3. Describe the *dos* and *don'ts* when interviewing for a technical position.

2. Course Resources

2.1 Asking Questions

If students have questions outside of class, there are several venues where they can ask them:

- Course email:
    - If students have questions that they don't want to send publicly in the group chat (e.g. a question that includes a snippet of their code), they can email the course staff at tx23-ads-leads@techexchange.in
- Office hours
    - Instructors will host office hours throughout the semester via video chat. You should see invitations to office hours on your Google Calendar. Feel free to attend office hours for any of the instructors, even if they aren't the instructor for your section.
- Individually email your instructors or the Course PgM
    - If you have questions specifically for your instructors or the Course PgM - all emails can be found within Google Classroom
- Discord
    - You should've been added to our Google Tech Exchange 23 Discord. In the different channels you can ask general questions, chat with instructors, TAs mentors,mock interviewers and other students
    - Channels: #general-techX → general TechX questions
    - Channels: #applied-data-structures → Course questions, student collaboration, office hour reminders
    - You can also DM any of the ADS Instructors
- Edlyft
    - You can access the Edlyft server on discord.
    - For specific Edlyft questions DM: Erika Hairston (Edlyft) ehairston#2641
    - Please reach out to TX team if you have further questions about Edlyft

2.2 Required textbooks

None. Students will access assignments and readings via Google Classroom and Repl.it. Some useful resources that you may want to consult are:

- Python tutorial by tutorialspoint.com
- LeetCode tutorials on common data structures & algorithms
- Python Wiki

2.3 Department resources

The course will be co-taught with instructors from Google and faculty from partner universities.

4. Staff & Course Times

Section	Role	Name	Email
A M/W 9:10-10:45am Eastern	Google Instructor	Daniel Shanker	shankerd@google.com
	Faculty Instructor	Jinsheng Xu	jxu@ncat.edu
B M/W 9:10-10:45am Eastern	Google Instructor	Alice Reyzin	areyzin@google.com
	Faculty Instructor	Andrea Edwards	aedwards@techexchange.in
C M/W 1:10-2:45pm Eastern	Google Instructor	Nathan Tempelman	natet@google.com
	Faculty Instructor	Inna V Pivkina	ipivkina@cs.nmsu.edu
D M/W 3:10-4:45pm Eastern	Google Instructor	Alice Reyzin	areyzin@google.com
	Faculty Instructor	Alcibiades Bustillo-Zarate	alcibiades.bustillo@upr.edu
E M/W 3:10-4:45pm Eastern	Google Instructor	Nicholas Reid	nicholasreid@google.com
	Faculty Instructor	Sajid Hussain	sajid.fisk@gmail.com
F T/Th 9:10-10:45am Eastern	Google Instructor	Daniel Shanker	shankerd@google.com
	Faculty Instructor	Alireza Izaddoost	aizaddoost@csudh.edu
G T/Th 5:10-6:45pm Eastern	Google Instructor	Nathan Tempelman	natet@google.com
	Faculty Instructor	Kianoush Gholamiboroujeni	drboroojeni@gmail.com

5. Weekly Topics

Date	Topic
Jan 11/12	1.1 Python Review

Jan 16/17	**No class for MLK day**
Jan 18/19	1.2 Lists and Tuples
Jan 23/24	1.3 Complexity
Jan 25/26	2.1 Dictionaries
Jan 30/31	2.2 Sets
Feb 1/2	3.1 Recursion Fundamentals
Feb 6/7	3.2 More Practice with Recursion
Feb 8/9	Review
Feb 13/14	4.1 Searching
Feb 15/16	4.2 Sorting
Feb 20/21	**No class for Presidents Day**
Feb 22/23	5.1 Linked List Fundamentals
Feb 27/28	5.2 Practice with Linked Lists
Mar 1/2	6.1 Stacks and Queues
Mar 6/7	6.2 Practice with Stacks and Queues
Mar 8/9	Review
Spring Break	
Mar 20/21	7.1 Binary Trees
Mar 22/23	7.2 Tree Traversals, BSTs
Mar 27/28	7.3 Binary Tree List Representation, Heaps
Mar 29/30	7.4 Binary Trees Practice
Apr 3/4	8.1 Building Graphs
Apr 5/6	8.2 BFS Part 1
Apr 10/11	8.3 BFS Part 2
Apr 12/13	8.4 DFS Part 1
Apr 17/18	8.5 DFS Part 2
Apr 19/20	8.6 Graphs Practice/Dijkstra's Algorithm
Apr 24/25	Review

Apr 26/27	Review

5.1 Tentative Assignment Dates

	HW Assigned	HW Due	Exam Assigned	Exam Due
HW 0	1/12	1/19		
Unit 1	1/19	1/31	1/31	2/6
Unit 2	1/26	2/7	2/7	2/13
Unit 3	2/2	2/14	2/14	2/20
Unit 4	2/14	2/23	2/23	3/1
Unit 5	2/23	3/7	3/7	3/13
Unit 6	3/2	3/14	3/14	3/20
Unit 7	3/21	4/6	4/6	4/12
Unit 8	4/4	4/21	4/20	4/23

6. Assessment and Grades

6.1 Components of course grade

% of grade	Component	Description
10%	Section Attendance	Please attend the entire class and participate actively in group work
10%	In-class Quizzes & Surveys	There will occasionally be in-class quizzes and surveys. These will be relatively short (should take less than 15 minutes) and will be used to assess your understanding of concepts that have been covered in the course up to that point. They will be graded for completion, but NOT for correctness.
40%	Homeworks	Homework problems will be assigned for each unit.

		You **are** allowed to collaborate with others beside your instructors and TAs. These problems are designed to help you attain the unit's learning outcomes and sharpen your problem-solving skills in preparation for exams.
40%	Exams	There will be approximately 1 exam for each unit in the course. Exams will be time limited (1-3 hours) assessments that include multiple choice and/or coding problems. You are **not** allowed to collaborate with others for exams. Your TA will grade exams using unit tests and manual evaluation of your code.

6.2 Grading Scale

Some assignments during the course will be graded on a curve. Your grade will only be curved up - we will not curve your grade down. In other words, The final course grade will be determined by your overall percent score in the course:

Percent of points earned	Letter Grade
90+	A
80+	B
70+	C
60+	D
< 60	F

6.3 Late Work policy

For each day that a homework is late, 10% of the total homework points will be deducted. For example if a homework is worth 20 points and it's 3 days late, 6 points will be deducted. Homework submitted more than 5 days late will not be accepted (solutions will be posted 5 days after the homework is due). For example, if homework is due by Wednesday EOD, you can submit it up to Monday EOD but not after. If you need to request an extension, fill out this form.

To help provide flexibility, we will drop your lowest homework score and your lowest quiz score from your grade calculation. For example, if there are 4 homework assignments

throughout the course and you score: 86%, 94%, 92%, 82% – we will drop the 82% score from your final grade calculation.

6.4 Attendance policy

Students are expected to attend classes regularly and on time.

- You must attend classes on time and actively participate during group work to get credit. If you have to miss class, please reach out to the course instructors.
- There will be brief quizzes and surveys during some classes. These quizzes will be graded for completion, NOT for correctness. The quizzes are intended to provide feedback to your instructors.

6.5 Plagiarism policy & Guidelines for collaboration

All instances of plagiarism will be directed to the university administration, which will conduct the appropriate hearings. Use the table and guidelines below to determine if you can collaborate on a given assignment.

What kind of help can I get on an assignment?

Type of work	Help allowed from
Exercises during class	Your instructor, TA, or other student enrolled in this course
Quizzes during class	No help permitted
Homework assignments	Your instructor and TA. Read the homework instructions to determine how to collaborate with classmates.
Exams	No help permitted

Students are ***not*** allowed to share or look at another student's code for any exams. The following are **not allowed** for exams:

- Copy/pasting code. This includes but is not limited to copy/pasting code from:
    - Another student enrolled in the class
    - A person not enrolled in the class
    - Any online source
- Sharing your code with someone else. This includes but is not limited to:
    - Posting your code to a public github repo (repos are public by default - beware!)
    - Emailing your code

    - Sharing a google doc or other file containing your code
    - Sending code snippets in a chat/text
    - Screen sharing your code for someone else to see
    - Writing your code on a whiteboard for someone else to take a picture of
    - Reading your code aloud to someone else
    - Two or more people discussing line-by-line what code to write aloud over e.g. video chat as you are writing it, even if you are both typing up the code separately
- Reading someone else's code
- Asking other students questions about the quiz or exam.

We should not see two exams that have identical snippets of code. If that occurs, both submissions will receive an automatic **0**. Students should also not copy code from online sources. That is, we should not see code snippets in homework solutions that also appear online.

If you're worried that you'll be mistakenly flagged for plagiarism, include a comment in your submission explaining why. Students in general must be prepared to explain any code they submit.

7. ADA Policies and Procedures

If a student needs particular accommodations to be made, they must fill out this form.