EECE.3220: Data Structures

Spring 2019

Syllabus

Course Meetings

Section 202: MWF 3-3:50 PM, Ball 208

Course Website

Main page: http://mjgeiger.github.io/eece3220/sp19/

Schedule: http://mjgeiger.github.io/eece3220/sp19/schedule.htm

Course Discussion Group

All course announcements will be posted on the course Blackboard page. You are responsible for checking that site, as well as the sites listed above, on a regular basis.

Instructor

Dr. Michael Geiger
E-mail: Michael_Geiger@uml.edu
Office: Ball 301A
Phone: 978-934-3618 (x43618 on campus)
Office hours: Monday/Wednesday/Friday 1-1:50 PM; Tuesday/Thursday by appointment only

During office hours, student questions are our top priority. Feel free to stop by the office, e-mail questions, or schedule a one-on-one appointment. Office hours are subject to change.

Textbook

Larry Nyhoff, *ADTs, Data Structures, and Problem Solving with C++*, 2nd edition, 2005, Pearson/Prentice Hall.
ISBN: 0-13-140909-3

Course Overview

Catalog Description: Covers algorithms and their performance analysis, data structures, abstraction, and encapsulation. Introduces structures and their physical storage representation. Studies stacks, queues, linked lists, trees, graphs, heaps, priority queues, and hashing. Discusses efficient sorting (quicksort and heapsort) and introduces experimental analysis of algorithms as applied to engineering applications. Examines several design issues, including selection of structures based on what operations need to be optimized (insertion, deletion, traversal, searching, sorting, evaluation), encapsulation of algorithms using class and template techniques, and how and when to use recursion (versus explicit stack-based techniques). Laboratories include programming of data structures in C++ and Java applied to Engineering.

Prerequisites: EECE.2160: ECE Application Programming

Course Objectives: By the end of this course, you should understand and be able to use all of the following:

1. **C++ Programming:** Fundamentals of the C++ programming language
2. **Data Structures:** Common data structures, including arrays, vectors, stacks, queues, linked lists, trees, and hash tables
3. **Algorithmic Complexity:** Analyzing the performance of data structures and algorithms
4. **Object-Oriented Programming:** Classes, objects, templates, inheritance

Grading: Grades will be computed on an A to F scale; no A+ grades will be assigned, in accordance with UMass Lowell policy. The weights assigned to the various items are:

Programming/homework assignments	55%
Exam 1	15%
Exam 2	15%
Exam 3	15%

Incomplete grades will only be given in exceptional situations, and the student must be passing the class at the time the grade is requested.

The following rubric describes how grades will be assigned if no grading curve is applied. A grading curve may be used at the instructor's discretion, depending on the overall course average at the end of the term. Grades will not be curved down, meaning that the table below describes the minimum letter grade you will earn for a final average in each of the ranges shown:

Range	Grade	Range	Grade
> 92	A	78-79	C+
90-92	A-	73-77	C
88-89	B+	70-72	C-
83-87	B	68-69	D+
80-82	B-	60-67	D
		< 60	F

Your grade is based strictly on the work you do during the semester. Please do not ask for extra credit work to improve your grade—any extra credit work we give is available to the whole class, not just the students who ask for it.

Programming/homework assignments: Typically, you will have about one week to complete each assignment. Late assignments will lose 2^{n-1} points per *n* days late, including weekends and holidays. (So, -1 for 1 day late, -2 for 2 days late, -4 for 3 days late, etc.) You will submit your code through Blackboard.

For each program, you are allowed one resubmission to improve your grade without penalty. You must resubmit your code by the given deadline for that assignment; late penalties apply to late resubmissions. Late penalties on the original submission still apply to the resubmission—for example, an assignment that is 3 days late has a maximum score of 96 for the resubmission.

Exams: Make-up exams will only be offered in exceptional circumstances. You must notify your instructor as early as possible in order to determine an appropriate make-up date.

Class participation: You are responsible for all material discussed or announced in class. You are expected to attend class regularly and participate in any in-class discussions, as such exercises are essential to your learning. Although lecture attendance is not explicitly required, regular attendance will improve your understanding of the course concepts.

Academic Honesty

All assignments and exams must be completed individually unless otherwise specified. You may discuss concepts or material covered in class, but may not share any details of your solutions to assigned problems, including algorithms and code. Plagiarism (in this course, copying code from an outside source) is also unacceptable and will be treated as an instance of cheating.

Students are allowed to discuss assignments in general terms and to help one another fix specific errors, such as compiler errors or output formatting. In this case, students must note in their program header that they received assistance from a classmate. However, any code sharing—even if used only to help a classmate solve a specific error—is an academic honesty violation.

Any assignment or portion of an assignment violating this policy will receive a grade of 0 for all parties concerned. Depending on the severity of the infraction, or in cases of repeat violations, the instructor may give additional penalties, up to and including a failing grade in the course.

Further information on the University Academic Integrity policy can be found at:

https://www.uml.edu/Catalog/Undergraduate/Policies/Academic-Policies/Academic-Integrity.aspx

Course Schedule

This schedule contains a tentative schedule of topics we will cover throughout the term; the course website will contain the most up-to-date version. The web page will also describe which section(s) of the textbook are associated with each lecture and the due dates for each assignment.

The exam dates will be fixed shortly after the start of the semester. Tentative dates for the first two exams are shown below (during weeks 5 and 10), and the third exam will be held **during final exams, at a date and time to be determined by the registrar's office**.

Week	Date (M)	Lecture Topics
1	1/21	*No Monday lecture—classes start 1/22* Course introduction Going from C to C++
2	1/28	Going from C to C++ (continued)
3	2/4	Algorithmic complexity Abstract data types
4	2/11	Classes
5	2/18	*No Monday lecture—Presidents Day* *Tuesday, 2/19 follows Monday schedule* Classes (continued) **EXAM 1 (W 2/20 or F 2/22)**
6	2/25	Dynamic allocation Linked lists
7	3/4	Stacks
8	3/11	*No classes—Spring Break*
9	3/18	Queues
10	3/25	**EXAM 2 (M 3/25 or W 3/27)** Queues (continued)
11	4/1	Binary trees
12	4/8	*Tuesday, 4/9: Last day to withdraw* Heaps and priority queues Sorting algorithms
13	4/15	Hash tables
14	4/22	Composition and inheritance
15	4/29	Topics TBD Exam 3 Preview (Friday, 5/3) *Classes end Friday, 5/3*
	TBD	**EXAM 3: during finals; time/location TBD**