COMPSCI 186 Fall 2020 Syllabus

Marc Liberatore

August 25, 2020

(Note that the canonical version of this syllabus that will be updated with relevant changes is stored as a Moodle page; the PDF is the syllabus as it was before the start of the semester.)

Welcome

In this course, each voice in the classroom has something of value to contribute. Please take care to respect the different experiences, beliefs and values expressed by students and staff involved in this course. My colleagues and I support UMass's commitment to diversity, and welcome individuals regardless of age, background, citizenship, disability, sex, education, ethnicity, family status, gender, gender identity, geographical origin, language, military experience, political views, race, religion, sexual orientation, socioeconomic status, and work experience.

View this syllabus as a guide to the course. It provides important information regarding the course, its assignments, policies, grading, and available university resources.

You should read it once, thoroughly, at the start of the semester. However, this document should be considered a working document. It is possible throughout the semester that a topic may take more time than expected, topics or assignments may change, or some material may be canceled or delayed due to a snow day or another emergency. If that is the case, the syllabus and schedule will be updated and a revised version will be posted on the course web site.

Course overview

Description: This course introduces foundational abstract data types and algorithms. The main focus is on the use of data structures in designing and developing programs to solve problems in a variety of domains. Specific topics include lists, sets, maps, graphs, stacks, queues, searching, and sorting. There will be weekly programming assignments, programming and written exercises in discussion sections, regular self-assessments, and a cumulative final exam.

Prerequisites: C or better in COMPSCI 121 (or transfer credit, or appropriate score on the placement exam). This course is not a substitute for COMPSCI

187. If unsure of whether this course or COMPSCI 187 is more appropriate, contact instructor. 4 credits.

Objectives: COMPSCI 186 requires students to engage in extensive problem solving and critical thinking. In particular, through a project-based curriculum, students expand their ability to design, reason about, and implement non-trivial algorithms and data structures in a general-purpose programming language.

As the course progresses, project scope expands, and the amount of explicit guidance that students are given is decreased – to succeed, students must incrementally improve their programming ability. Thinking in abstractions, planning large (multi-file) programs, and learning to understand why a program does (or doesn't!) work as expected are activities that require significant critical thinking and problem solving.

In addition to programming projects, students complete take-home problem sets, about six group POGIL exercises, about six half-hour self-assessments, and a final exam. Among other things, these activities serve to evaluate other aspects of critical thinking and problem solving that are better assess in a written format. For example, students might be asked to determine the asymptotic running time of a snippet of code; to show (on paper) how a particular algorithm would operate on a given data set (such as a graph traversal algorithm); to identify logical errors in code; or to compare how operations defined by set theory and by a programming language's set implementation differ.

Learning outcomes

After completing this course, students should be able to:

- Design and write methods using the basic tools of imperative programming: variables and branches, loops over counters and linear data structures.
- Design and write methods using simple recursion for control flow.
- Analyze problems best solved with multiple classes, implement object-oriented solutions given some scaffolding or design advice, and determine which (if any) of the existing implementations of the List, Set, Map, Stack, and Queue ADTs are appropriate when solving these problems.
- Use, design and implement test methods to exercise both single methods and whole programs.
- Use both formal debuggers and program output to determine the cause of logical program errors.
- Explain the concept of worst-case running time analysis, and classify simple examples of linear, quadratic, and constant-time code.
- Categorize the worst-case running time of operations on arrays and on the List, Set, Map, Stack, and Queue ADTs, given information about their underlying implementation.
- Explain the O(n^2) sorting algorithms.
- Explain and implement breadth-first search, given an existing implementation of graphs.

- Explain the implementation details of array- and linked-list-backed Stacks and Queues.

What, when, where, who

COMPSCI 186: Using Data Structures
Online, asynchronous lectures using Echo360
Discussions meet synchronously online on some Mondays

Instructor: Marc Liberatore (please call me "Marc")
Email: liberato@cs.umass.edu (though see note below about CampusWire)
Phone: 413-545-3061 (on campus: 5-3061)
Office: Computer Science, Room 318 (I will not be in my office much, if at all, this semester, due to COVID)
Office hours: Tuesday and Thursday, 4pm via Zoom (see CampusWire for links)

Graduate Teaching Assistants (TAs):
Will Lee
Email: williamlee@cs.umass.edu
Office hours: (see CampusWire for links)

Mehmet Savasci
Email: msavasci@umass.edu
Office hours: (see CampusWire for links)

Undergraduate Course Assistants (UCAs):
Jonathan Julien, Stephen Scarano, Veronica Gusev, Zachary Schaffer, and Yan Chen.

Required and optional material

An Internet connection is required, since this is an online course.

A computer capable of running a recent version of the Java Development Kit is required. The ability to use Visual Studio Code is also strongly recommended, though not strictly required. As time permits, course staff may be able to help with minor technical issues, but we are not IT support staff; we cannot generally solve installation or configuration issues, especially remotely.

Most technical material will be presented in lecture or on the course website.

For students who want additional references, I suggest several *optional* resources:

- The Java Tutorials, likely already familiar to you from COMPSCI 121 or the equivalent, are guides to the Java language. I'll note in the schedule specific "trails" that you may find helpful.
- Similarly, the Java Platform API provides a comprehensive description of all classes Java Platform; we'll make extensive use of some of them, and they are fully documented by Oracle.

- Think Java: How to Think Like a Computer Scientist, 2nd edition, by Allen B. Downey and Chris Mayfield. Think Java is an alternative to the COMPSCI 121 textbook, and covers about the same material. It primarily focuses on the craft and science of problem solving with computers, with a secondary focus on Java. You may find this book useful to review before the start of the semester or during the first few weeks of this course.
- Learning Java, 4th edition, by Patrick Niemeyer and Daniel Leuck. This book is aimed at working programmers, not computer scientists, and focuses much more on use of Java, tooling in the Java ecosystem, and practical considerations than Think Java. You may find it useful when using some of the tools we require in this course, and when completing some of the advanced (and optional) parts of later programming assignments.
- Java Precisely, 3rd edition, by Peter Sestoft. If you want to know something about the Java language – syntax or semantics – this book is a great reference. Note that it is not a textbook or a how-to manual, but a reference book that explains what specific part of the language mean. It also provides some explanation of important parts of the Java standard library (also known as the class library or Java Platform API).

Code of conduct

- The course staff are committed to providing a friendly, safe and welcoming environment for all, regardless of level of experience, gender identity and expression, sexual orientation, disability, personal appearance, body size, race, ethnicity, age, religion, nationality, or other similar characteristic.
- Please be kind and courteous. There's no need to be mean or rude.
- Respect that people have differences of opinion and that differing approaches to problems in this course each carry a trade-off and numerous costs. There is seldom a single right answer to complicated questions.
- Please keep unstructured critique to a minimum. Criticism should be constructive.
- We will informally warn you, once, if you insult, demean or harass anyone. That is not welcome behavior. After that we will report your behavior to the Dean of Students office. We interpret the term "harassment" as including the definition in the Citizen Code of Conduct under "Unacceptable Behavior"; if you have any lack of clarity about what might be included in that concept, please read their definition. In particular, we don't tolerate behavior that excludes people in socially marginalized groups.
- Private harassment is also unacceptable. No matter who you are, if you feel you have been or are being harassed or made uncomfortable by a member of this class, please contact a member of the course staff immediately (or if you do not feel safe doing so, you should contact the Chair of the Faculty of CICS, currently Prof. James Allan, allan@cs.umass.edu, or the Dean of Students office). Whether you've been at UMass for years or are a newcomer, we care about making this course a safe place for you and we've got your back.

- Likewise any spamming, trolling, flaming, baiting or other attention-stealing behavior is not welcome.

Communication policy

Per the University Email Policy, you are expected to check your email regularly – at least once a day. I will use your UMass email address as your primary point of contact in all online tools we use (Moodle, CampusWire, and Gradescope) and as my primary means to contact you individually outside of class. Group announcement will be posted to CampusWire, which can be configured to send you via email whenever an instructor makes a post.

For course-content related questions (especially questions that other students might benefit from seeing the answers to), please use CampusWire. For other questions (like unusual logistics stuff), private messages on CampusWire or email are both OK, but please check the syllabus and course web site before emailing the course staff. If you send me (or the TAs) email, please include "COMPSCI 186" in the subject line to make sure we answer them in a timely fashion.

Course staff typically respond to emails and CampusWire questions within one to two business days, but I (Marc) do not typically respond to communications after about 5pm or on weekends. Course staff tend to get a higher volume of messages when a deadline is approaching. If you contact the course staff (that is, at least one TA and the instructor) at least two full business days *before* a deadline, you are guaranteed a reply before the deadline. Otherwise we'll do our best, but no guarantees.

Zoom!

We'll be using Zoom for "face-to-face" meetings in this class, such as the Monday discussion sections and office hours throught the week. You have a University-specific Zoom account (accessed through your NetID) that you'll need to use for this course. We expect that you'll use Zoom courteously. In particular this means:

- Keep your camera on (if at all possible). I understand it's a little strange for your classmates and friends to see your bedroom, kitchen, backyard, whatever, during class, but that's the new reality we're living in. This is especially important for when you're working in small groups during discussion.
- In larger meetings, please keep your microphone muted unless you're speaking. The person hosting the meeting (me, the TAs, the UCAs) will let you know the etiquette for speaking. Typically we'll use the "raise hand" feature, or if there are few enough people to fit on one Zoom screen, just ask you to actually raise your hand.
- Be forgiving and understanding of Zoom (and other tech) weirdness, and the fact that we're living in very weird times. For example, cut your classmates (or your instructor!) some slack if a family member starts

yelling in the other room when their microphone is live, or if they freeze on Zoom, or whatnot. We're all doing the best we can.

CampusWire

CampusWire is a online discussion management system. It will be used as the main hub for questions and answers in this course. CampusWire is a great tool but it can be abused. Please follow these guidelines in your use of CampusWire:

- You should use CampusWire to ask questions and get advice on assignments. But you may not use CampusWire to step through each problem you encounter in an assignment.
- You may not post assignment solutions to CampusWire, either in questions or answers to others' questions.
- If you must post code you are working on, you should do so only through private posts to the course instructors.
- You should not post code without a thoughtful and articulate question. Do not post code and ask only, "what is wrong with my code?" See, for example, http://stackoverflow.com/help/how-to-ask or https://jvns.ca/blog/good-questions/ for constructive advice on asking questions.
- You are encouraged to help other students by answering questions.

The course staff will monitor CampusWire and answer your questions in a timely manner (generally within a business day). But do not expect us to provide real-time answers on CampusWire, especially in the last few hours before an assignment is due!

If a question has already been answered in a previous post we may not respond to you. If a question does not follow the guidelines above we may not answer it. If we find that a private question is relevant to a larger audience, we may make mark it public to help others in the course.

Time management and what to expect

As a general guideline, the university suggests that students spend three to four hours of time on a class per credit hour. This is a four-credit course, therefore you should plan to spend twelve to sixteen hours a week on this class.

In a typical week, you will:

- either attend discussion (where there will be either a group activity) or take a graded self-assessment
- view pre-recorded lecture material and read the associated lecture notes as needed
- complete several short problems based upon the lectures materials (also known as "problem sets")
- complete one larger programming assignment (where you will spend the bulk of your time in this course)
- optionally, attend office hours

(Virtual) attendance

I expect you to attend lectures and discussions, and to complete assignments by their due date. That is, I expect you will keep up with posted lecture material (watch by the date listed in Echo360) and to attend (via Zoom) the scheduled synchronous discussion you are enrolled in.

- If you will be absent or miss deadlines due to religious reasons, you must provide me with a written list of such dates within one week of your enrollment in the course.
- If you will be absent or miss deadlines for a University-related event, such as an athletic event, field trip, or performance, you must notify me as soon as possible.
- If you are absent or miss deadlines for health reasons, I expect you to notify me as soon as possible and, if you seek excusal or an extensions, to provide written documentation.
- If you are absent or miss deadlines for other extenuating non-academic reasons, such as a military obligation, family illness, jury duty, automobile collision, etc., I expect you to notify me as soon as possible and provide written documentation (again, if you seek excusal or extensions).

If you add the class late, I will excuse you from missed work, but you are responsible for both notifying me when you add, and for completing the work on your own.

Incompletes

Incompletes will be granted only in exceptional cases, and only if you have completed at least half the course with a passing grade. Prior to that, withdrawal is the recommended course of action.

Schedule

Please see the main Moodle site for a specific schedule. Approximately:

Week 1: Java review
Week 2: Testing and debugging
Week 3 & 4: The List Abstract Data Type (ADT)
Week 5: The Set ADT
Week 6: The Map ADT
Week 7 & 8: Introduction to algorithms
Weeks 9, 10, 11: Graphs and search
Week 12: Implementing ADTs (Stacks and Queues)
Week 13: Introduction to recursion & course wrap-up

Grading

I urge you not to focus on numeric scores and grades. Most students get good grades in this course. The median is typically between a B and a B+, and very few fail. So instead, focus on where assessments show that you can or need to improve. A decade from now the grade you got in this course will be irrelevant, but what you learned about programming might just be crucial.

Nonetheless, in the midst of a busy semester we all end up with moments of triage in which we need to understand where to concentrate our efforts. So to give you a sense of the relative importance of each form of assessment, we expect the breakdown for the final course grade to be as follows:

30% programming assignments
25% problem sets
10% discussion exercises
20% self-assessments
15% final exam

The numerical cutoff for final course letter grade assignment will be made after all grading is completed. As a rough guide, expect to require at least a 93 to get an A, a 90 to get an A-, an 87 to get a B+, an 83 to get a B, an 80 to get a B-, and so on.

Individual grade items are not curved, so you should not get stressed about means, standard deviations, etc. related to particular scores you receive. Similarly, emailing us is not a viable strategy to increase your final grade. What matters is your weighted average; we do not give favorable (or unfair) treatment by raising or lowering individual students' letter grades.

There are no unannounced opportunities for extra credit in this course; please do not ask.

Also: It's 2020. Storage and bandwidth are virtually free. Back your work up, store it in the cloud, whatever. "My computer crashed" won't be acceptable as an excuse in this class.

I will retain all graded materials for this course until the end of next semester. If you wish to review them, please come to see me during office hours (or make an appointment).

You are responsible for monitoring your grades. Grades will be available through Moodle (though note that some will be available in Gradescope first) and you should check them regularly and review any provided feedback. If you encounter any issues with your grades, you will have one week past the first posting of a particular assignment's grade to Moodle to contact the course staff so that we can investigate. Please contact us via the regrade request system in Gradescope. We will not generally accept questions about an individual assignment's grade beyond this one week, so you must be prompt.

Programming assignments

I will post programming assignments about once a week, and you will typically have about one week to complete them. They will be posted to the course web site's assignment page and must be submitted through Gradescope. Programming assignments may be completed individually or with a partner! See the course honesty policy, below, for more details.

You are responsible for submitting your work to the autograder. Email submissions, whether late or on time, will be deleted without response, regardless of whether the autograder or Moodle appeared to be online or not. If either did truly go down for an extended period, I will find out by the next workday, and the entire class will get an extension.

You are responsible for verifying that you have submitted the intended versions of your files. Similarly, you are responsible for ensuring that the final submission you make in either Gradescope or Moodle is the one you wish to have graded. Requests to substitute another version may be granted entirely at the instructor's discretion, and may incur a heavy penalty.

You are responsible for submitting code that compiles and runs on the autograder; if you submit code that does not compile or that gets stuck in an infinite loop, you will receive no credit.

You are responsible for uploading your submission before the deadline. The deadline for an assignment is not the time by which you must finish writing a solution; rather, the deadline is the time by which you must successfully upload your solution files and confirm the system has recorded the correct versions of those files. We recommend that you upload your files at least one hour before the deadline, in case Gradescope or Moodle happens to lag or go down near the deadline. File system timestamps on your local hard drive or in a Github repository or the like are never acceptable as evidence of existence of a file prior to the deadline, because you are under complete control of that timestamp. Requests to submit after the late submission deadline will be handled as described in "Extensions and late policy."

Attempts to manipulate, game, or otherwise incorrectly use the autograder will be treated as academic dishonesty.

Problem sets

There will usually be a collection of short exercises ("problem sets") due each week. They will be generally posted to Moodle at the start of the week and will be submitted through either Moodle or Gradescope, as indicated.

You may work in a group with your classmates on problem sets, though you must clearly indicate in your submission with whom you worked.

Discussion exercises

There will be six or seven small group exercises given during discussion section meetings (over Zoom). These will be POGIL-style exercises, where each team member has a defined role. A series of questions will guide you through a cycle of exploration, concept invention, and application; your teaching assistant will facilitate the exercise. Groups that make a reasonable effort to complete the exercise will receive full credit.

Note that there are 13 scheduled discussion meetings, but only six or seven group exercises. On the other weeks you will engage in self-assessments.

Self-assessments and the final exam

There will be about seven self-assessments given over the course of the semester. They will be announced, and will be given in lieu of a discussion section meeting. Each will consist of a few programming questions, and they may include a few other short-answer questions as well.

Unlike other elements of the class, these **must be completed on your own**, without collaborating with your peers or performing Internet searches or the like. It's critical that you (and we) be able to evaluate your *own* learning progress.

There will also be a cumulative, online final exam, given on our scheduled exam day. It will be in the style of the self-assessments, but it will be significantly longer. **You must achieve a passing grade on the final exam to pass the class.** Like the self-assessments, the final exam is an individual assessment.

Please note (from the Academic Rules and Regulations):

> . . . it is University policy not to require students to take more than two final examinations in one day of the final examination period. If any student is scheduled to take three examinations on the same day, the faculty member running the chronologically middle examination is required to offer a make-up examination if the student notifies the instructor of the conflict at least two weeks prior to the time the examination is scheduled. The student must provide proof of the conflict. This may be obtained from the Registrar's Office, 213 Whitmore.

You are responsible for clearing your schedule at the beginning of the semester to take exams. The Registrar announces exam dates before the beginning of classes. If you cannot commit to taking the final exam on the scheduled date, you should drop this class immediately. Makeup exams will be offered only in those cases where required by university policy.

Extensions and late policy

Assignments are due at the time specified in Moodle or Gradescope, and we ask that you always try to submit before that deadline, even if your solution is

incomplete. That at least shows us that you are making progress.

But life happens: you miss the deadline because of other courses, illness, events in your personal life, and so forth. Rather than the burden of you having to ask for an extension in those circumstances, we will configure Moodle and Gradescope to accept late submissions, usually up to 48 hours after the deadline. There will be no penalty for these late submissions. They are an automatic extension you may choose to grant yourself. We trust you to do so wisely – often, making use of them will result in you having less time to work on the next assignment. The purpose of automatic extensions is to give you a tool to manage the demands of life, including the following:

- routine illness
- minor injury
- travel
- job fairs
- job interviews
- large workloads in other courses
- extra-curriculars

Be aware that after Moodle or Gradescope decide that the late submission deadline has passed – even one second past it – no further submissions will be accepted. **Do not email them to us or post them on CourseWire, we will not accept them.**

Beyond those automatic extensions, we will grant exceptional extensions in, well, truly exceptional circumstances. Those require accompanying documentation: a letter from a professor, advisor, or coach requesting the extension on your behalf; an obituary or wedding announcement published in a newspaper; a letter on official letterhead from a medical provider explaining a major illness or injury; or official DS letters. Contact the instructor to request such an exceptional extension.

A message from Student Success

Your success in this course is important. Your instructor has partnered with Student Success and your academic advisors to assist you in better understanding course material which can aid you on your path to success. Resources available to students include: **Supplemental Instruction, ExSEL Group Tutoring, and 1:1 Tutoring**. Visit the Learning Resource Center online at http://www.umass.edu/lrc.

Throughout the semester, your instructor will communicate with Student Success & academic advisors regarding your progress in the course. If you are contacted, please consider scheduling appointments such as tutoring or academic advising and talk with your professor. Referrals are not punitive and are meant to assist you in connecting with resources at UMass. Please email academicalert@umass.edu if you have any questions or need assistance connecting with resources.

Academic honesty

General academic honesty statement

Since the integrity of the academic enterprise of any institution of higher education requires honesty in scholarship and research, academic honesty is required of all students at the University of Massachusetts Amherst. Academic dishonesty is prohibited in all programs of the University. Academic dishonesty includes but is not limited to: cheating, fabrication, plagiarism, and facilitating dishonesty. Appropriate sanctions may be imposed on any student who has committed an act of academic dishonesty. Instructors should take reasonable steps to address academic misconduct. Any person who has reason to believe that a student has committed academic dishonesty should bring such information to the attention of the appropriate course instructor as soon as possible. Instances of academic dishonesty not related to a specific course should be brought to the attention of the appropriate department Head or Chair. Since students are expected to be familiar with this policy and the commonly accepted standards of academic integrity, ignorance of such standards is not normally sufficient evidence of lack of intent.

In addition, you should read the UMass Academic Honesty Policy (ignorance of the policy is no excuse).

Course-specific academic honesty policy

Academic dishonesty is usually the result of other problems in school. Please come see me or the TAs if you are unable to keep up with the work for any reason and we will do our best to work something out. I want to see you succeed, but I will not tolerate academic dishonesty.

Investigating academic dishonesty is an unpleasant experience for both the instructor and the student. Please help me by avoiding any questionable behavior.

Be aware that if something looks like academic dishonesty to us, we will treat it as such, unless you can provide strong evidence to the contrary. When in doubt, it is your responsibility to contact the course staff about whether a potential action would be considered academic dishonesty.

What is permitted and what is not? You may discuss material with others, but when collaboration is forbidden (specifically: on the self-assessments and the final exam), your submission (code and prose) must be entirely your own.

You may not get help with your work from anyone who is not a current COMPSCI 186 course staff member. "Help" includes designing algorithms, writing code, debugging, developing test cases, and so on.

You may not receive code from anyone except your partner(s) on a particular assignment, or the course staff.

You may not provide your solutions to others, either directly or via some sort of public posting, except when collaboration is explicitly permitted and when both you and the other person(s) are currently enrolled in this course.

You may not copy code from online sources – except for the current semester website. Copying and pasting code from another student or a third party is a violation of academic honesty, and we will endeavor to detect this by any means available to us, including automated similarity analysis of submitted assignments.

You may not use third-party online forums such as StackOverflow to ask for specific help on assignments, nor third-party course "notes" sites that traffic in solutions to assignments, nor may you search for solutions online.

When you ask for help, either in person or on CampusWire, it's good practice to ask your question by describing the problem you're having, or using a small synthetic example that illustrates your difficulty. If you must include a large chunk of your code to ask your question on CampusWire, mark it as a "private" question, and only the course staff will be able to see it.

Group work policy Programming assignments permit a single partner (a partner means one other person, besides yourself). Other work may permit groups of up to four people in total.

Having a partner is optional for programming assignments. A group is mandatory for discussion exercises, and optional for problem sets.

For discussion exercises, your partners must be from your discussion section.

For programming assignments and problem sets, your partner(s) may be from any discussion section. It does not have to be your own section.

You are free to work with different partners on each assignment. Especially for programming assignments, make sure to acknowledge any past partners whose work might have influenced your own in the Authors section.

If you work with partner(s), you must do all the work together. It is against the rules to split up the work, or to have one person do it and another person "check" it, or to have one person write the code and another person write the tests, etc. When you submit as a partnership or group, you are asserting that all the work was done together.

The above rule has implications for academic honesty policy violations. If one of you is guilty, all of you are guilty. Consider that carefully. If your partner went off and implemented a lot of code "on their own," how do you know they wrote it? Every year, it turns out somebody actually copied the code or solution from elsewhere. You will be culpable if your partner does this, because by submitting as a partnership (or group) you are claiming the solution to be jointly written by all of you.

Nonetheless, much like exceeding the speed limit on the highway, if you're going to violate the rules, there are norms. On the highway, that means speeding in the left lane, and sticking to the speed limit in the right lane. In this class, we likewise know some people will violate the rules and split the work anyway. So if you do that, make sure to detail in the Author section who did what work. That is your best defense should a violation of the honesty policy be detected.

Other academic regulations

The Office of the Registrar publishes Academic Regulations yearly. You should be familiar with them. Particularly relevant are the policies on attendance, absences due to religious observance, and examinations.

A word about putting your solutions on GitHub, GitLab, BitBucket, etc.

Per the course-specific academic honesty policy, you are not permitted to make your solutions to the assignments in this class available to others. This includes reposting them to public GitHub repositories (or other service where another student might plausibly see them).

A word about copyrights

Most of the material (lecture notes, lectures, assignments, and so on) in this course is original work created by the instructor (Marc Liberatore); exceptions are clearly noted. These works are protected by U.S. copyright laws and by university policy. I am the exclusive owner of the copyright in materials I create.

You may take notes and make copies of course materials for your own use in this class. You may also share those materials with another student who is registered and enrolled in this course.

You may NOT reproduce, distribute, upload, or display any lecture notes or recordings or course materials in any other way – whether or not a fee is charged – without my express written consent. If you do so, you may be subject to disciplinary action under the UMass Code of Student Conduct.

While you are welcome to use the material for your own personal and educational use, you may not redistribute them to others outside the class. In particular, selling or otherwise redistributing your notes (or mine!), making or selling audio, video, or still recordings of course material, is not allowed without express written permission from me.

I make this stuff available on the web for you to use easily and without the hassle of sign-ups, logins, and the like, not for you to abuse for a buck. As Carol Barr (Senior Vice Provost and Dean of Undergraduate Education) and Enku Gelaye (Vice Chancellor for Student Affairs and Campus Life) noted at the start of

the Fall 2018 semester, usage of notes or in-class recordings without the faculty member's permission is a violation of the faculty member's copyright protection.

This content is protected and may not be shared, uploaded or distributed.

Accommodation statement

The University of Massachusetts Amherst is committed to providing an equal educational opportunity for all students. If you have a documented physical, psychological, or learning disability on file with Disability Services (DS), you may be eligible for reasonable academic accommodations to help you succeed in this course. If you have a documented disability that requires an accommodation, please notify me within the first two weeks of the semester so that we may make appropriate arrangements.

Acknowledgments

Some material taken from the Rust Code of Conduct.

Some material taken from the Cornell CS 3110 syllabus and related policies.
