CS 758/858 Algorithms
Fall, 2023
The course website is `http://www.cs.unh.edu/~ruml/cs758`.

Overview

This class has several objectives. By the end of the class, you will

1. be familiar with the most common data structures and algorithms: understand how they work, recognize when to use them, and be able to design simple variations of them,
2. be able to implement non-trivial algorithms and data structures efficiently in C,
3. be able to analyze time and space complexity of algorithms,
4. recognize intractable problems and prove they are NP-complete,
5. be able to prove simple correctness properties, and
6. be able to complete nontrivial programming tasks under a deadline.

To accomplish these goals, we'll cover the 'greatest hits' of computer science: a selection of the most important fundamental concepts in the design and analysis of algorithms and data structures, including correctness, complexity analysis, and NP-completeness. In addition to implementing some of the classics, you will design some algorithms yourself. Note that, although this class involves writing some proofs, it also involves significant programming. You should already be a fluent programmer in a C-like language, be able to generate standalone command-line programs under Linux, understand some basic data structures (linked lists, binary trees, and hash tables), and have some basic experience writing proofs. We typically will not help students with programming-specific issues (with the possible exception of the first week).

Contact Info

	Wheeler Ruml	Devin Thomas
email:	`ruml@cs.unh.edu`	`dwt@cs.unh.edu`
office hours:	Kingsbury N215D	see `piazza`
	Tuesdays 11-noon	see `piazza`
phone:	2-2683	

Please come to our office hours! We're always interested in talking with students. If you'd like to talk to us but can't attend office hours, it's easy to set up a time — just email us or catch us after class. However, if you have a specific question about an assignment, please **do not** talk with us or email us — please instead post on the course forum at `piazza.com`! This ensures that you get the fastest possible response and that everyone in the class stays on the same page. Please don't post any code on `piazza` — the discussion should remain at the level of algorithms.

Required Texts

1. Cormen, Leiserson, Rivest, and Stein, *Introduction to Algorithms*, Fourth Edition, MIT Press, 2022.
2. Kernighan and Ritchie, *The C Programming Language*, Second Edition, Prentice Hall, 1989.

I think that these are both important books worth owning if you are serious about computer science. If you buy used books (I'm a fan of `used.addall.com`), please be sure to select the correct editions. Note that 'exercises' in CLRS appear at the end of each section (eg, exercise 7.2-1 is the first one at the end of section

2 in chapter 7) and 'problems' appear at the end of each chapter (eg, problem 7-2 is at the end of chapter 7).

I also recommend Kleinberg and Tardos, *Algorithm Design* (2005). It is not as complete as CLRS but it has great motivation, discussion, and examples. Bentley's *Programming Pearls* (Second Edition, 2000) has case studies of real applications. Garey and Johnson's *Computers and Intractability* (1979) is a readable introduction to NP-completeness. Books like Aziz et al's *Elements of Programming Interviews* (2012) have helpful practice problems.

Evaluation

Tentative weighting:

80%	14 weekly assignments. Some will have more programming, some will have more written work.
9%	midterm exam
11%	final exam

Each assignment has extra challenges for those students taking the graduate version of the course.

Attending lectures and recitations is recommended but not required. If you attend, please participate and don't sleep, text, or work on something else.

Depending on the number of students in the class, we reserve the right to grade only a selected subset of each assignment, which will be chosen just before grading begins and will be the same for all students. Clarity will be considered when grading your work—if it's hard for us to understand, it will be hard for us to give you credit.

Assignment deadlines are not flexible and there will be no credit for late work. This means that you can ask about solutions to homework problems immediately after the deadline. The schedule indicates when assignments are due so that you can plan your work in advance. If a severe illness or the death of a close family member interferes with your work for multiple days, please have the Dean of Students send me a letter and contact me as soon as possible (and certainly well before any assignment deadline). When work is handed back, please look it over promptly. You may ask for something to be re-graded within 7 days of when it was handed back. Note that the new grade will take effect whether it is higher or lower.

I use a mapping from scores to letter grades developed over the previous years that this class has been offered. You will receive occasional email with your current tentative final grade. However, the course changes every year and I reserve the right to change the mapping at any time.

General policies:

Any work you turn in must be yours. The purpose of the assignments is for you to think about the material yourself. Sometimes this will require more than an hour or two of hard thinking, even for a single problem. You may need to come back to a problem several times or try out several possible approaches before inspiration strikes. This is an important part of the learning process for this class. Talking with others about the class concepts and high-level algorithms is fine; talking about how to solve specific homework problems or talking at the level of code is not. Any collaboration must be cleared with me in advance and you must cite all sources you use in preparing your work, other than the textbook itself, my lectures, and the recitations. Using AI coding assistants (e.g., GitHub CoPilot) is not allowed (just as it would be in a coding interview). You can read additional articles about the algorithms we discuss in class (for example, on Wikipedia) but you should not look at any code (including, for example, any example code embedded in Wikipedia articles). We may ask you to explain your code in detail at any time. The course material is very popular and there are solutions to the problems available on the open Internet, including on the textbook website linked from the course homepage. Do not look at them. If you do, not only are you defeating the learning process and wasting your tuition money, but if your cheating is detected, you will receive a failing grade in the class. For some students, failing the class can have additional repercussions, including loss of TAships or dismissal from the University. We reserve the right to hold unannounced quizzes or increase the weight of the written exams at any time.

If a religious commitment will interfere with your completing assignments on time, please let me know at the start of the semester so that I can provide proper accommodation. If you have a disability that requires accommodations, please have Student Accessibility Services (SAS) contact me right away so that I can provide proper accommodation. After they have done so, please arrange a meeting with me so that I can be sure your needs are met. If you find emotional or mental health issues interfering with your performance, contact Psychological and Counseling Services (`www.unh.edu/pacs`). For help with time management and study skills, contact the Center for Academic Resources (`www.unh.edu/cfar`). If you need to isolate due to serious illness, please let me know as soon as possible when you expect to be able to return to class.

Programming assignments:

All programming assignments will be in C and we will provide some scaffolding code and sample inputs to get you started. Just as in the real world, some of this pre-existing code is not great and might require some time to understand. You are not required to use it, but it is probably worth the trouble. Your code must compile and run on `agate.cs.unh.edu`. It should compile without errors or warnings with the `-ansi`, `-pedantic`, `-Wall`, and `-Werror` flags to `gcc` (unless you are using a dialect like C99 in which case substitute something like `-std=c99` for `-ansi`). You will probably want to use tools like `valgrind` to catch bugs.

Written exams:

No notes or books will be allowed. The final exam will be cumulative, but with emphasis on the second half of the course. Please be sure to attend the exams, as they are only given once.