CPSC S365: Algorithms Preliminary Syllabus

Instructor: Dylan McKay

Summer 2025

1 What is this document?

This document details some important syllabus items for the course CPSC S365. It is meant to provide enough information for students to make a decision as to whether or not they would like to take the course, and it is meant to help the home institutions of these students determine credit for the course. A complete syllabus will be provided on the first day of class.

2 Course Overview

In this course, we will study the design and analysis of algorithms. We will learn many important algorithms and develop an understanding of how we can design our own algorithms.

This is a theory course, and we will not be coding for any of our assignments. Instead, we will practice describing algorithms at a high level as well as writing pseudocode, and we will support our algorithms with proofs. Mathematics and proofs are in some sense the language of this course, and students will be expected to do a great deal of proof writing.

2.1 Why Algorithms?

Here are some reasons why you'll benefit from taking a class on Algorithms! You'll...

...learn about and practice thinking about computational problems abstractly.
...learn to recognize variety of tractable and intractable problems.
...learn cool algorithms and techniques for making your own.
...learn many of the important ingredients for crushing technical interviews.

When you leave this class, you should expect to:

...know a variety of algorithms and how to apply them.
...be more capable of thinking about computational problems abstractly.
...be more capable of seeing problems as other problems.
...know a variety of techniques for designing algorithms.
...know how to prove correctness and analyze running times for simple and some advanced algorithms.

2.2 Course Topics

- Basic algorithm analysis
- Divide and Conquer
- Greedy Algorithms

- Dynamic Programming
- Graph Algorithms
- Flow Networks
- Reductions
- The P vs NP question and NP-Completeness

We also tend to spend time on a few data structures. Usually these will come up as part of algorithms which us them, and we tend to have a lecture on the nuances of Hash Tables. If we find extra time in the schedule, we will use that time for a unit on Randomized Algorithms.

2.3 Course Format

This Summer, we will meet 5 days/week for one 75-minute lecture each day, and we will additionally meet for a 50-minute discussion session on two of those days. This is a new format for the course, and I am still working out exactly how this time will be spent. Attendance is required for all lectures and discussion sessions.

2.4 Assessments

We will have a variety of assessment types in this course.

There will be homework, which is intended to be *formative*. That is, the homework is meant to be challenging and to facilitate learning the material. Homework itself will include open-ended problem sets in which students will design and analyze algoriths. Nearly every problem in these problem sets will require providing one or more proofs. There will also be electronic assessments completed online. These will sometimes be used to have students read about something we did not have time for in class and to think through them in an interactive way. Other times, they will be a medium to think about problems in an environment where the students can get instant feedback on their answers. These will allow unlimited submissions up to their given deadlines.
Then there will be in-class assessments, which are intended to be *summative*. That is, these are meant to be a measure of what students have learned. These will take place in the form of smaller assessments (quizzes) and longer assessments (exams). I have not yet figured out exactly how many of each there will be.

2.5 Required Materials

Students will need access to a computer in order to submit their work and complete our electronic assessments. There is no strict requirement for books for this coure. However, for students who find textbooks helpful, I strongly recommend having access to the following books:

- Algoriths Illuminaged (Omnibus Edition) by Tim Roughgarden.
- Algorithm Design (any edition) by Kleinberg and Tardos (KT)
- Introduction to Algorithms (any edition) by Cormen, Leiserson, Rivest, and Stein.

I will list recommended readings from each book, and they will mostly be redundaent with each other. Intoduction to Algoriths is the most comprehensive of the books, but Algoriths Illuminated and Algorith Desigh line up a bit more directly with how I teach the course.

2.6 Grading

I have not determined the weight of each tyre of assessment. However, the greatest emphasis will be on the in-class assessments, as these are meant to be a measure of the skills the students have develored. Homework will still be important, and a grade will still be given for homework. But because I want students to be comfortable making mistakes on the homework, and because cheating using LLMs is a real concern, I will not be giving a great deal of weight to the homework. I will also be giving a grade for participation in both class and discussion sessions.

2.7 Prerequisites

Students are expected to have taken a programming course in Data Structures and Algorithms, or to have equivalent experience. For I will expect that every student is comfortable with arrays, ArrayLists, LinkedLists, Stacks, Queues, and Binary Search Trees. I will expect students to have seen and be comfortable with the goals of Heaps, Hash Tables, and at least on type of balanced binary search tree, sucn as AVL Trees or Red-Black Trees. I will also expect students to have seen several algorithms such as Linear Search, Binary Search, Selection Sort, Insertion Sort, Bubble Sort, Merge Sort, Quick Sort, Breadth-First Search, Depth-First Search, and Dijkstra's Algorithm. I will expect a basic level of comfort in analyzing runtimes for simple algoriths (primarily those made up of simple loops) using Big-O notation. For Yale students, this would CPSC 223.

Students are also to expected to have taken an introductory course in Discete Mathematics and writing proofs, or to have equivalent experience. I will expect that every student is comfortable with basic set theory, basic logic, and basic proof techniques such as direct proof, proof by contrapositive, proof by contradiction, and proof by induction. Proof by strong induction is a particularly important tool for this course. I will expect students to be comfortable with the basics of Graph Theory, Combinatorics (counting, such as permutations and combinations), Probability, and Modular Arithmetic. For Yale students, this would CPSC 202 or MATH 244.

3 Contact Information

Please feel free to contact me with any questions about the course. My email address is d.mckay@yale.edu.