

Midterm Review

CS/MCS 401: Computer Algorithms I

Lecture 13

Professor: Jonathan Liu

Icebreaker: Provide some feedback for me!

Announcements

The midterm is Thursday! **Please try to arrive early: you will have the entire course period to work on it, and no more than that.**

Mock midterm has been released online.

You are allowed one hand-written cheat sheet. Please bring that, and a pencil (with eraser), but nothing else.

What should you know for the
midterm?



Unit 1: Algorithm Analysis

Algorithm Modeling:

- Given a real-world problem, how should we define an input and output?
- What are the real-world implications and shortcomings of this model?

Basic Principles for the course:

- Big-O Notation
- Runtime analysis
- Writing proofs of correctness about algorithms

Things you won't be tested on:

- Gale-Shapley Algorithm
- Pagerank Algorithm

Unit 2: Divide and Conquer

Algorithm Structure and Design: Base Case, Divide, Conquer, Combine steps

Runtime Analysis:

- Write a runtime recurrence from pseudocode
- Use Master Theorem to determine the runtime from a runtime recurrence

Proof of Correctness: Induction

Key Algorithms: Mergesort, Binary Search

Things you won't be tested on:

- Using induction or recurrence trees to analyze runtimes
- Karatsuba Multiplication, Closest Points

Unit 3: Graph Algorithms

Algorithm Structure and Design: Graph modeling, Alter a graph (or algorithm), Run a known graph algorithm

Runtime Analysis: Runtime of known algorithm + Runtime of modifications

Proof of Correctness: Direct Proof, using known facts about known algorithms

Key Algorithms: DFS + Topological Sort + Kosaraju, BFS + Bipartition, Dijkstra's

Things you won't be tested on:

- Graph algorithm data structures

What should you know about the exam format?



Two Things to Know

1. I can not state enough how similar the mock exam is to the actual midterm.
2. This is a 75-minute exam. For that reason:
 - a. You will not be expected to write any formal proofs.
 - i. However, you will be asked to justify the correctness and runtime of algorithms. These justifications should have the important content that any formal proof would require. Make sure you understand what claim needs to be proven, and what should be used to prove it.
 - ii. I will be more lenient with structure, but not with correctness. Don't stress over whether your code or explanation is properly formatted, stress over whether it's correct.

The rest of this time is your time!

