CSE 373: Data Structures & Algorithms
Course Victory Lap

Riley Porter

Winter 2017

Course Logistics

- HW4 grades out tonight.
- HW6 dropbox open now, sorry for the delay. Remember about the course message board!
- Course evaluations open. Please do them!
- Still waiting on a room for the review session, info will be posted on the announcements section of the website when we know

Today

- Rest-of-course logistics: exam, etc.
- Victory lap: review of main course themes
- What comes next?

Exam Policies

- Bring your Husky Card / Student ID
- Closed book / Closed notes / No calculators or other electronic devices
- **Begins promptly at 2:30 and ends promptly at 4:20 on Tuesday, March 14th**
- If an emergency happens, contact Riley as soon as possible. You must receive permission from Riley before the start of the exam in order for the option of an alternative.
- Any open exams (writing, reading, erasing, etc) before the exam starts or after the time is called is a **10 point deduction.**

Fair Exam Topics

- Union-Find
- Graphs
- Traversals, shortest cost path, Topological sort
- Minimum Spanning Trees
- Sorting
- Writing Pseudocode
- Analysis of runtime of pseudocode and ADT operations
- Any ADT we covered
- Preserving abstraction of client vs implementor: deep copy, immutability, copy-in and copy-out
- Anything from the homework assignments.

Not Fair Exam Topics

- Proof of lower bound for comparison sorting
- Proof of average case runtime for quick sort
- Proof by Induction
- Recurrence Relations
- BST or AVL deletion
- Calculating number of nodes in a tree (total number or number at a level).
- Skew Heaps
- B Trees
- Memory and Locality
- Concurrency or Parallelism
- Complexity Theory
- Mini-Max Search
- Testing theory or JUnit
- Technical Interviews

Victory Lap

A victory lap is an extra lap around the track by the exhausted victors (that's us) ☺

Review course goals

- Slides from Lecture 1
- What makes CSE 373 special

Big Thanks to your TAs!

Zelina Chen
Paul Curry
Josh Curtis
Chloe Lathe
Trung Ly
Matthew Rockett
Kyle Thayer
Raquel Van Hofwegen
Pascale Wallace Patterson
Rebecca Yuen
Hunter Zahn

Also... Thank you!

And huge thank you to all of **you**

- Great attitude
- Awesome questions
- Hard work (this is not an easy class!)

There are some things I would do differently the second time around, but as a whole, it has been great! I'm sad it's over ☹

- I welcome feedback, please use the course evals!

Where did we come from?

Now, three **completely un-edited** slides from our very first lecture!

- Hopefully they make more sense now
- Hopefully we succeeded

Slide 1: Topics Outline

- Introduction to Algorithm Analysis
- Lists, Stacks, Queues
- Trees, Hashing, Dictionaries
- Heaps, Priority Queues
- Sorting
- Disjoint Sets
- Graph Algorithms
- *May have time for other brief exposure to topics, maybe parallelism, technical interview questions, dynamic programming*

Slide 2: What 373 is about

- Deeply understand the basic structures used in all software
  - Understand the data structures and their trade-offs
  - Rigorously analyze the algorithms that use them (math!)
  - Learn how to pick "the right thing for the job"
  - More thorough and rigorous take on topics introduced in CSE143 (plus more new topics)

- Practice design, analysis, and implementation
  - The elegant interplay of "theory" and "engineering" at the core of computer science

- More programming experience (as a way to learn)

Slide 3: Goals

- Be able to make good design choices as a developer, project manager, etc.
  - Reason in terms of the general abstractions that come up in all non-trivial software (and many non-software) systems
- Be able to justify and communicate your design decisions

Dan Grossman's take:

- Key abstractions used almost every day in just about anything related to computing and software
- It is a vocabulary you are likely to internalize permanently

Success?

Were there other things we should have covered?

- Again, feedback on the course evals welcome!
- Also, extra extra topic resources posted

Hopefully, this just got you excited for more, and CSE 373 will not be your last exposure to computer science. **Where do you go next?**

CSE Non-Majors Courses

CSE 374 (~CSE 333): Intermediate Programming Concepts & Tools

Concepts of lower-level programming (C/C++) and explicit memory management. Software design, implementation, and testing strategies

CSE 410 (~CSE 351): Computer Systems

Machine organization, including central processor and input-output architectures; assembly language programming; operating systems, including process, storage, and file management

CSE 413 (~CSE 341): Programming Languages

Basic concepts and implementation strategies for modern functional and object-oriented programming languages such as Scheme and Java.

CSE 414 (~CSE 344): Introduction to Databases

Writing applications that use DBMS; data models, query languages, transactions, database tuning, data warehousing, and parallelism.

Moar CSE Non-Majors Courses

CSE 415 (~CSE 473): Introduction to Artificial Intelligence

Knowledge representation, logical and probabilistic reasoning, learning, language understanding, intro to game theory

CSE 416 (~CSE 446): Machine Learning

Provides practical introduction to machines learning. Modules include regression, classification, clustering, retrieval, recommender systems, and deep learning, with a focus on an intuitive understanding grounded in real-world applications. Offered jointly with STAT 416.

CSE 417 (~CSE 421): Algorithms & Computational Complexity

Complexity, P vs NP, undecidable problems, more graph theory, dynamic programming

Web Development

CSE 154: Introduction to Web Programming

Whitaker Brand teaching it next quarter and he is awesome. High level speed through of the technologies powering websites. Will not make you a web developer, but will teach you how to search Google and how to learn.

- HTML/CSS for content and styling
- JavaScript for client side programming
- AJAX for getting data and JSON and XML as data types
- PHP for server side programming
- SQL for querying a database

INFO 343 (client side) and INFO 344 (server side)

Dave Stearns is awesome and teaches both of these classes. Probably other people teach it too, but he's the one I know. Much more in-depth than CSE 154, and more likely to make you a "web developer". You could take CSE 154 first and then go more in-depth with Dave's classes.

Explore Web Libraries

After getting introduced to web technologies, you can dive deeper with powerful libraries and APIs.

Data Visualization JS library: https://d3js.org/

Angular JS (https://angularjs.org/)

React JS (https://facebook.github.io/react/)

HTML, CSS, and JS framework for managing interactive styles: http://getbootstrap.com/

Free UW Student Web Hosting: https://itconnect.uw.edu/connect/web-publishing/shared-hosting/

Other Departments!

Informatics

Careers as "business analysts, user experience designers, information architects, and product managers". Also can be a software engineer. Lots of interdisciplinary topics and classes to make you a well rounded applicant to the tech industry. New design class by Andy Ko in the Spring.

Human Centered Design and Engineering (HCDE)

"Students and faculty in HCDE design solutions to global challenges by tailoring technology to human needs and interests." Can also be a software engineer or product manager. HCI and user focused.

Applied and Computational Mathematical Sciences (ACMS)

"Designed for students interested in the application of mathematical and computational concepts and tools to problems in research or in the business world." Can also be a software engineer or a data scientist.

Other resources outside of UW

Coursera:

- Python: https://www.coursera.org/specializations/python
- Machine Learning: https://www.coursera.org/learn/machine-learning
- Computer Security: https://www.coursera.org/course/security
- Computational Neuroscience: https://www.coursera.org/course/compneuro
- Principles of Computing: https://www.coursera.org/learn/principles-of-computing-1

Code Academy: https://www.codecademy.com/

Khan Academy: https://www.khanacademy.org/computing

Treehouse: https://teamtreehouse.com/

Or take a boot camp!

Command Line and Source Control

If you've never used version control to manage a project, try it! Make a GitHub account, start a repository with just some text files, learn to edit, commit, branch, etc!

My favorite visual tool to learn git: http://learngitbranching.js.org/

Learn to navigate the command line with your terminal. On *nix, use the terminal and start immediately. On Windows: Cygwin, PuTTY, Indigo.

Learning: https://www.codecademy.com/learn/learn-the-command-line

Learn bash scripting. Some tutorials I found online:

- http://ryanstutorials.net/bash-scripting-tutorial/
- http://www.bash.academy/
- http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html

Learn a new Language!

- **Processing:** https://processing.org/
- **Haskell**: http://learnyouahaskell.com/chapters
- **Go:** https://golang.org/
- **C++**: http://www.learncpp.com/
- **Scala**: http://www.scala-lang.org/documentation/
- **Ruby**: https://www.codecademy.com/learn/ruby
- **PHP**: https://www.codecademy.com/learn/php
- **Racket**: https://learnxinyminutes.com/docs/racket/

There are resources of 100's of languages online. Pick one and mess with it!

Learn to code games and apps

- Using **Unity**: https://www.udemy.com/unitycourse/

- Using **ActionScript**: https://www.siteground.com/tutorials/actionscript/

- Make an **Android App** (using mostly Java):

http://developer.android.com/training/basics/firstapp/index.html

So much more!

- Contribute to open source projects
- Participate in a hackathon
- Learn how to write scripts to automate things you don't like spending time on!
- Create an account on StackOverflow
  - Ask and answer questions!
- Get a Raspberry Pi and start messing around with it: https://www.raspberrypi.org/blog/raspberry-pi-zero/
- Subscribe to the Programming subreddit or read hacker news (the people are only a little pretentious ☺)

Today's Takeaways

- We learned **a ton** in the past 10 weeks
- There is **a ton more** to learn and there are so many paths forward into the computer science field / software and tech industry / interdisciplinary fields
- Fill out the course evals!
- Thanks for the great quarter! Keep in touch!