CSC 212: Data Structures and Abstractions

Introduction

Marco Alvarez

Department of Computer Science and Statistics
University of Rhode Island

Fall 2020

Welcome !

- Lectures
  - TR 11 - 12:15p @ Zoom
- Labs
  - W 12 - 1:45p @ Zoom
  - F 10 - 11:45a @ Zoom
- Team
  - Marco Alvarez, **Instructor**
  - Christian Esteves, John Bertsch, Johann Muller, **TAs**
- Course Website
  - https://homepage.cs.uri.edu/~malvarez/courses/csc-212.html

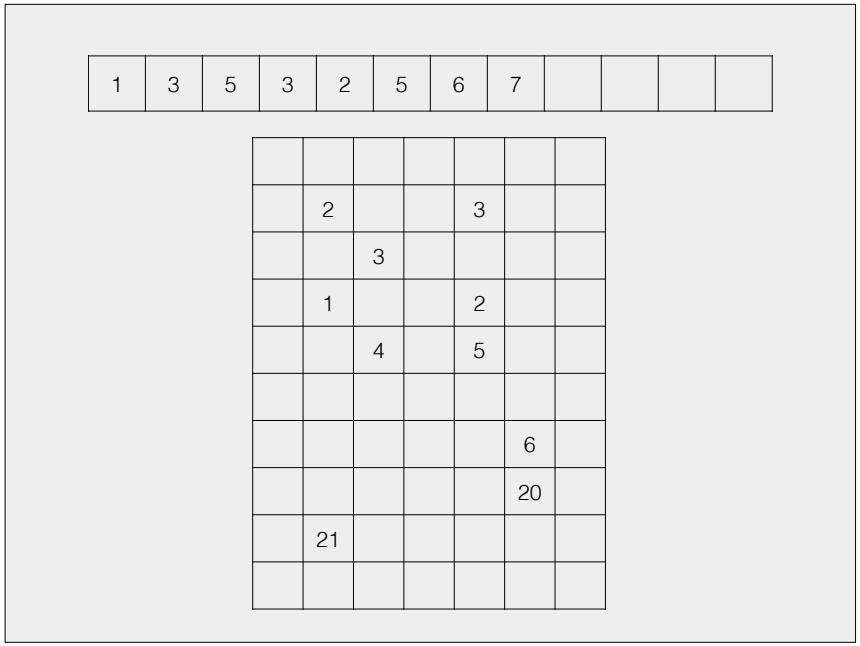
Typical Schedule

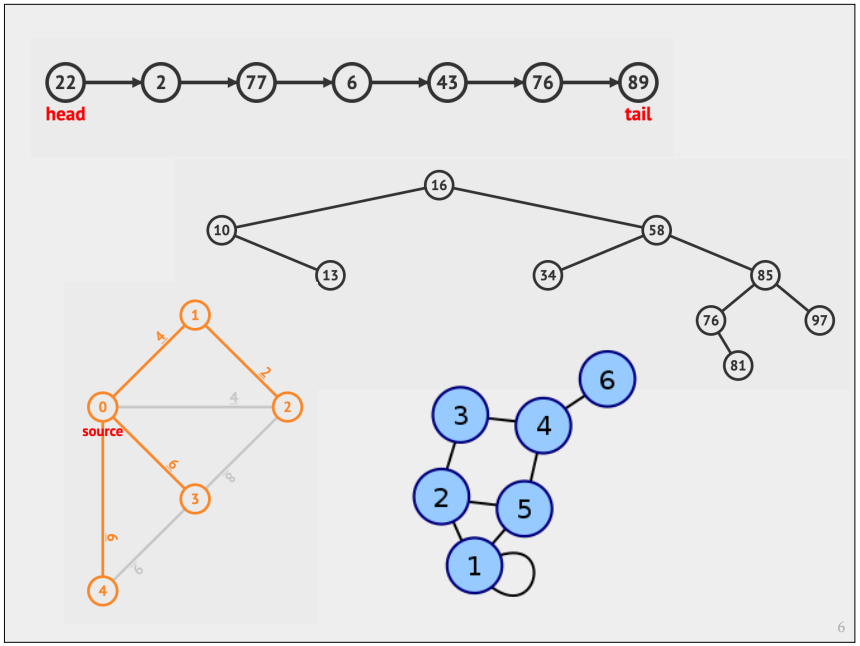
	Monday	Tuesday	Wednesday	Thursday	Friday
8a					
9a					
10a	Office Hours (Johann) 10 - 12p		Office Hours (Johann) 10 - 12p		Lab 2 10 - 11:45a
11a	Office Hours (Johann) 10 - 12p	Lecture 11 - 12:15p	Office Hours (Johann) 10 - 12p	Lecture 11 - 12:15p	Lab 2 10 - 11:45a
12p		Lecture 11 - 12:15p	Lab 1 12 - 1:45p	Lecture 11 - 12:15p	Office Hours (John) 12 - 1p
1p		Office Hours (Christian) 12:30 - 2:30p	Lab 1 12 - 1:45p	Office Hours (Christian) 12:30 - 2:30p	TA Meeting 1 - 2p
2p	Office Hours (John) 2 - 4p	Office Hours (Christian) 12:30 - 2:30p		Office Hours (Christian) 12:30 - 2:30p	
3p	Office Hours (John) 2 - 4p				Office Hours (Marco) 3 - 4p
4p					
5p					

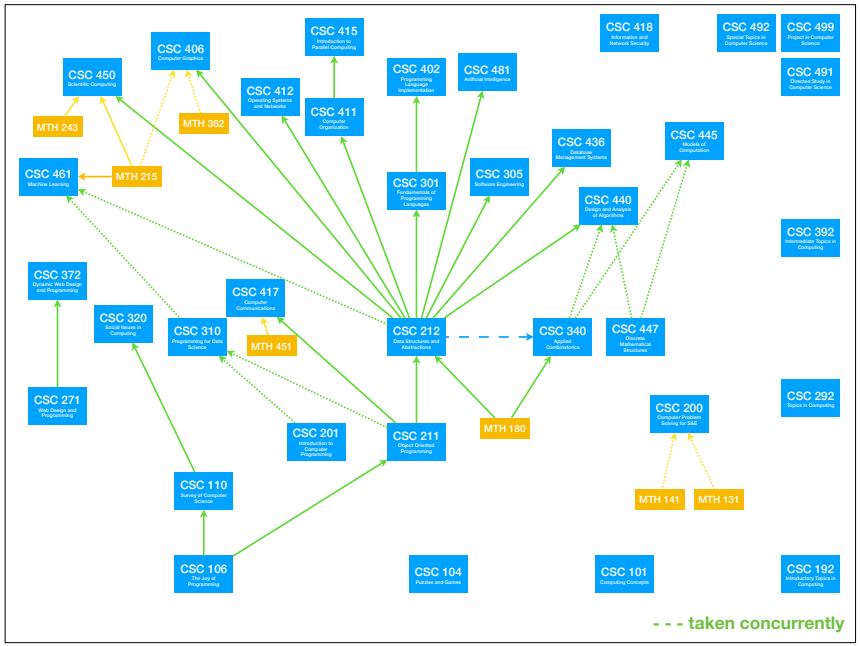
CSC 212?

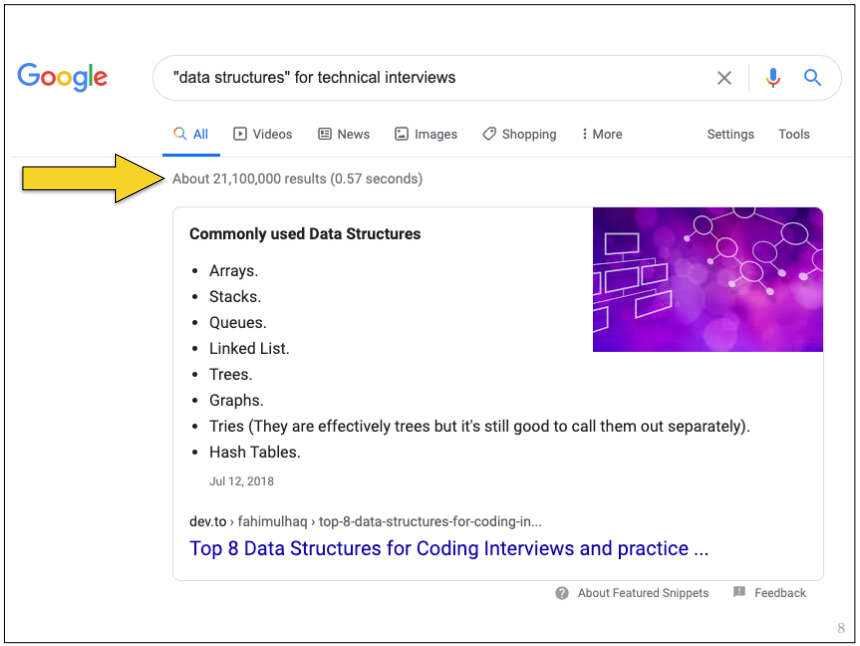
- Review of basic principles of analysis of algorithms
- Introduction to fundamental data structures and their algorithms
  - arrays, lists, stacks, queues, trees, hash tables, graphs
- Survey of classic algorithms for sorting and searching

Prerequisites: CSC 211 (at least C-) and MTH 180

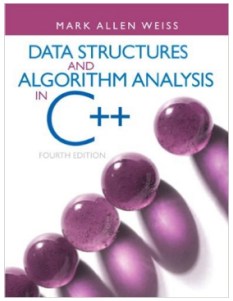

1 3 5 3 2 5 6 7
2 3
3
1 2
4 5
6
20
21

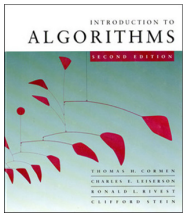

22 2 77 6 43 76 89
head
tail
16 10 58 13 34 85 76 97 81
1 4 2 0 4 2 source 6 3 8 9 6 4
6 3 4 2 5 1
6

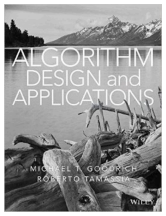

CSC 450
CSC 406
CSC 415
CSC 418
CSC 492
CSC 499
CSC 491
CSC 402
CSC 481
CSC 412
CSC 411
MTH 243
MTH 362
CSC 461
MTH 215
CSC 301
CSC 305
CSC 436
CSC 445
CSC 440
CSC 392
CSC 372
CSC 417
CSC 320
CSC 310
MTH 451
CSC 212
CSC 340
CSC 447
CSC 271
CSC 292
CSC 200
CSC 201
CSC 211
MTH 180
CSC 110
MTH 141
MTH 131
CSC 106
CSC 104
CSC 101
CSC 192
- - - taken concurrently


Google
"data structures" for technical interviews
All
Videos
News
Images
Shopping
More
Settings
Tools
About 21,100,000 results (0.57 seconds)
Commonly used Data Structures
Arrays.
Stacks.
Queues.
Linked List.
Trees.
Graphs.
Tries (They are effectively trees but it's still good to call them out separately).
Hash Tables.
Jul 12, 2018
dev.to › fahimulhaq › top-8-data-structures-for-coding-in...
Top 8 Data Structures for Coding Interviews and practice ...
About Featured Snippets
Feedback

Recommended Textbooks







OpenDSA

Need a refresher on C++ programming?

- Read a book
- Enroll in a MOOC (massive open online course)

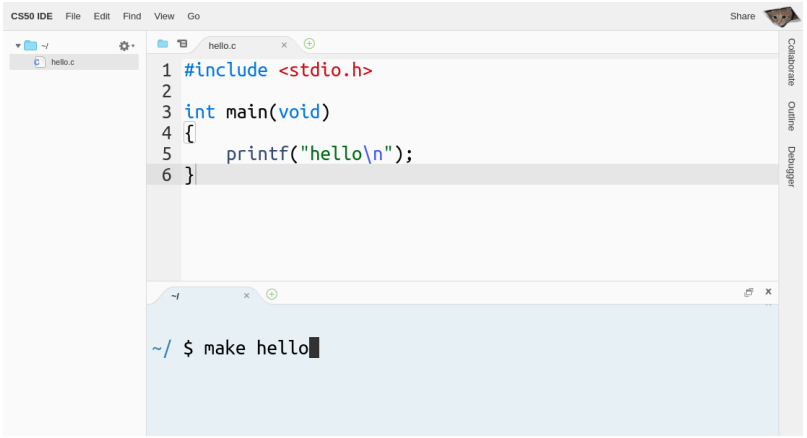
coursera edX

- Solve Challenges

Kattis

HackerRank
Revolutionizing Tech Recruiting

CS50 IDE



https://ide.cs50.io

Grading (subject to change)

- Lab attendance
  - synchronous labs (5%)
- Assignments
  - 7 programming assignments (25%)
  - 1 final project (20%)
- Exams
  - 1 midterm exam (25%)
  - 1 final exam (25%)

gradescope

All exams are based on lecture materials and assignments

Homework Assignments

- Discussions and collaboration are allowed, however you **must** write your own code and solutions
- All assignments are to be turned in on **Gradescope** by the due date
  - late submissions are **NOT** accepted

`Plagiarism?`

- just **don't do it**
- if you get caught (chances are very high), your name(s) will be immediately reported for further sanctions

What is expected from you?

- **Attend** **synchronous** lectures/labs
  - I do not spend time taking attendance … but … students skipping lectures will (**very**) likely **fail** this class
- **Organize** your time
  - lectures, labs, homework assignments, project, exams
- **Participate** and think critically
  - ask questions (lectures, labs, office hours, Piazza, …)
- **Start** working on assignments **early**
  - avoid copying/pasting or google'ing answers

Need help?

- Post questions on **Piazza**
  - answer questions, share information

piazza

- Contact your TAs
- Come to **Office Hours**

Programming Assignment 1

Warming up

- Adjacent elements sum
  - find the **maximum sum** of any pair of adjacent elements in an array of integers