CSE 373

DECEMBER 8TH – EXAM REVIEW

ASSORTED MINUTIAE

- **Course evaluations**
  - You thought I was kidding
  - Only 35% so far
  - I don’t WANT to have to email everyone
  - Leaving some time at end of class for evals, be prepared to have your computer out to pretend like you’re doing them

ASSORTED MINUTIAE

- **Course evaluations**
  - You thought I was kidding
  - Only 35% so far
  - I don't WANT to have to email everyone
- **Pre-final Grades**
  - Should be complete and to you by Sunday
- **Monday**
  - Office hours: 12:00 – 2:00; exam prep
  - After 2:00, prep + grade discussion (email)

TODAY'S LECTURE

- **Exam Review**
  - Important topics
  - Exam is comprehensive, but review will focus on the new material

EXAM FORMAT

- 1:50 to complete 10-12 problems
- First questions are short answer, which has many parts of varying difficulty, it is not likely to be the easiest
- Runtime and debugging questions
- Technical questions
- Algorithm Design question

EXAM FORMAT

- **We will be our most strict grading yet, don't make any assumptions that aren't explicit**
- **Analysis work needs to be thorough and concrete, recurrences and summations will be required**
- **Show all of your work. Many algorithms are trivial to solve by hand. Just providing "the solution" will not earn points. Algorithms are about process.**

EXAM FORMAT

- **This exam will feel shorter than the midterm, but a time crunch may be present**
  - There are many topics that need to be covered
  - Get down things that you know, and if you don’t make progress move on and come back

EXAM FORMAT

- **Short answer (3 problems)**
  - Graphs
  - Algorithms
  - Assorted
    - Design
    - Data Structures
    - Computability

EXAM FORMAT

- **Analysis (1-2 problems)**
  - Code
    - Runtime
    - Memory
  - Recurrences
  - Amortized analysis

EXAM FORMAT

- **Technical Portion (3-4 Problems)**
  - Quick sort
  - Merge sort
  - Heap Sort
  - Radix Sort
  - Prim's Algorithm
  - Kruskal's Algorithm
  - Union-find / Uptrees
  - 1st half of course?

EXAM FORMAT

- **Technical Portion (3-4 Problems)**
  - https://www.cs.usfca.edu/~galles/visualization/Algorithms.html
  - Good for practice and generating practice problems
  - Important to show all work
  - Practice makes perfect (and fast), the less time spent on these, the more time for the rest of the exam

EXAM FORMAT

- **Algorithm Design (Final Problem)**
  - Correctness
  - Runtime
  - Memory
  - Justification
- **Use any Data structure / Algorithm given in the course, but know its runtime**

EXAM FORMAT

- **Other (1-2 Problems)**
  - Debugging
  - Long answer
  - Problem reduction

TOPICS

- **Definitions**
  - ADT – Abstract Data Type – Describes a certain set of functionality and behavior
    - e.g. PriorityQueue
  - Data structure – Theoretical storage method that implements an ADT.
    - e.g. Heap
  - Implementation – Low-level design decisions that are often language dependent
    - e.g. Using an array for the heap

TOPICS

- **Stacks and Queues**
  - LIFO and FIFO ordered storage respectively
  - Can be implemented with arrays or linked lists
  - Understand the desired behavior and how to implement these structures

TOPICS

- **Algorithm analysis**
  - bigO, bigOmega, bigTheta
  - c and n_0
  - Asymptotic behavior
  - Memory analysis
  - Recurrences
  - Summations

TOPICS

- **Analysis**
  - Lower bound for comparison sorts
  - Memory usages for sorting
  - Best and worst case runtimes
  - Amortized Analysis
  - Recurrences

TOPICS

- **Testing**
  - White box v. Black box
  - Identifying edge cases
  - Difficulties and techniques
- **Debugging**
  - Programming process
  - Understanding code and potential problems

TOPICS

- **Memory**
  - Temporal and Spatial localities
  - Pages and their use
  - Tiered caching
  - Impact on cloud computing

TOPICS

- **Dictionary**
  - ADT- insert(k,v), find(k) delete(k)
  - Many possible underlying data structures
  - Different runtimes (and support)

TOPICS

- **Binary search trees**
  - Best and worst case
  - Traversals
- **Balance property – AVL**
  - Rotations and correctness

TOPICS

- **Hashtables**
  - Linear, quadratic, secondary hashing
  - Separate chaining
  - Load factor and resizing
  - Primary and Secondary clustering
  - Runtime and memory constraints

TOPICS

- **B-Trees**
  - Runtime and memory constraints
  - Insertion and deletion
  - Calculating M and L
  - Caching and the page size
  - Spatial and temporal locality

TOPICS

- **Priority Queues**
  - Insert(key, priority)
  - findMin()
  - deleteMin()
  - changePriority()

TOPICS

- **Heaps**
  - Usually array implementations
  - Heap property
  - Complete trees
  - Runtimes and buildHeap()

TOPICS

- **Sorting**
  - Insertion and Selection
  - Heap, Merge and Quick
  - Bucket and Radix
- **Properties**
  - Comparison sorts
  - Stable
  - In place
  - Interruptible (top k)

TOPICS

- **Graphs**
  - Notation G(V,E)
  - Traversals
  - Topological Sorts
  - Properties
    - Directed v. Undirected
    - Dense v. Sparse
    - Weighted v. Unweighted
    - Cyclic v. Acyclic

TOPICS

- **Graphs**
  - Algorithms
    - Dijkstra's – path finding
    - Prim's and Kruskal's – Minimum spanning trees
  - Know their runtimes and the data structures they rely on for those runtimes…

TOPICS

- **Graphs**
  - Problem symmetry
  - Min-cut and Max-flow
  - Candidate paths and Ford-Fulkerson

TOPICS

- **Union find**
  - ADT – Disjoint sets
  - Partitions
  - Weighted Union
  - Path compression
  - Uptree – single array representation

TOPICS

- **Algorithm Design**
  - How can you approach the problem?
    - Guess and check (Approximation)
    - Brute Force (Linear Work)
    - Divide and Conquer
    - *Greedy algorithms (make best decision for a local sub-problem)*
    - Randomization, Las Vegas and Monte Carlo
    - Preprocessing

TOPICS

- **Algorithm Design**
  - Find an approach to the problem that finds the solution
  - Understand what the edge cases are
  - Be able to analyze best-case, worst-case and memory usage of your algorithm
  - Randomization is okay if you can show it's faster than a more clever solution.

TOPICS

- **Computability and Complexity**
  - Computer science is based on the Turing Machine and the von Neumann architecture
  - Different Complexity classes,
    - P, NP, EXP
  - Some problems are unsolvable (HALT)
  - Possible to show problems are the same through reductions
    - CircuitSAT and 3-Color

STRATEGIES

- **Go through the exam from easiest to hardest**
  - Problems in the middle may be the easiest
- **Be as thorough as possible, if you think it's relevant and correct, include it**
- **Algorithm Design problem is as much about analysis as it is about clever solutions, so don't leave that done poorly**
- **Think about what things make certain algorithms tricky – highly likely for this final**

FINAL WORDS

- **Interview questions**
- **Studying from other Universities**
- **Other CS Non-majors courses**
  - CSE 374 – Software Development
  - CSE 413 – Programming Languages
  - CSE 414 – Databases
  - CSE 415 – Intro to AI
  - CSE 416 – Intro to Machine Learning
  - CSE 417 – Algorithms and C&C

FINAL WORDS

- **Great quarter!**
- **Stressful week**
  - Nothing feels better than walking out of class

FINAL WORDS

- **Great quarter!**
- **Stressful week**
  - Nothing feels better than walking out of class and… filling out course evaluations!

FINAL WORDS

- **Great quarter!**
- **Stressful week**
  - Nothing feels better than walking out of class and… filling out course evaluations!
- **Course Evaluations are due Sunday**
- **12:00 – 2:00 on Monday**
- **Have a nice break!**