Overarching topics
• proofs;
• induction;
• summations;
• $\mathcal{O}$, $\Omega$, $\Theta$, the complexity of an algorithm;
• reasoning about the behavior of algorithms.

Exam 1 topics
• stable sorting, the Gale-Shapley algorithm;
• divide and conquer techniques and algorithms;
• recurrences, recursion trees, the Master Theorem
• dynamic programming (recursive and iterative);
• optimal substructure and the Bellman equation;
• the recursion tree and the subproblem graph;
• complexity of a dynamic algorithm.

Exam 2 topics
• dynamic programming (recursive and iterative);
• optimal substructure and the Bellman equation;
• recursion tree and the subproblem graph;
• complexity of a dynamic algorithm;
• graphs, related definitions;
• BFS, DFS, Dijkstra’s algorithm, related proofs and techniques;
• data structures used in graph algorithms (adjacency list, matrix).

Exam 3 topics
• greedy algorithms;
• interval scheduling and variants;
• Huffman coding;
• $\mathcal{P}$, $\mathcal{NP}$, $\mathcal{NP}$-complete, $\mathcal{NP}$-hard;
• Karp/polynomial reducibility;
• showing a problem is $\mathcal{NP}$-hard, $\mathcal{NP}$-complete, etc;
• $\mathcal{NP}$-complete problems: 3-SAT, Hamiltonian path, Knapsack, etc.

Date: December 11, 2019.
Problems on previous exams sorted in ascending order by average score

- Exam 1 question 6;
- Exam 2 question 3;
- Exam 2 question 2;
- Exam 2 question 1;
- Exam 2 question 4;
- Exam 1 question 4;
- Exam 1 question 5;
- Exam 1 question 2;
- Exam 2 question 5;
- Exam 1 question 3;
- Exam 2 question 6;
- Exam 1 question 1.

What you should do

- read the book;
- read and understand all the homework solutions;
- read all the homework problems and make sure you can do them;
- read all the previous exam problems and make sure you can do them;
- think about the problems below.

For questions on Exam 1 and Exam 2 material, please refer to the respective review sheets posted on the course website.

1. (Prim’s algorithm) Given some graph with positive weights and a node, give a trace showing the execution of Prim’s algorithm.

2. (Kruskal’s algorithm) Given some graph with positive weights, give a trace showing the execution of Kruskal’s algorithm.

3. (MSTs) Given a graph with positive weights, \textit{not all distinct}, give a tight upper bound on the number of distinct minimal spanning trees.

4. (MSTs) Given a graph with positive weights, \textit{not all distinct}, compute every distinct minimal spanning trees without brute forcing it. Give an efficient algorithm for doing this.

5. (MSTs) Find a graph $G$ such that Prim’s algorithm and Kruskal’s algorithm both produce the same minimal spanning tree. Given a minimal spanning tree $T$ produced by Kruskal’s algorithm, is there a node $s$ for which $T$ is produced by Prim’s algorithm?

6. (MSTs) Prove or disprove: every minimal spanning graph is a tree.

7. (MSTs) Prove or disprove: if $T$ is a minimal spanning tree and $(x, y)$ is not an edge of $T$, then there is some edge $(u, v)$ of $T$ such that $w(u, v) < w(x, y)$. 
8. (interval scheduling) State the interval scheduling algorithm and use it to solve a given problem.

9. (interval scheduling) State the interval partitioning algorithm and use it to solve a given problem.

10. (Huffman) Given some alphabet and frequencies, produce the Huffman tree for that alphabet. Give the coding, and calculate the cost of the tree.

11. (Huffman) When does the Huffman coding produce character codes all of the same length?

12. (Huffman) Given an alphabet, what is the maximum depth a Huffman tree can have? Prove your answer is correct.

13. (Huffman) Given an alphabet, what is the minimum depth a Huffman tree can have?

14. (Huffman) Prove or disprove: in Huffman coding, each character has bit length $\leq$ its best fixed-length coding bit length.

15. (Huffman) Prove without using a Huffman tree: if $x, y \in \Sigma$ are minimal frequency, then there is an optimal prefix free coding in which they are siblings in the coding tree.

16. (Huffman) Prove or disprove: in Huffman coding, if characters $a, b, c \in \Sigma$ have $\nu(a) < \nu(b) < \nu(d)$ then $d$ must have a longer bit length than $a$.

17. ($\mathcal{P} / \mathcal{NP}$) Prove some problem class is in $\mathcal{P}$.

18. ($\mathcal{P} / \mathcal{NP}$) Given a certain kind of problem, prove that the decision version of the problem and the optimization version of the problem are $\approx_{\mathcal{P}}$.

19. ($\mathcal{P} / \mathcal{NP}$) Prove some problem class is in $\mathcal{NP}$ by showing that checking the answer is correct is in $\mathcal{P}$.

20. ($\mathcal{P} / \mathcal{NP}$) Prove some problem class is in $\mathcal{NP}$-hard (show that some $\mathcal{NP}$-complete problem is $\leq_{\mathcal{P}}$ it).

21. ($\mathcal{P} / \mathcal{NP}$) Prove some problem class is in $\mathcal{NP}$-complete (show that it is $\approx_{\mathcal{P}}$ some $\mathcal{NP}$-complete problem).