Topics covered on final:

1. Divide-and-Conquer:
   (a) Asymptotic complexity: $O$, $\Omega$, and $\Theta$.
   (b) Setting up runtime recurrences, and solving them using Master theorem.
   (c) Binary search, merge sort.
   (d) EXCLUDED:
       i. fast multiplication,
       ii. runtime recurrences not solvable using Master theorem,
       iii. other ways of solving runtime recurrences such as guess-and-check.

2. Graphs:
   (a) Bellman-Ford algorithm for computing shortest paths, and its termination conditions (either with distance, or existence of negative cycle).
   (b) Graph reachability (and how to solve it using simplified versions of the Bellman-Ford algorithm).
   (c) Definition of minimum spanning tree, and existence of good algorithms for it.
   (d) EXCLUDED:
       i. Shortest path algorithms faster than Bellman-Ford (e.g. BFS/ Dijkstra’s).
       ii. Extracting negative cycles from a non-terminating state of Bellman-Ford.
       iii. Details/correctness of Kruskal’s algorithm.
       iv. Cut/cycle property for minimum spanning trees.
       v. Directed acyclic graphs.

3. EXCLUDED: Dynamic Programming (this was extensively tested on Test 2).

4. Optimization
   (a) Definition of linear program, linear constraints.
   (b) Formulating linear / integer linear programs.
(c) Formulation of max-flow, residual graphs, and the Ford-Fulkerson algorithm.

(d) **EXCLUDED:**
   i. Standard forms of linear programs.
   ii. Solving linear programs.
   iii. Extracting minimum cut from maximum flows.
   iv. Proof / applications of max-flow/min-cut.

5. NP-hardness

   (a) Decision problems, conversion between decision and optimization problems.
   (b) Definition of $P$, $NP$.
   (c) Reductions, $NP$-hardness, and $NP$-completeness.
   (d) Definition of $3SAT$, and statement of $3SAT \rightarrow A$ for any problem $A$ that’s in $NP$.
   (e) **EXCLUDED:**
      i. Prior knowledge of which problems are NP-complete: The reduction problem will specify / define the problems to reduce from/to.
      ii. Reductions from 3SAT.
      iii. Reductions to Knapsack.

6. Approximation algorithms

   (a) Definition of approximation ratios of minimization/maximization problems.
   (b) Proofs of approximation ratios. For a maximization problem, upper bound OPT while lower bound the algorithm’s output, and vice versa for a minimization problem.

Practice questions:

1. Divide-and-conquer and Master Theorem
   (a) Homework 1 problems 2 and 3.
   (b) Test 1, problems 2, 3c, and 4.
   (c) Textbook Exercises 2.4, 2.5 (a) - (f).

2. Shortest paths and reachability
   (a) Test 2, problem 2a.
   (b) Textbook Exercises 3.7, 3.13, 3.17, 3.21, 3.22, 4.2, 4.11, 4.15 (but all in $O(n^2m)$ time), 4.20.

3. Linear programs and maximum flows:
(a) Homework 3, problems 1 and 3.
(b) Test 3, problem 3a.
(c) Textbook Exercises 7.2, 7.3, 7.4, 7.5, 7.8 (without the ‘solve the program’ parts), 7.17, 7.21.

4. NP-completeness proofs
   (a) Homework 4, problems 1 and $2a - 2c$.
   (b) Textbook Exercises 8.1, 8.2, 8.3, 8.4, 8.5. (Note: the textbook refers to Hamiltonian Tour as Rudrata Tour).

5. Approximation algorithms
   (a) Homework 4, Questions 2d and 3.
   (b) Textbook Exercises 9.6, 9.7, 9.8, 9.9.