Correspondence between Course Topics and CLRS References

[CLRS] denotes the main textbook by Cormen, Leiserson, Rivest and Stein

TOPICS:

1. Main topics in the course (informal discussion). Textbook references: Chapter 1 of [CLRS].

2. Notions related to problems, problem instances, running time and complexity.
   Definitions of order notations and examples: The Big-Oh notation
   Textbook references: Sections 2.2 and 3.1 of [CLRS].

3. Definitions of order notations and examples: Omega, Theta, little-o and little-omega definitions.
   Useful properties of order notation. Textbook references: Sections section 3.1, 3.2 of [CLRS].

4. Strategies for algorithm and loop analysis. Examples of loop analysis.
   Review of the Mergesort algorithm. Textbook references: Section 2.2 of [CLRS].

5. Analysis of Mergesort: Recurrence relation. Dealing with floors and ceilings.
   Guess and check method applied to mergesort: Proof that $T(n)=O(n\log n)$.
   Textbook references: [CLRS] section 2.3.

6. Multi-precision integer multiplication: the "grade-school" algorithm and a better D&C algorithm.
   The recursion tree method. Textbook references: Section 4.2 of [CLRS].

7. The Master Theorem: Statement and examples.
   More examples on the Guess and Check (or Substitution) method.
   Textbook references: [CLRS] sections 4.1 and 4.3.

8. Formal proof of the Master Theorem, when n is a power of b (floors and ceilings ignored).
   Textbook references: [CLRS] section 4.4 (section 4.4.2 is optional reading).

9. Bentley's problem: Three algorithms (emphasis on the Divide-and-Conquer algorithm). (Class notes only).

10. The closest pair problem and a divide-and-conquer approach.
    Textbook references: Section 33.4 in [CLRS].

11. A D&C algorithm for solving the selection problem in linear-time.
    Textbook references: Section 9.3 in CLRS (the recursion is slightly different than the one we used in class, though both correct).

12. Introduction to Greedy algorithms: framework and main properties.
    The coin-change problem, as a greedy algorithm.
    The activity selection (or interval scheduling) problem: greedy approaches which fail, and an optimal greedy algorithm. Strategies for showing optimality of greedy algorithms.
    Textbook references: Section 16.1 [CLRS] and first part of section 4.1 [CLRS] considers greedy algorithms as a special case of dynamic programming.

13. Proof of optimality of a greedy algorithm for the activity scheduling problem.

14. The Knapsack problem and its variants.
    Proof of optimality of the greedy algorithm for the fractional knapsack problem.
    Textbook references: last section of 16.2 in [CLRS].

15. The Task Scheduling problem.
    A greedy algorithm and its proof of optimality. Class notes suffice.

16. Proof of optimality of the greedy algorithm for task scheduling (continued).
    Introduction to Dynamic Programming: The "coins" problem revisited.
    Naive implementation of recursion takes exponential time.

17. A dynamic-programming approach for the coins problem.
    Main characteristics of dynamic programming: an overview.
    Textbook resources: [CLRS] Introduction of chapter 15.

18. The Longest Common Subsequence problem (LCS).
    An algorithm for LCS based on dynamic programming, and its analysis.
    Textbook resources: [CLRS] section 15.4.

    MIDTERM COVERS MATERIAL UP TO THIS POINT.

19. The 0/1 Knapsack problem, and a dynamic-programming algorithm.
    Textbook references: (0/1 Knapsack is problem 16-2-2 in [CLRS]).

20. Shortest-length triangulation of convex polygons: problem statement and an algorithm based on dynamic programming (only slides/class notes).

21. Graph Algorithms
    - definitions, terminology, properties.
    - data structures for representing graphs
    Textbook references: [CLRS section 22.1]

22. Graph search strategies - BFS/DFS
    Depth-first search (DFS)
    Identifying connected components - application of DFS.
    Textbook references: [CLRS] section 22.3 (not all the formal proofs are required material).

23. An application of DFS: Topological Sort.
    Textbook references: [CLRS] section 22.4.

24. Breadth-first search (BFS).
    Textbook references: [CLRS] section 22.2.

25. All-pairs shortest paths: The Floyd-Warshall algorithm.
    Textbook references: [CLRS] section 25.2.

26. Single-source shortest paths: Dijsktra's algorithm.
    Textbook references: [CLRS] section 24.3.

27. Complexity and proof of correctness of Dijsktra's algorithm.
    Kruskal's algorithm for MST's.
    Textbook references: [CLRS] section 23.2.

28. Proof of correctness of Kruskal's algorithm. Prim's algorithm for the Minimum Spanning Tree Problem. Textbook references: [CLRS] section 23.2.

29. Introduction to Computational Complexity.
    Textbook references: [CLRS] Introduction of chapter 34, section 34.2.

30. Definition of polynomial time reductions. Proof that HAMILTONIAN CIRCUIT reduces to TSP.
    Textbook references: [CLRS] Section 34.3.

31. Definition of 3SAT. Proof that 3SAT reduces to VERTEX COVER.
    Sources: Course Slides.

32. Definitions of NP-HARD, NP-COMPLETE problems. Verification algorithms and the class NP.
    Textbook references: [CLRS] sections 34.2, 34.3 relevant parts of section 34.4.

33. Seven known NP-COMPLETE problems.
    Proof of NP-Completes of SUBSET-SUM.
    Textbook references: [CLRS] 34.5.5.

34. Proof of NP-Completeness of KNAPSACK.
    Proof sketch of Cook's Theorem (SATISFIABILITY is NP-COMPLETE).
    (Course slides).

35. Undecidable problems. Proof that the halting problem is undecidable.