Theoretical Computer Science Syllabus

Sources:

[1] Christopher Moore and Stephan Mertens. *The Nature of Computation*.
https://nature-of-computation.org/ (official ECS 220 course textbook)
[2] Sanjiv Arora and Boaz Barak. *Computational Complexity: A Modern Approach*
https://theory.cs.princeton.edu/complexity/
[3] Christos Papadimitriou. *Computation Complexity*.
https://www.amazon.com/Computational-Complexity-Christos-H-Papadimitriou/dp/0201530821
[4] Cormen, Leiserson, Rivest, and Stein. *Introduction to Algorithms*.
https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/
[5] Kleinberg and Tardos. *Algorithm Design*.
https://www.pearson.com/en-us/subject-catalog/p/algorithm-design/P200000003259/9780137546350
[6] Donald Knuth. *The Art of Computer Programming*.
https://www.oreilly.com/search/?q=the%20art%20of%20computer%20programming&rows=100

1. Overview of Algorithm Design and Analysis
1.1. Design techniques [5, Ch 1] or [4, Ch 1]
1.2. Analysis techniques [5, Ch 2.1 – 2.4]

2. Basic data structures
2.1 Stack, queue [4, Ch 10]
2.2 Heap [5, Ch 2.5] or [4, Ch 6]
2.3 Balanced binary trees [4, Ch 12]

3. Greedy algorithms [4, Ch 4.1 – 4.6] or [5, Ch 15.1 – 15.2]

4. Dynamic Programming [5, Ch 6.1 – 6.6] or [4, Ch 14]

5. Graph Algorithms
5.1 Connectivity [4, Ch 20] or [5, Ch 3.1 – 3.4]
5.2 Single-source shortest paths [4, Ch 22] or [5, Ch 4.4]
5.3 All pairs shortest paths [4, Ch 23.1 – 23.2]

6. The Classes P and NP
6.1 NP-hard and NP-complete problems
6.2 Use of reductions
6.3 Dealing with NP-complete problems

7. Computability theory
7.1 Turing machines [1, Ch 7.1] or [2, Ch 1.2 and Ch 1.3]
7.2 Undecidability [1, Ch 7.2]

8. Space-bounded computation
8.1 L-completeness [1, Ch. 8.3]
8.2 NL-completeness [1, Ch. 8.3]
8.3 PSPACE-completeness [1, Ch 8.6]