CS 3353: Data Structures and Algorithm Analysis I   Spring 2021
Course Outline (Tentative)   (Last Revised on January 19, 2021)

---

1. ABET (Accreditation Board for Engineering and Technology, Inc.): Student Outcomes

1. Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.
2. Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.
6. Apply computer science theory and software development fundamentals to produce computing-based solutions. [CS]

2. ACM (Association for Computing Machinery): The Body of Knowledge

Knowledge Area	Total Hours of Coverage
Software Development Fundamentals (SDF)	2
Programming Languages (PL)	1
Algorithms and Complexity (AL)	34

3. Body of Knowledge Coverage

Knowledge Area	Knowledge Unit	Topics Covered	Hours
SDF	algorithms and design	concept and properties of algorithms, role of algorithms, problem-solving strategies, separation of behavior and implementation	1
SDF	fundamental data structures	stacks, queues, linked structures	1
PL	objected-oriented programming	objected-oriented design, encapsulation, iterators	1
AL	basic analysis	all core-tier 1: best-, expected-, and worst-case behaviors of an algorithm, asymptotic notions/notations (big-O), complexity classes such as constant/logarithmic/polynomial/exponential, empirical measurements of performance, time/space trade-offs in algorithms all core-tier 2: big-O/Omega/Theta and little-O/Omega notions/notations and asymptotic analysis, recurrence relations and some version of Master Theorem, analysis of iterative and recursive algorithms	6
AL	algorithmic strategies	core-tier 1: brute-force, greedy approach, recursive divide-and-conquer, dynamic programming core-tier 2: heuristics	6
AL	fundamental data structures and algorithms	all core-tier 1: simple numerical algorithms, binary search, sorting algorithms (selection, insertion, quicksort, heapsort, mergesort) and their worst- or average-case analyses, hashing, binary search trees and their common operations, graph-representations and simple graph-traversals core-tier 2: heaps, graph-algorithms: shortest-path algorithms (Dijkstra's and Floyd's algorithms), minimum spanning tree (Prim's and Kruskal's algorithms)	12
AL	basic automata computability and complexity	core-tier 2: introduction to the *P*, *NP*, and *NP*-complete classes	1.5
AL	advanced computational complexity	some classic *NP*-complete problems, reductions	1.5
AL	advanced automata theory and computability		0
AL	advanced data structures algorithms and analysis	balanced trees, B-trees, disjoint-sets, graph-algorithms: topological sort and strongly connected components	7

4. Course Outline (Tentative)

1. Mathematical Preliminaries and Introductory Material
   Source: Lecture Notes, and [CLRS09] Chapters 1, 2, 3, and 4
   1.1 Asymptotic Notations
   1.2 Recurrences

2. Sorting
   Source: Lecture Notes, and [CLRS09] Chapters 6, 7, and 8
   2.1 Heapsort, Mergesort, and Quicksort
   2.2 Lower Bounds for Sorting

3. Elementary Data Structures: Lists, Stacks, and Queues
   Source: Lecture Notes, and [CLRS09] Chapter 10

4. Hashing
   Source: Lecture Notes, and [CLRS09] Chapter 11
   4.1 Hash Functions
   4.2 Chaining
   4.3 Open Addressing

5. Trees, Tree Traversals, Binary Trees, and Search Trees
   Source: Lecture Notes, and [CLRS09] Chapter 12
   5.1 Trees and Their Implementations
   5.2 Binary Trees and Their Implementations
   5.3 Tree Traversals
   5.4 Binary Search Trees

6. Priority Queues and Their Implementations
   Source: Lecture Notes, and [CLRS09] Chapter 6

7. Advanced Design Techniques and Data Structures
   Source: Lecture Notes, and [CLRS09] Chapters 15, 16, and 18
   7.1 Dynamic Programming
   7.2 Greedy Algorithms
   7.3 B-Trees

8. Graph Algorithms
   Source: Lecture Notes, and [CLRS09] Chapters 22, 23, and 24
   8.1 Undirected and Directed Graphs, and Their Representations
   8.2 Some Fundamental Undirected and Directed Graph Algorithms
   8.3 Graph Traversals
   8.4 Minimum Spanning Trees
   8.5 Shortest-Path Problems