

## Schedule

(subject to change)

Date	Material
Tu 1/17	<b>Analyzing algorithms (Ch. 2.2)</b> Best case and worst case runtimes; insertion sort, incremental algorithm
Th 1/19	<b>Asymptotic notation (Ch. 3, Ch. A)</b> $O$ , $\Omega$ , $\Theta$ , $o$ , limit-theorem; runtime for code-snippets <i>Homework 1 assigned</i>
Tu 1/24	<b>Heapsort (Ch. 6)</b> Abstract data types (ADT), priority queue, heap, heapsort, linear-time buildheap
Th 1/26	<b>Divide-and-conquer (Ch. 2.3) and recurrences (Ch. 4.1, 4.2)</b> Divide-and-conquer, merge sort, binary search; Runtime recurrences. Solving recurrences with recursion tree; solving the recurrence with the substitution method (induction) <i>Homework 1 due; homework 2 assigned</i>
Tu 1/31	<b>Master theorem (Ch. 4.3), more divide-and-conquer (Ch. 31.6 pages 879–880; Ch. 30 pages 822–824; 28.2)</b> Use of master theorem to solve recurrences. Repeated squaring for exponentiation, Fibonacci numbers, polynomial multiplication, Strassen's matrix multiplication.
Th 2/2	<b>Randomized algorithms (Ch. 5.1–5.3), random variables and expected values (Ch. C.3)</b> Hiring problem; Expected runtime analysis. Random variables, expected value. <i>Homework 2 due; homework 3 assigned</i>
Tu 2/7	<b>Quicksort (Ch. 7.1–7.4)</b> Quicksort, best-case and worst-case runtimes, randomized quicksort.
Th 2/9	<b>Sorting (Ch. 8.1, 8.2, 8.3)</b> Decision trees, lower $\Omega(n \log n)$ bound for comparison sorts, counting sort, radix sort <i>Homework 3 due; homework 4 assigned</i>
Tu 2/14	<b>Order statistics (Ch. 9)</b> Order statistics (find $i$ -th smallest element); Randomized selection, deterministic selection in linear time
Th 2/16	<b>Hashing (Ch. 11; not 11.3.3 and not 11.5)</b> Direct-address tables, chaining, open addressing with linear probing, quadratic probing, double hashing. Hash functions <i>Homework 4 due; homework 5 assigned</i>
Tu 2/21	<b>Red-black trees (Ch. 13.1, 13.2, 13.3)</b> Red-black tree property, rotations, insertion; abstract data types, ADT dictionary
Th 2/23	<b>B-trees (Ch. 18.1, 18.2)</b> k-ary search trees, B-tree def., height, insertion <i>Homework 5 due</i>
Tu 2/28	<b>Test 1</b> Material until 2/16 (inclusive)
Th 3/2	<b>Augmenting Data Structures (Ch. 14)</b> Augmenting red-black trees; Dynamic order statistics, interval trees <i>Homework 5 due; homework 6 assigned</i>
Tu 3/7	<b>Range Trees</b> Range trees, in 2 dimensions and in $d$ dimensions; preprocessing time, query time.

Date	Material
Th 3/9	<b>Dynamic programming (Ch. 15.2, 15.3, 15.4)</b> Fibonacci, binomial coefficient, LCS: fill table, then construct solution from the table. <i>Homework 6 due; homework 7 assigned</i>
Tu 3/21	<b>Dynamic programming (Ch. 15.2, 15.3, 15.4)</b> Matrix chain multiplication; general outline of dynamic programming: Optimal substructure (recurrence), overlapping subproblems, fill table bottom-up or by memoization.
Th 3/23	<b>Greedy algorithms (Ch. 16.2 pages 380 middle – 384; problem 16-1 on page 402; Ch. 16.3)</b> Greedy algorithms (greedy-choice property, optimal substructure). Making change, fractional knapsack. Huffman codes <i>Homework 7 due; homework 8 assigned</i>
Tu 3/28	<b>Amortized analysis (Ch. 17.1, 17.2, 17.4)</b> Aggregate analysis (total runtime of $n$ operations), accounting method (prepay for later operations); binary counter, dynamic tables
Th 3/30	<b>Union-Find (Ch. 21.1, 21.2, 21.3)</b> Operations, list implementation, tree implementation, union-by-weight / union-by-rank, path compression. Ackermann function, and inverse Ackermann function $\alpha$ . <i>Homework 8 due; homework 9 assigned</i>
Tu 4/4	<b>Elementary Graph Algorithms (Ch. 22.1–22.4)</b> Representations of graphs, breadth-first search (BFS), depth-first search (DFS), topological sort
Th 4/6	<b>Minimum Spanning Trees (Ch. 23)</b> Prim (grows single tree), Kruskal (grows forest; uses union/find data structure) <i>Homework 9 due</i>
Tu 4/11	<b>Test 2</b> Material from 2/21 until 3/30 (inclusive)
Th 4/13	<b>Single-source shortest paths (Ch. 24 without 24.4)</b> Optimal substructure, triangle inequality, relaxation step; Dijkstra (only for non-negative edge weights), predecessor tree (shortest path tree); Bellman-Ford, detection of negative-weight cycles; Shortest paths in a DAG <i>Homework 10 assigned</i>
Tu 4/18	<b>All-Pairs Shortest Paths (Ch. 25.2)</b> Dynamic programming: Floyd-Warshall
Th 4/20	<b>P and NP (Ch. 34)</b> Decision problems, definition of classes P and NP, polynomial-time reductions <i>Homework 10 due</i>
Tu 4/25	<b>P and NP (Ch. 34)</b> NP-hardness, NP-completeness; Show that problems are NP-complete by reducing from other problems; TSP, Clique, Independent Set, Vertex Cover, Hamilton Path, Hamilton Circuit <i>Homework 11 assigned</i>
Th 4/27	<b>Maximum Flow (Ch. 26)</b> Flow networks; Max-flow min-cut, augmenting path, residual network
Tu 5/2	<b>Maximum Flow (Ch. 26)</b> Ford-Fulkerson, Edmonds-Karp <i>Homework 11 due</i>

The final exam will be on Saturday May 6th 8:00am – 10:45am.