

## COURSE SUMMARY

Below is a list of topics we covered this semester. All are fair game for the final exam.  
*Note:* the final exam is **open book**; you may bring any materials you want to the exam.

*Mathematical Preliminaries:*

- asymptotic notation ( , , , , )
- recurrence equations
- simple summations
- basic combinatorics
- basic probability (event spaces, expected values)

*Sorting:*

- Comparison based sorting
  - selection sort
  - insertion sort
  - bubble sort
  - merge sort
  - quick sort
  - heap sort
  - tree sort (insert elements into binary search tree, and extract them via in-order traversal.)
  - Stooge sort
  - decision tree model  $\Rightarrow O(n \lg(n))$  best worst-case time
- Linear sorting
  - counting sort
  - radix sort
  - bucket sort

*Order Statistics (find the kth smallest element of a collection)*

- naive method (first sort, then find kth element)
- quicksort-like partitioning
- median-of-median-of-c ( $c = 5 \Rightarrow$  CLR's `select` in Chapter 10.)

*Dynamic Set s (operations = search, insert, delete, minimum, maximum, predecessor, successor)*

- array implementations (sorted/unordered)
- list implementations (sorted/unordered; singly-linked/doubly-linked)
- binary search trees
- red-black trees
- augmenting red-black trees

*Dynamic Programming/Memoization*

- exponentiation
- fibonacci numbers
- Pascal's triangle
- longest common subsequence
- paragraph formatting

### *Greediness*

- knapsack problems (0/1, fractional)
- coin-changing (arbitrary denominations/restricted denominations)
- minimum spanning trees (Prim, Kruskal)
- single source shortest path (BST, Dijkstra)
- compression (Huffman coding)

### *Graphs*

- minimum spanning tree (Prim's algorithm, Kruskal's algorithm)
- single-source shortest path (breadth-first search, Dijkstra's algorithm)
- depth-first search
- topological sorting
- weakly/strongly connected components

### *Computational Complexity*

- problems and languages
- P, NP, and co-NP
- polynomial time reduction
- NP-completeness

### *Compression*

- simple methods
- Huffman coding
- substitution methods
- amount of information in file = length of shortest program that produces file.