

Final Review

1. REVIEW TOPICS

- Java basics
- Sorting and searching
 - linear and binary search
 - bubble sort, insertion sort, selection sort
- Linked lists
 - lists vs. arrays
 - operations on lists and analysis
 - singly LL, doubly LL, circular lists
- Program analysis
 - growth rate: big-Oh, big-Theta
 - finding the order of growth of an expression
 - analyzing running times of algorithms
 - comparing (running time of) algorithms
- Recursion
 - simple recursion examples
 - towers of Hanoi
 - blob counting, flow, maze
 - generating permutations, subsets, subset sum
- Stacks and queues
 - functionality
 - implementation with vectors and lists
- Searching with stacks and queues
 - the general framework
 - breadth-first search and depth-first search
 - trade-offs between DFS, BFS
 - examples: missionary cannibals puzzle, maze
- Maps and hashing
 - operations supported by a map
 - hashing and collisions with chaining, open addressing
 - load factor and performance
 - what is expected of a good hash function
- Graphs
 - terminology and basic properties
 - traversal: BFS, DFS
- Trees and binary search trees
 - definition and functionality
 - computing height, level, size
 - complete binary tree; number of nodes at each level, height
 - traversals: BFS, DFS, in-order, post-order, pre-order

- operations: search, insert, delete, min, max, successor, predecessor
- Priority queues and the binary heap
 - operations supported by a priority queue, and difference to a search tree
 - general idea of insert and extract-min and analysis
 - sorting with a priority queue

2. COURSE OUTCOMES

After this class you should be comfortable with the fundamental computer science algorithms and data structures, be able to use them to model and solve a problem, discuss their efficiency, be able to go from concepts to details, from theory to practice and implement a problem from scratch, and be able to debug your code.

More precisely,

- Know the fundamental data structures (arrays, vectors, lists, stacks, queues, trees, binary search trees, heaps, maps, hash tables) and basic algorithmic techniques (recursion; divide-and-conquer; backtracking, breadth- and depth-first search).
- Analyse the asymptotic performance of fundamental data structures and discuss which structure is better in what circumstances and what are the trade-offs.
- Be able to use the structures as black-boxes to solve a problem at a high level of abstraction.
- Be able to implement the details of a data structure.
- Be familiar with the general ideas for sorting (insertion sort, selection sort, bubble sort, merge-sort, heap sort)
- Know the major ways to implement searching (linear search, binary search, binary search trees, hashing)
- Be able to implement your code in Java, search the Java doc files, debug and get it to work.