Study Guide for Final Exam

CSE 30331/34331: Data Structures

The midterm exam will be on Thursday, Dec 15, 10:30am to 12:30pm in 102 DeBartolo. You only need a pen/pencil. Books, notes, smart phones, and computers must be closed. You may bring a four-function calculator. The exam is worth 60 points, or 20% of your grade. The exam is comprehensive; about 25% will cover material from before the midterm, and about 75% will cover material from after the midterm.

Structure

The exam will have three parts of roughly equal length and value:

- Factual questions. These will test your recall of material in the book, though they will not refer to any specific examples or implementation details in the book.
- Short answers. These require some thought but not much writing.
- Long answers. These may involve writing pseudocode or C++ code, simulating an algorithm or data structure, or discussing how an algorithm/data structure works or why it is good/bad.

Topics

Before midterm:

- Complexity analysis. Expect to be given a piece of C++ code and to figure out what the big-O time and/or space complexity are.
- Linked lists.
- Stacks, queues, and deques (implemented as arrays or as linked lists).
- Priority queues and binary heaps, including the handout on `make_heap` (excluding the proof).
- Sorting algorithms: insertion, selection, heap, quick, merge. You do not need to memorize all the quicksort partitioning schemes (Lomuto, Sedgewick, etc). If we ask you to implement quicksort, you can choose the partitioning scheme, or we will describe one to you.
- Binary search.
- Binary search trees (insertion and deletion).
- B-trees and red-black trees. We won't ask you to write C++ code for these, and we won't ask you about deletion.

After midterm:

- Hash tables. Both separate chaining and open addressing.
- Bucket sort.
- Graphs.
- Breadth-first search, depth-first search, and Dijkstra's algorithm.
- Bloom filters (excluding the proof of the optimal table size).
- Hadoop/MapReduce and distributed filesystems.

Sample Questions

All the sample questions from the midterm, plus:

HW4Q1. Time complexity of lookup, insertion, and deletion in a hash table.
HW4Q2. Working out a simple example of a hash table.
HW4Q3. Comparing separate chaining and open addressing.
HW4Q4. Properties of a good hash function.
HW4Q5. What hash functions are valid for bucket sort.
HW4Q8. Complexity analysis of bucket sort.
HW5Q1. Graph terminology.
HW5Q2. Graph representations.
HW5Q3. Properties of BFS.
HW5Q4. Uses of BFS and DFS.
HW5Q5. Details of Dijkstra's algorithm.
HW5Q6. Details of Dijkstra's algorithm.
HW5Q7. Details of Dijkstra's algorithm.
HW5Q8. Implementation of BFS or DFS.

Expect another high-level design question like: You want to develop a social networking application for people looking for fistfights. (Caution: Fighting can lead to injuries. Please don't fight.) It should keep track of who has fought whom, and the winner of each fight, and should be able to make suggestions for new people to fight. Describe, at a high level, the data structures and algorithms you would use to build such an app.

Bloom filters: What's a practical way of creating the *k* independent hash functions needed for a Bloom filter?

Distributed filesystems and MapReduce: Name one of the design principles of a GFS/MapReduce cluster.