Algorithms and Data Structures

Summer Semester 2022

Topics for the Final Exam, August 8, 2022

Topics in addition to the Mock Exam Topics in blue.

1. Basics of Big-Oh and Big-Omega notation: Definition (see, e.g., GTG Exercise R-3.8 for a precise understanding), relative ordering of growth rates (GTG Exercise R-3.8), spotting characterization of running time from simple pieces of code (GTG Exercises R-3.23–27). Many of the other R-Exercises from Chapter 3 are also worth looking at.

2. Recursion: The four basic examples from GTG Section 4.1 are worth reviewing in depth, including the analysis of run-time later in the chapter. To see further good examples for recursion, jump ahead to the recursive definition of a tree, the recursive formulation of tree traversal problems and the recursive computation of depth and height in trees, all in Chapter 8.

3. Turning recursive algorithms into non-recursive ones (GTG Section 4.6, Exercise C-4.9 – can you do it both ways?), or vice versa.

4. Dynamic arrays: implementation and run-time analysis, in particular amortization (see, e.g., GTG Exercise R-5.5), use of index arrays (this is a very wide field, but see, for example Exercise R-5.7 for an interesting twist).

5. Stacks, queues, and deques: know the basic operations on these abstract data types (see most of the R-Exercises from Chapter 6; can you implement a stack or a queue using the Priority Queue as a basis? – see Exercises C-9.26, C-9.27, and following).

6. Linked Lists: know the basis operations on singly and doubly linked lists (see, e.g., Exercises R-7.1, R-7.3, R-7.5), running time comparison of linked lists vs. arrays (Section 7.7)

7. Trees: major part of possible exam topics, in particular Sections 8.1, 8.2, 8.4.1–3. The chapter exercises contain a lot of good practice material, e.g. Exercise R-8.2, R-8.5, R-8.7, R-8.8, R-8.13, R-8.20.

8. Priority queues: definition of the abstract data type, priority queue implementations via an unsorted list, a sorted list, or a heap. Performance analysis of the heap data structure, using priority queues for sorting, in particular the heap sort? Practice problems are plenty in the Chapter 9 exercises, in particular R-9.2–12, R-9.18–20.

9. Maps and hash tables: definition of data type, expected running time of hash table access, collision avoidance. The details of particular hash functions or load factors will not be on the exam. Exam questions will be at the level of Exercise R-10.18 or similar.

10. Sorted maps and search trees: Skip lists (basic idea only – understand their properties relative to the tree-based concepts that follow), binary search trees (in particular implementation of the fundamental operations: walking the tree, searching, insertion, and deletion of items), AVL trees. Understand that there are other methods for balancing search trees, but these were not covered in detail and will not be on the exam. See, e.g., R-11.1–5, R-11.7, R-11.10, C-11.29, C-11.39. (Have a good look, in particular, at these last two exercises!)

11. Sorting: look at questions like Mock Exam Question 2, but also R-12.13, R-12.22, R-12.24, C-12.26.

12. Graphs: Know the basic definitions from Chapter 14.1. Graph Traversals: we have not covered them in any detail, but be aware of the problem statement and how a graph traversal is similar to a tree traversal (which we *have* covered in detail). Shortest path and Dijkstra's algorithm: there won't be a direct question on this topic, but there will be a (simpler) graph problem with a similar flavor. Thus, having a high-level idea of what is going on might be of some help.

General advice:

- Reading code fragments from the book is always a good idea. Translation from code into a more schematic description, or vice versa, is a possible exam problem. The general emphasis in this class is not on the fine details of providing an object oriented API in Python, but rather on the core structures of the algorithms.

- Review the coding exercises done in the tutorials. There might be questions about finding errors in code, or on any of the algorithmic patterns covered. Exercise R-10.21 might also be a good problem to look at.