CSE 100: Algorithm Design and Analysis. Final exam key topics.
Spring 2019

Note:

- The final will be a cumulative exam, covering Chapters 2, 3, 4, 6, 7, 8, 9, 11, 12, 15, 16, 22, 23, and 24.
- More than 70% of problems will be from Chapters 12, 22, 23, and 24.
- For the chapters covered by the previous exams, most problems will be easy to answer if you understood the contents rather than simply memorized. For example, some questions could be on the RT of some lines of a given pseudo-code.
- Warning: I do not guarantee that the following list is complete.

Implementing/correcting pseudo-codes:

- Ch 2: Mergesort
- Ch 7: Quicksort
- Ch 9: Randomized Selection
- Ch 15: Rod Cutting, Fibonacci, LCS
- Ch 16: Huffman
- Ch 12: BST (you can skip the delete operation)
- Ch 22: DFS and BFS.
- Ch 23: Kruskal's and Prim's algorithms.
- Ch 24: Bellman Ford and Dijkstra.

Ch 2. No need to study proving correctness using loop invariant or induction. Perhaps one of the chapters with the lowest priority.

Ch 3. O, Ω, Θ notations. Comparing functions asymptotically. Mostly T/F questions.

Ch 4. How to use the recursion tree method and Master's theorem (no need to study the substitution method) to solve a recurrence. High-level understanding of matrix multiplication algorithms (naive divide-and-conquer vs Strassen's).

Ch 6. Binary heap. You should know the RT of basic operations of binary heap and min/max priority queue.

Ch 7. Running time of deterministic and randomized quicksort.

Ch 8. Decision tree's implications to the running time of comparison based sorting algorithms. Stable sorting. When you use counting, radix, and bucket sorting.

In this class you learned Insertion sort, Merge sort, Quick sort, Heap sort, Counting sort, Radix sort, Bucket sort (and Topological sort). You should know their RT and their pros and cons. Like which algorithms are in-place sorting algorithms. Their average, worst case running time.

Ch 9. Only need to study the randomized Select algorithm. But you should remember that there exists a deterministic algorithm with RT $O(n)$ for the Selection problem.

Ch 11. Why do we use hashing? Applications of hashing. No need to study how to resolve collisions and what are good hash functions. The assumption of simple uniform hashing.

Ch 15. Only bottom-up. Fibonacci, Rod-cutting, and LCS. Describing a DP algorithm: define subproblems, recursion, and in which order to fill the DP table. Analysis of RT.

Ch 16. Greedy algorithms: Interval selection, Huffman code.

For Ch 12, 22, 23, and 24, see the work-out handouts.