

CS535: Algorithm Design and Analysis

Coverage for Midterms

- Framework for analyzing algorithms.
Why are we not measuring the runtimes of algorithms based on “clock time”? By saying that an algorithm runs in $O(n)$ time in the worst case, what do we really mean?
- Asymptotic notation.
Know how to order functions based on their growth rate; see exercise R-1.6 and section 1.2, p.20. What are the pitfalls of using asymptotic notation in describing the runtimes of algorithms?
- Math review.
You will not be asked to explicitly compute for logarithms or exponentials. However, the properties of these functions as well as sums of sequences will come in very handy when you compute for runtimes of algorithms.
- Analyze runtimes of algorithms.
Again, you will need to do this after you’ve designed your own algorithm.
- Stacks, queues, vectors, lists and sequences.
The basic methods for each ADT consist of searching, inserting or deleting an item. Know the different implementations for each ADT and how they affect the runtimes of the methods.
- Trees.
Know how trees are implemented and the different ways of traversing them.
- Priority queues and heaps.
Understand what heaps are and how they are used for implementing priority queues. When are priority queues used?
- Dictionary ADT and hashing.
How does hashing work? What are the different ways of resolving “collision”, and the advantages and disadvantages of these schemes?
- Binary Search Trees and its variants: AVL and splay trees.
Understand the BST and the different ways used to keep its height short. Know how the methods for searching, inserting and deleting an item are implemented. What are their runtimes?
- Designing algorithms.
Do review the designing algorithms questions in your homeworks. Try answering some of the problems in your book as well.