Design and Analysis of Algorithms - Midterm Overview

Analysis of Algorithms
- Definition of O, Θ, Ω **I won't ask you about little-oh, little-omega.**
- Show that *f(n)* is *O(g(n))*
- Express and justify the runtime of an algorithm in Big-Oh notation
- Amortized analysis

Elementary Data Structures
- For each data structure, know what **operations** can be performed on it and understand the **complexity** of those operations.
- Stack and Queue
    - implementations using array or linked list
    - resizing and amortized analysis for stack operations
- List, Vector, and Sequence
    - implementations using array or linked list
- Trees
    - traversals (preorder, postorder)
    - representations/implementation using linked structures
- Binary Trees
    - traversals (inorder, Euler tour)
    - representation/implementation using linked structures or an array
- Priority Queues
    - implementation using a sorted/unsorted sequence
    - sorting using a priority queue - selection sort, insertion sort
- Heaps
    - definition (heap order property: $key(v) \geq key(parent(v))$, and complete binary tree)
    - height
    - heap-sort (algorithm and analysis)
    - implementation using a vector
    - bottom-up heap construction (algorithm and analysis)
- Dictionaries (unordered - log-files and hash tables; ordered - lookup table and search trees)
    - log-file (unsorted sequence implementation)
    - hash tables
        - hash functions (hash code map, compression map)
        - insert/search/remove using chaining, linear probing, double hashing as collision handling strategies
        - performance in relation to load factor
        - definition of universal hashing
    - lookup table (sorted sequence implementation)
        - binary search

Search Trees
- Binary Search Trees
    - definition
    - operations and run-times: insert, find, remove
    - height
- Balanced Binary Search Trees
    - Red-Black trees
        - definition (BST, root property, external nodes property, internal nodes property, depth property)
        - height
        - operations and run-times: insert, find
        - run times: remove. **I won't ask you to demonstrate the removal of an item from a red-black tree.**

Sorting and Selection
- Merge Sort
    - divide and conquer technique
    - merging sorted lists
    - algorithm
    - analysis: $O(n \log n)$
- Quick Sort
    - pivot, partition
    - algorithm
    - analysis: $O(n^2)$ worst case, $O(n \log n)$ expected
- Comparison of sorting algorithms
- Lower bound on sorting
- Bucket and Radix Sort
    - bucket-sort algorithm (stable) and run-time
    - lexicographic sort
    - radix-sort algorithm and run-time
- Selection
    - problem formulation
    - quick-select algorithm
    - expected run-time
- Set data structure
    - implementation with sorted sequence
    - operations using generic merge
    - run-times