CSCI 310, Data Structures

Spring 2020 Final Exam Topics

1. Multi-dimensional arrays
2. Complexity Analysis
    a. Complexity – Time vs. Space
    b. Input size
3. Measuring running time
    a. Identify “Basic Operations” ≡ Constant-time operation
    b. “Count” the number of times the Basic Operations execute
        i. Don’t really “count” them
        ii. Develop a “complexity function” f : Input-Size → Real
        iii. f(n) is how many the Basic Operation executes for input size n
    c. Common “Basic Operations” (constant time operations)
        i. Assignment statement
        ii. Method calls
        iii. Arithmetic operations
        iv. Comparisons
        v. Array indexing
        vi. Following object reference
        vii. Returning from a method
    d. Asymptotic running time
        i. For Complexity function:
            1. Drop lower order terms
            2. Drop multiplicative constants
            3. Result is Θ
    e. Determine running time of code segments
4. Generic classes
5. Lists
    a. List ADT
    b. Array-based list
    c. Singly linked list
    d. Doubly linked lists
6. Circular buffers
    a. Circular buffer ADT
    b. Array-based circular buffer
    c. Linked circular buffer
7. Queues
    a. Queue ADT
    b. Array-based queue
    c. Linked queue

8. Stacks
    a. Stack ADT
    b. Array-based stack
    c. Linked stack
9. Sets
    a. Set ADT
10. Sparse Matrix
11. Binary Trees
    a. Binary Search Tree (BST)
        i. BST ADT
    b. AVL Trees
    c. Traversals of Binary Trees
        i. Left-Right-Root
        ii. Left-Root-Right
        iii. Root-Left-Right
12. Graph
    a. Graph ADT
    b. How to represent a graph in a class
    c. BFS/DFS
        i. When there is a choice, visit neighbors in order
        ii. Graph traversals using BFS and DFS
        iii. BFS/DFS order
        iv. BFS/DFS tree
        v. Connectivity
        vi. SSSP
        vii. BFS gives shortest paths
            1. Recovering shortest paths from BFS
    d. Topological Sort
        i. Directed Acyclic Graph (DAG)
    e. Minimum Spanning Trees (MST)
13. Hashing
    a. Hash functions
    b. Resolving collisions
14. Priority Queues
    a. Implement Priority Queue with an ordered list
    b. Implement Priority Queue with an unordered list
    c. Implement Priority Queue with a Heap
15. Heaps
    a. Comparators
    b. Structure Property – complete tree
    c. Heap Order Property – each node has lower priority than its parent
    d. Adding to heap – percolate up
    e. Removing from heap – percolate down
    f. Heap ADT

16. Sorting
    a. Bubble Sort
    b. Selection Sort
    c. Insertion Sort
    d. MergeSort
    e. QuickSort
    f. Radix Sort