Name: Hung Le

Course: CSCI 2125 – Data Structures

Professor: Dr. Minhaz Zibran

Reading Guide for Final Exam

I. Algorithm Analysis

a) Materials
- ♦ Textbook sections: 2.3, 2.4
- ♦ Moodle material: Algorithm Analysis Tips/Rules
- ♦ Quiz Solution: 1

b) Question patterns
- ♦ Given a code snippet, compute its big-oh complexity
- ♦ Definition and Distinction of different kinds of bounds: tight, loose, big O, little o, Omega, etc.

II. Lists

a) Materials
- ♦ Textbook sections: 3.2 through 3.5, focus: List ADT, ArrayList, Linked-List, Iterator
- ♦ Moodle material: Implementation of Iterator (source code)
- ♦ Homework:
  - Homework 1: ArrayList
  - Homework 2: Linked-List and Iterator

b) Question patterns
- ♦ Simple implementations of some common operations on Lists: add, remove.

III.Stacks & Queues

a) Materials
- ♦ Textbook sections: 3.6, 3.7
- ♦ Moodle material: Source code: Stack, Queue, Solving Palindrome using Stack

b) Question patterns
- ♦ Distinction between Stack and Queue (LIFO and FIFO)
- ♦ Implementations of add/ remove on Stack/Queue
- ♦ Reverse Strings or Array using Stack

Implementation alternatives: ArrayList, Fixed Size Array, Circular Array, Growing Array

IV. Trees

a) Materials

- ♦ Textbook sections:
  - Tree implementation and overview on traversals: 4.1
  - Binary Trees: 4.2
  - Binary Search Trees: 4.3
  - AVL Trees: 4.4
  - Splay Trees: 4.5
  - More on tree traversals: 4.6
  - ~~B-Trees: 4.7~~
  - Red-Black Trees: 12.2
- ♦ Moodle material:
  - Source code: Binary Tree Construction and Traversal
  - Slides: Useful slides on Binary Tree
- ♦ Homework:
  - Homework 3: Binary Tree and Binary Search Tree
- ♦ Quiz:
  - Quiz 2: Binary Tree, Binary Search Tree
  - Quiz 3: AVL Tree: definition, rotation analysis
  - Quiz 4: AVL, Splay and Red-Black Trees

b) Question patterns

- ♦ Binary Tree/ Binary Search Tree:
  - be able to tell them apart from normal trees
  - given a Binary Search Tree, add some values into it
  - add/ remove/ contain implementation
- ♦ AVL Trees:
  - Definition/ Advantage
  - Be able to handle rotations: given an AVL Tree, insert some value into it
- ♦ Splay Trees:
  - Definition/ Advantage
  - Be able to handle rotations to push an accessed item to the root
- ♦ Red-Black Trees:
  - Definition/ Advantage
  - Be able to tell whether or not a tree is a red-black tree
  - Be able to handle rotations and color repainting as items are inserted/removed
- ♦ Tree traversals:
  - Be able to distinguish and recognize the four traversal methods: in-order, pre-order, post-order, and breadth-first
  - Implement one of the four traversal methods

V. Priority Queues/Heaps

a) Materials

- ♦ Textbook sections: 6.1 through 6.4
- ♦ Moodle material:
    - Source code: Bounded Priority Queue
    - Slides: useful slides on Binary Heap
- ♦ Homework:
    - Homework 4: Heap and Heap Sort.
    - Homework 5: Simulation of task scheduling using priority-queue.
- ♦ Quiz Solution:
    - Quiz 5: Max Heap/ Min Heap

b) Question patterns

- ♦ Definition/ Advantage
- ♦ Be able to show how insert/ remove an element from a Max/Min Heap works

VI. Hash Tables and Functions

a) Materials

- ♦ Textbook sections: 5.1 through 5.6, focus: Hash Functions, Collision Resolving, Rehashing
- ♦ Moodle material: Slides: useful slides on Hash Table
- ♦ Quiz solution: quiz 6: insert a value, rehashing, function analysis

b) Question patterns

- ♦ Given a hash table, insert a new value, deal with probing and show the result
- ♦ Given a full hash table, rehash it to a bigger hash table
- ♦ Analyze hash functions: which is the best?

VII. Algorithm Design Techniques

a) Materials

- ♦ Textbook sections:
    - Greedy Algorithms: 10.1 **Dijkstra's Shortest Path, Prim’s and Kruskal’s MST algorithms**
    - Divide and Conquer: 10.2 **Quick Sort**
    - Dynamic Programming: 10.3 **Optimal Coin Collection problem on Moodle, Fibonacci**
- ♦ Moodle material:
    - Slides: Divide and Conquer: Quick Sort and Merge Sort
    - Tutorial: Dynamic Programming
    - Source Code:
        - ∗ Dynamic Programming
        - ∗ Fibonacci and Factorial

b) Question patterns

- Definitions / Advantages
- Be able to show the steps of Merge Sort
- Be able to analyze the efficiency of Dynamic Programming.
- Implement Fibonacci or Factorial using dynamic programming

VIII. Graphs

a) Materials

- Textbook sections: 9.1, 9.2, 9.3.2: Dijkstra's Shortest Path
- Moodle material: Source Code: Greedy Algorithm: Dijkstra's Shortest Path

b) Question patterns

- Definitions / Advantages
- Understand the mechanism of Dijkstra's Shortest Path

General Guidelines for every Data Structure:

- Draw diagrams of states to demonstrate operations with the data structure.
- Know the runtime complexity of the operations in the data structure.
- Know at which situation, the data structure is appropriate to use.
- Strengths and weaknesses of the data structure and associated algorithms.
- Given partial code of a data structure, implement a certain operation.

* **Different alternatives for implementing a data structure, and their trade-offs**

THE END