CS302 – Data Structures

Spring 2012 – Dr. George Bebis

Midterm Exam Study Guide

C++ review material (Sections 2.2-2.4, 3.3, 4.2 and my slides)

- Function call by value or reference
- Pointers and dynamic array allocation.
- Differences between static and dynamic array allocation. Constructors, copy-constructors, and destructors.
- Operator overloading (especially the assignment operator).

Comparison of Algorithms (Section 2.6 and supplement)

- Why do we need to compare algorithms?
- How do we compare algorithms?
- Asymptotic Analysis
- Big-O notation

Stacks and Queues (Sections 5.1, 5.2, 5.3, 5.4, 6.1)

- Very good knowledge of the Stack/Queue specification and implementation using arrays and linked-lists.
- Implementation details (e.g., using "front" and "rear" to determine whether the queue is empty or full).
- Running times for all the functions of the Stack and Queue implementations.
- Tradeoffs (e.g., memory or time) between array-based and linked-list-based implementations.

Unsorted and Sorted Lists (Sections 3.1, 3.2, 3.4, 3.5, 4.1, 4.2, 4.3)

- Very good knowledge of the Unsorted/Sorted list specification and implementation using arrays and linked-lists.
- Implementation details (e.g., insert at the end/front of the list for unsorted lists implemented using arrays/linked-lists).
- Binary search algorithm

- Running times for all the functions of the Unsorted and Sorted List implementations.
- Tradeoffs (e.g., memory or time) between array-based and linked-list-based implementations.

. **Recursion** (Chapter7)

- What is recursion?
- How do we implement a recursive function?
- Why are function calls expensive?
- Activation record and run-time stack.
- Iterative solutions vs recursive solutions (trade-offs).
- Recursive implementation of binary search.
- Inserting/Deleting items to/from a sorted list recursively.
- Redundancy of computations

Binary Search Trees (Chapter 8)

- Definitions: binary tree, height, node level, root, leaves.
- How to compute the height if we know the number of its nodes? **Learn the proof**.
- How to compute the number of nodes if we know its height? **Learn the proof.**
- Binary search tree property.
- Very good knowledge of the Binary Search Tree specification and implementation.
- Inserting/Deleting nodes (understand them from an algorithmic point of view - don't memorize code) - Running times are important.
- Tree traversals: inorder, preorder, postorder.
- Running times for all the functions of the Binary Search Tree type.
- Binary search (array-based) versus binary search tree.

Red-Black Trees (slides and supplement)

- Definitions
- Properties of red-black trees
- log(N) height **Learn the proof**.
- Tree rotations
- Tree operations with emphasis on InsertIterm.

General Comments

The midterm exam will be closed-books, closed-notes. The format will be similar to the sample midterm exams posted on the course's webpage (True/False, Short Answer Questions, Coding). Make sure that you understand the *trade-offs* between different implementations (e.g., array-based queues versus linked-list-based queues). It is also very important to know how to compute the running-time of an algorithm and how to express it in terms of big-O notation. Make sure you that you go over the examples we did in class. Also, do as many problems as you can from each chapter and review questions from quizzes and homework.