Midterm Exam Review

Midterm Exam Duration: 2 HRS Portion: Chapter 1-10

Midterm Consists of the following,

- True or False, give your answer why it is true or false
- Multiple choice questions
- Brief answers
- Able to write a Pseudo code for the given problems.
- Able to write a Recursive Algorithms.
- Able to trace manually the working principles learned.

Prepare yourself in the following areas,

1. Asymptotic Analysis

- Big-O, Big-Omega, and Big-Theta
- Analyze the runtime performance of the given codes.
- Understand the difference between best-case, average-case, and worst-case runtime.

2. ADTs and data structures:

- The difference between an ADT and a data structure.
- Stacks, Queues, Array list, Linked list, Trees, Heaps and Priority Queues ADT: properties, operations, runtimes, and when to use them.

  Note: ADT operations will be given in the exam. No need to memorize. You should know the purpose and how to use it.

3. Sorting

- Need to know the summary table of all types of sorting and its performance.
- Need to know the working procedure by using the following sorting.
  - Heap sort

    - Merge sort
    - In-Place Quick sort
    - Bucket Sort
    - Radix Sort

4. Brief Answers

- Know different strategies of picking the pivot for QuickSort and compare the performance of them.
- What is In-place and Stable sort. Give examples.
- Min-heap operations and how to represent in an array.
- How recursion plays a role in Merge sort and Quick sort?
- Compare heaps vs array to implement Priority queue in terms of performance of PQ operations.
- How to analyse the merge sort time efficiency using tree structure.

5. Pseudocodes and Runtime Analysis for the given problem requirements. You will get ADTs to work on these problems.

- Lesson-3,How to work with Stack. Refer your homework.
- Lesson-4, Able to provide a recursive solution and analyse it's efficiency.
- Lesson-5, Binary Tree Algorithms (Traversal, In-Class Exercises)
- Lesson-6, Heap Algorithms. (Properties, Procedure to insert, delete, findMin, how to represent heap in a tree structure and array)