EECS560 Final Exam Review Guide Fall, 2018

Administrative

- **When:** 1:30pm-4:00pm, Friday, December 14, 2018
- **Where:** 1136 Learned

Read carefully all of the following items....

- Bring your KUID. You will not be allowed to take the final exam without your KUID.
- Unless otherwise stated, all material from the assigned readings, lectures and lecture notes, written and lab assignments are fair game for exams.
- The exam will be closed book and closed notes.
- No calculators, cell phones, head phones, or electronic devices of any sort will be allowed. No such devices should be out in the open.
- Once you start the exam, you will not be excused from the room for any reason unless you turn in your exam. Once it is turned in, you cannot come back and continue working on it.
- You must write legibly and show all your work clearly for credit. Partial credit will only be given to meaningful answers.
- You will be graded according to your approach to the problems, mathematical rigor, and quality of your solutions. A correct but inefficient algorithm or data structure will receive very little credit in this course.
- No make-up exam will be given for any unauthorized absence.
- Any unauthorized absence will automatically result in a zero for this final exam.

Warning

Any unauthorized absence from this final exam will result in a zero for this exam. If you come in late after the exam has already begun, you will not be allowed any extra time to complete your exam.

Exam Coverage

This is a comprehensive final examination and you are responsible for all topics covered in this course including all materials in our lectures, lecture notes, reading assignments, homework, labs and lab assignments. Please be sure to re-examine and study all those topics covered in the first and second Exam Review Guides as posted earlier this semester.

Topics from Exam 1 and Exam 2

- See Exam 1 and Exam 2 Review Guides.

Performance Analysis of Algorithms and Data Structures

- Understand the RAM computational model and be able to use it to analyze the performance of algorithms and data structures.
- Understand the definitions and basic properties of basic asymptotic relations including big-O, big-Ω, and big-Θ.
- For a given algorithm and/or program segment, be able to compute the best-case and worst-case complexity in closed-form and/or asymptotic forms using (1) detailed analysis, (2) basic operation(s), and (3) dominating steps as discussed in class.
- Understand the importance of efficiency of algorithms and data structures and be able to

compare the performance of different algorithms and estimate their resource consumption based on their complexity functions and input size.

General Data Structure Design and Analysis

new chapter represents highlights

- Understand the functionalities and applications of ***all*** ADTs discussed in class, including Hash table, binary search tree and optimal binary search tree, 2-3 tree, k-heap, dual heap, minMax heap, leftist heap, skew heap, pairing heap, Binomial queue, Fibonacci heap, and AVL tree. All data structures and their implementations are based on our lectures and lecture notes; any other variations, if any, will not be accepted.
- Understand the most efficient implementation of each ADT, its implication on performance, and be able to illustrate the underlying data structures used in implementing each ADT.
- Understand the "good" and "bad" of each ADT and be able to compare the performance of different ADTs both analytically and experimentally.
- For a given set of records, be able to build and illustrate a given ADT using the most efficient algorithm.
- For a given ADT, be able to implement, execute, and illustrate different static and dynamic operations associated with the given ADT. Typical operations may include findMin, findMax, find, insert, deleteMin, deleteMax, delete, changeKey, and concate operations.
- For a given ADT, based on its most efficient implementation, understand and be able to illustrate the best-case, worst-case, average-case, and amortized complexity of various operations as discussed in class.
- For any given application, understand and be able to select an efficient ADT and its data structure to support the implementation of efficient algorithms.

Experimental Profiling

- Understand and be able to design and perform experimental profiling in comparing the performance of different ADTs.

12/1/2018