
Exam 2 Announcement

When: Tuesday, April 3, 2007, 8:15 P.M.-10:15 P.M. Please show up a few minutes early so that you can take advantage of the full time available.

Where: Room 113 IST (the same building as our offices).

Crib sheet: The exam is *closed book*. You may, however, bring one *handwritten* crib sheet on an $8\frac{1}{2} \times 11$ or A4 *colored* sheet of paper. The color of the paper has to be significantly different from white. Preparing a crib sheet can be a useful study aid, so take time in selecting material for it. You may use both sides of the paper and write as small as you like, but you are allowed only one sheet and it must be handwritten. Calculators and programmable devices are not allowed for this exam.

Material covered: You are responsible for all material covered in the lectures, reading assignments, and homework through the lecture on Friday, March 23 (the last lecture before we started graphs). The course home pages has a record of the material covered in lectures and reading assignments.

Exam review: The lecture on Monday, April 2 will be devoted to review for the exam. Please come prepared with questions.

Practicing for the exam: In addition to the practice exam, if you need more problems to practice on, please look at the problems and exercises in the chapters we covered.

General Notes:

- For all the algorithms in the list of topics below, you should understand how to prove their correctness and analyze their running time. You should also understand subroutines we developed while building up the algorithms above (e.g. merging sorted arrays, partitioning a sorted array around a pivot, heapify, etc). Most importantly, you should be able to adapt the ideas to we've seen to **design new algorithms**.
- You should understand the pseudocode used in CLRS and be able to describe your algorithms both in English and using pseudocode (your pseudocode does not have to be the same as in the book).
- Some questions will require you to be comfortable with material covered in the mathematical prerequisites. For example: arithmetic and geometric sums, logarithms and exponents, elementary probability. Appendices A, B and C of CLRS are a good place to review this material.

List of topics. Material from Exam 1, plus:

- Median and order statistics. How long does it take to find the i th smallest element (element of rank i) in the array? Randomized Selection and how it works. PARTITION.
- Abstract Data Types (ADT's): what they are and how to formulate a list of required operations.
- Basic structures: Linked lists, queues, stacks.
- Priority Queues and Heaps:
 - Basic operations on heaps: HEAPIFY, INCREASE-KEY, INSERT, DELETE, EXTRACT-MAX, BUILDHEAP, HEAPSORT. Combinatorics of heaps: height and depth in a binary tree. Counting the number of nodes of a given height or depth.
- Hash tables
 - Dictionary ADT
 - Hashing with chaining. INSERT, DELETE, SEARCH. Simple uniform hashing assumption. Analysis of running time of unsuccessful search with constant load.
 - Choosing a hash function. Multiplication and division methods.
- Search trees:
 - Binary search trees: search, insert, delete, minimum, maximum, predecessor, successor. Traversals: inorder, preorder, postorder. Range search (Homework 7).
 - 2-3 trees. Basic operations: search, insert, delete. Combinatorics of 2-3 trees: counting nodes at different depths. Proof that 2-3 trees have logarithmic depth.
- Augmenting Data Structures
 - Basic methodology. You should be able to apply it to design a new data structure that supports a given list of operations.
 - Examples of augmented data structures: Order-statistic trees: OS-SELECT, OS-RANK. Interval trees: INTERVAL-SEARCH (finds one matching interval), how to find all matching intervals. Implementing a dictionary that supports searching on more than one key.

List of topics from first exam:

- Asymptotic notation ($O, \Omega, \Theta, o, \omega$).
- Correctness proofs using loop invariants.
- Sorting algorithms: Insertion Sort, Merge Sort, Quicksort, Randomized Quicksort. You should know how each algorithm works, its running time, and whether or not it is in place. You should be aware of input arrays which cause the algorithm to perform especially well or especially poorly.
- Recurrences: recursion trees, substitution, Master Theorem (know by case number).
- Randomized algorithms: what are they? What sort of analyses are meaningful? Randomized Quicksort and Randomized Selection.
- Divide and conquer: Merge Sort, binary search, integer multiplication, exponentiation, Quicksort, polynomial multiplication, median of two sorted arrays, stock price problem from homework 3, Groundhogs and Holes. For each of these problems, you should be able to come up with the recurrence that the running time satisfies, and know how to solve it.