

Homework 5 – Due Friday, March 2, 2007

Photos on Angel You have a chance to earn 5 bonus points on this homework! And it is really easy! To earn the bonus (and our gratitude), please put a recognizable photo of yourself on Angel by next Friday. That easy! Hopefully, the pictures will help us remember your names.

Exercises These should not be handed in, but the material they cover may appear on exams.

- Abstract data types: CLRS Ex.10.1-6, 10.1-7
- Heaps: CLRS Ex. 6.1-1 to 6.1-7, 6.2-1 to 6.2-4, 6.2-6, 6.3-1, 6.3-2, 6.4-1, 6.4-3, 6.4-4 and the following problem:
 1. (**Spaghetti sort**) Imagine a handful of uncooked spaghetti, individual rods whose lengths represent numbers that need to be sorted.
 - (a) Outline a “spaghetti sort” – a sorting algorithm that takes advantage of this unorthodox representation.
 - (b) What does this have to do with heapsort?
- Priority Queues: CLRS Ex. 6.5-1 to 6.5-7

Problems to be handed in

1. Please give a short English description and pseudocode:
 - (a) (**Iterative MAX-HEAPIFY**) CLRS Ex. 6.2-5
 - (b) (**Merging k sorted lists**) CLRS Ex. 6.5-8
2.
 - (a) (**Heaps: counting elements of height h**) CLRS Ex. 6.3-3
 - (b) (**Correctness of heapsort**) CLRS Ex. 6.4-2
3. (**Implementing Priority Queues with HEAP-INCREASE-KEY and HEAP-DECREASE-KEY**) You need to choose a data structure to schedule jobs on your supercomputer. Each job x has the following attributes: $key[x]$ is the priority (higher priority means more important) and $id[x]$ is the identifier of the job, which is an integer from 1 to 999. You need to ensure that jobs with higher priorities take precedence. You decide that the data structure you need is a priority queue, and that you are going to implement it using heaps.
 - (a) (**warm up**) Since your heap will contain jobs, not just numbers, you need a tiny modification to the code for manipulating heaps, given in the book. Explain what you need to change in HEAP-INCREASE-KEY.

- (b) Using HEAP-INCREASE-KEY and MAX-HEAPIFY as a guide, write pseudocode (from scratch) for HEAP-DECREASE-KEY(A, i, key) which takes max-heap A , index i and value key , decreases $key[A[i]]$ to key and restores the heap-property. Your procedure should run in $O(\log n)$ time, where n is the current number of elements in the heap.
- (c) Implement the following priority queue operations using heaps and making calls *only* to procedures HEAP-DECREASE-KEY and HEAP-INCREASE-KEY. All operations should run in time $O(\log n)$. Indicate which operations run in constant time.
- INSERT(Q, x) inserts element x into the queue Q .
 - DELETE(Q, i) deletes the element in position i from the queue Q . (Don't forget to restore the heap property after deleting.)
 - FINDMAX(Q) returns the maximum element in Q , but does not delete it.
 - EXTRACT-MAX(Q) deletes the maximum element from the queue Q and returns it.
- (d) You want to allow removing a job from the queue or changing its priority when only the job's id is specified, but not its position in the queue. You decide to keep an array *Position* that stores the current position of each job in the heap. Show how to implement the two operations that access jobs by id. Make sure the running time is still $O(\log n)$.

4. (Young tableaus) CLRS Problem 6.3