Problem Set 3

This problem set is due **in lecture** on **Monday, October 6.**

Reading: Chapter 9, Sections 11.1–11.3, Sections 12.1–12.3

There are **four** problems. Each problem is to be handed in separately. Mark the top of each sheet with your name, the course number, the problem number, your recitation section, the date, and the names of any students with whom you collaborated.

You will often be called upon to "give an algorithm" to solve a certain problem. Your write-up should take the form of a short essay. A topic paragraph should summarize the problem you are solving and what your results are. The body of your essay should provide the following:

1. A description of the algorithm in English and, if helpful, pseudocode. English explanations should be used liberally in your pseudocode. For convenience, pseudocode may use standard arithemetic and logical operations, as well as loops and data structures like arrays. Pseudocode should be understandable to anyone who can program–it should **not** be ready to compile!

2. At least one worked example or diagram to show more precisely how your algorithm works.

3. A proof (or indication) of the correctness of the algorithm.

4. An analysis of the running time of the algorithm.

Remember, your goal is to communicate. Graders will be instructed to take off points for convoluted and obtuse descriptions.

Problem 3-1. Heap Operations

For parts (a), (b), (c) and (d) of this problem, you may assume that the *input* heaps are nearly complete binary trees; for parts (a), (b) and (c) your *output* heaps need not be nearly complete. Note that the input heaps are not given to you in the form of arrays and your output heaps need not be in the form of arrays.

(a) Give an algorithm to merge two heaps of size m and n (assume $n \geq m$), into a single heap of size $(m + n)$ in time $O(\log n)$.

(b) Suppose you are given a heap H of size m and n unordered elements, where $n \geq m$. What is the runtime to insert these n elements into the heap H? What is the total runtime required to build a second heap H' of size n and then merge H and H' into a single heap?

(c) Give an algorithm to merge k heaps each of size n into a single heap of size nk in time $O(k \log n)$.

(d) Given a min-heap of n distinct elements and a real number x, give an algorithm to determine whether the kth smallest element is less than x in $O(k)$ time.

Hint: You do not have to extract the kth element. You only need to find its relation to x.

Problem 3-2. Political Machinations

Noted actor Ranier Wolfcastle decides to hold a "town hall" meeting as part of his campaign for Governor of California. His political advisor, Rover Karlson, is in charge of hand-picking an audience and asks you for help.

(a) Rover tells you that he only wants middle-class voters in the audience. He decides to eliminate anyone in the top or bottom $(1/k)$th income bracket. Unfortunately, you cannot find out anyone's actual income. You can only compare two people and see who makes more money. Given a list of n potential audience members, give an $O(n)$ time algorithm to find the people with the n/kth largest and smallest incomes.

(b) Rover decides to sort the list of potential audience members into k equal-sized (to within 1) groups by income. Denote these groups as $G_1, \ldots, G_k$. The groups must have the property that $\forall i < j$, every person $g_a \in G_i$ has a lower income than every person $g_b \in G_j$. Rover doesn't care about the ordering within any particular group G_i. Give an $O(n \log k)$ time algorithm to produce k such groups.

Problem 3-3. Set Operations

Suppose you are given two sets of integers, S and T. Let $m = |S|$ and $n = |T|$ and suppose $n \geq m$. You may assume any comparison or arithmetic operation (e.g. addition, multiplication) on two integers takes unit time.

(a) Give a deterministic algorithm for finding $S \cap T$ in $o(n^2)$ time.

(b) Give a randomized algorithm for finding $(S \cap T)$ in expected $O(n)$ time.

Problem 3-4. $d(v)$**-ary Search Trees**

A $d(v)$-ary Search Tree ($d(v)$-ST) is a Search Tree in which each non-leaf node has $d(v)$ children. The function d maps nodes to integers.

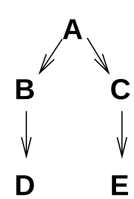
(a) Let $d(v) = c$ for some integer c. Suppose we build a complete $d(v)$-ary Search Tree on n nodes. What is the height of the tree in terms of n and c? What is the runtime of a SEARCH operation?

(b) Let $size(v)$ be defined as "the number of nodes in the subtree rooted at v, not including v". For a leaf node v, $size(v) = 0$. Suppose $d(v) = \lfloor\sqrt{size(v)}\rfloor$. The figure below depicts such a tree in which the function d has the following values:

$d(A) = \lfloor\sqrt{size(A)}\rfloor = \lfloor\sqrt{4}\rfloor = 2$,

$d(B) = d(C) = \lfloor\sqrt{size(B)}\rfloor = \lfloor\sqrt{1}\rfloor = 1$

$d(D) = d(E) = \lfloor\sqrt{size(D)}\rfloor = \lfloor\sqrt{0}\rfloor = 0$



What is the height of a complete $d(v)$-ary Search Tree with n nodes? What is the runtime of a SEARCH operation? For convenience, you may assume that $\sqrt{size(v)}$ is always an integer.