Problem Set 4

This problem set is due **at 11:59pm** on **Thursday, March 5, 2015.**

Please turn in each problem solution separately. Each submitted solution should start with your name, the course number, the problem number, your recitation section, the date, and the names of any students with whom you collaborated.

Exercise 4-1. Read CLRS, Chapter 17. (Chapter 16 in 4th Edition)

Exercise 4-2. Exercise 17.1-3. (16.1-3 in 4th Edition)

Exercise 4-3. Exercise 17.2-2. (16.2-2 in 4th Edition)

Exercise 4-4. Exercise 17.3-2. (16.3-2 in 4th Edition)

Exercise 4-5. Read CLRS, Chapter 7. (Chapter 7 in 4th Edition)

Exercise 4-6. Exercise 7.1-3. (7.1-3 in 4th Edition)

Exercise 4-7. Exercise 7.2-5. (7.2-5 in 4th Edition)

Exercise 4-8. Exercise 7.4-4. (7.4-4 in 4th Edition)

Problem 4-1. Extreme FIFO Queues [25 points]

Design a data structure that maintains a FIFO queue of integers, supporting operations ENQUEUE, DEUEUE, and FIND-MIN, each in $O(1)$ amortized time. In other words, any sequence of m operations should take time $O(m)$. You may assume that, in any execution, all the items that get enqueued are distinct.

(a) [5 points] Describe your data structure. Include clear invariants describing its key properties. *Hint:* Use an actual queue plus auxiliary data structure(s) for bookkeeping.

(b) [5 points] Describe carefully, in words or pseudo-code, your ENQUEUE, DEQUEUE and FIND-MIN procedures.

(c) [5 points] Prove that your operations give the right answers. *Hint:* You may want to prove that their correctness follows from your data structure invariants. In that case you should also sketch arguments for why the invariants hold.

(d) [10 points] Analyze the time complexity: the worst-case cost for each operation, and the amortized cost of any sequence of m operations.

Problem 4-2. Quicksort Analysis [25 points]

In this problem, we will analyze the time complexity of QUICKSORT in terms of error probabilities, rather than in terms of expectation. Suppose the array to be sorted is $A[1\,..\,n]$, and write x_i for the element that starts in array location $A[i]$ (before QUICKSORT is called). Assume that all the x_i values are distinct.

In solving this problem, it will be useful to recall a claim from lecture. Here it is, slightly restated:

Claim: Let $c > 1$ be a real constant, and let α be a positive integer. Then, with probability at least $1 - \frac{1}{n^\alpha}$, $3(\alpha + c)\lg n$ tosses of a fair coin produce at least $c \lg n$ heads.

Note: High probability bounds, and this Claim, will be covered in Tuesday's lecture.

(a) [5 points] Consider a particular element x_i. Consider a recursive call of QUICKSORT on subarray $A[p\,..\,p+m-1]$ of size $m \geq 2$ which includes element x_i. Prove that, with probability at least $\frac{1}{2}$, either this call to QUICKSORT chooses x_i as the pivot element, or the next recursive call to QUICKSORT containing x_i involves a subarray of size at most $\frac{3}{4}m$.

(b) [9 points] Consider a particular element x_i. Prove that, with probability at least $1 - \frac{1}{n^2}$, the total number of times the algorithm compares x_i with pivots is at most $d \lg n$, for a particular constant d. Give a value for d explicitly.

(c) [6 points] Now consider all of the elements $x_1, x_2, \ldots, x_n$. Apply your result from part (b) to prove that, with probability at least $1 - \frac{1}{n}$, the total number of comparisons made by QUICKSORT on the given array input is at most $d' n \lg n$, for a particular constant d'. Give a value for d' explicitly. *Hint:* The Union Bound may be useful for your analysis.

(d) [5 points] Generalize your results above to obtain a bound on the number of comparisons made by QUICKSORT that holds with probability $1 - \frac{1}{n^\alpha}$, for any positive integer α, rather than just probability $1 - \frac{1}{n}$ (i.e., $\alpha = 1$).

6.046J / 18.410J Design and Analysis of Algorithms
Spring 2015