CS 181b
Advanced Topics in Algorithms
Spring 2002
Problem Set 1a
Due Thursday, January 24 in class

Please remember that all problem set submissions must be typeset, preferably in LaTeX. Please keep an electronic version of your solution set for your records.

1. **[10 Points] Backup Stacks.** In class we considered a stack data structure with operations PUSH, POP, and MULTIPOP. Now assume that there is some limit on the size of the stack. Let's call this maximum stack size k. Imagine that after every k operations are executed, the entire contents of the stack gets backed up (saved) to disk. The real cost of backing up the stack is equal to the number of elements on the stack. Show that the cost of any n stack operations is still $O(n)$. You should give 2 proofs of this: One using the aggregation method and one using the accounting method.

2. **[15 Points] Minqueues.** Consider an extension of the queue abstract data type, called a minqueue, which supports operations ENQUEUE, DEQUEUE, and FINDMIN. Assume that the elements to be stored are integers (or any other totally ordered data type). FINDMIN simply returns the smallest element in the minqueue, but does not remove it. Using a standard implementation of a queue, the FINDMIN operation takes $O(n)$ time in the worst case. However, consider a more clever data structure for this abstract data type which works as follows: There are two regular queues used. The first queue is called the "real queue" and the second queue is called the "helper queue". When the ENQUEUE(x) operation is performed, the element x is enqueued in the regular way on the real queue. The element x is also enqueued at the end of the helper queue. However, if the element y immediately "in front" of x on the helper queue is larger than x, then y is removed from the helper queue. This process of x annihilating the element immediately in front of it is repeated until the element immediately in front of x is less than or equal to it or x is at the front of the helper queue.

   (a) Describe how the DEQUEUE and FINDMIN operation are implemented in this data structure.

   (b) Give the worst case running tim for each of ENQUEUE, DEQUEUE, and FINDMIN using this data structure.

   (c) Give three different arguments (one using aggregation, one using accounting, and one using potential) to show that any sequence of n ENQUEUE, DEQUEUE, and FINDMIN operations requires only $O(n)$ time.