Advanced Data Structures 1	Moshe Lewenstein
Exercise 3	Submit by 27.12.2005

Include your name, ID number, and exercise number. **Solutions:** Try to solve the problem on your own. You may also discuss the problem with others. However, (1) you **must** write the answer on your own and in your own words, (2) write down the names **next to each answer** of all the people you collaborated with on each question (this does not reduce points, but is intended to increase honesty).

1. Say we have a data structure supporting insert, delete, predecessor and successor over the integers ($\{0, 1, 2, ...\}$). Show that in the comparison model one of the operations requires time $\Omega(logn)$. Does this mean that y-fast-tries do not work in the comparison model?

2. Let $S \subseteq U = \{0, 1, ..., u-1\}$. A range query is of the form: $range(x, y) = \{z \in S | x \leq z \leq y\}$ for $x, y \in U$, and the range size is: $rangesize(x, y) = |range(x, y)|$. The dynamic range query problem over universe U is to maintain S allowing insertions, deletions and range queries (i.e. range(x,y)).

   The parameters of the problem are $u = |U|$, $n = |S|$, and $rangesize(x, y)$.

   (a) Suggest efficient algorithms for the range query problem (efficiency - relate to space and time). Do not forget to provide an analysis.

   (b) Say we want to support insertions, deletions and $rangesize$ queries, i.e. given x, y answer $rangesize(x, y)$ (here we do not bother with range queries). Suggest efficient algorithms for the range query problem (efficiency - relate to space and time). You may provide a few answers and explain the tradeoffs. Do not forget to provide an analysis.

3. Assume we are given a dynamic perfect hash function construction which constructs perfect hash functions h such that h is

   (a) constructible in deterministic (rather than randomized), linear time.

   (b) $h(.)$ can be evaluated in $O(1)$ worst case, deterministic time.

   (c) insertions and deletions take $O(log^5 u)$ (worst case).

   Suggest a data structure supporting insertion, deletion, predecessor and successor in amortized, deterministic (rather than randomized) $O(loglogu)$ time and linear space (linear in n, not u).

4. Give a constant time algorithm for determining the Hamming weight of a k-bit string, i.e. the number of 1's it contains) where the wordsize $w \geq k(2 + \log_2 k)$.