Homework 6 (Posted 19th March, Due during or before class 26th March (programming assignment due by 28th March 11.59 p.m.))

Policy for Programming Assignment: Problem 5 has a programming assignment. There is no design part for this one. However, the programming assignment is still due by 28th March 11.59 p.m.. If your name is John Smith, then name your program as JohnSmith.c and email it to yjkim78@gradient.cis.upenn.edu.

Problem 1: 6 You need to sort $n$ integers in the range 1 to $n^2$. Give an $O(n)$ algorithm.

Problem 2: 8 pts You have $n$ integers in the range 1 to $k$. You need to preprocess the input suitably, so that after preprocessing you can answer some queries in constant time. More specifically: given any two real numbers $a$ and $b$, you have to answer queries about how many of the integers fall into the range $(a, b]$ in $O(1)$ time (the range excludes $a$ and includes $b$). During preprocessing, you do not know the values of $a$ and $b$. You can use $O(k)$ additional storage and $O(n + k)$ preprocessing time.

Problem 3: 8 pts You have an array of $n$ data records. Each record has a key 0 or 1. Design a $O(n)$ algorithm to sort the data records according to the key values (those with key 0 should come before those with key 1). You can only use constant amount of additional storage during the sorting. Can you use your solution in radix sort so as to sort $n$ records with $b$ bit keys in $O(bn)$ time? (every key has $b$ bits). Justify your answer.

Problem 4: 8 pts You have to sort a sequence of $n$ elements. The $n$ elements have $n/k$ subsequences of size $k$ each. The subsequences have the following property: All elements of a subsequence are less than those of the preceding one and greater than those of the following subsequence. An example sequence of 6 elements with $k = 2$ is 2, 1, 5, 6, 21, 12. The subsequences here are 2, 1, 5, 6 and 21, 12. Note that all elements of 2, 1 are less than those of 5, 6 and so on. You know the value of $k$. Give an $O(n \log k)$ algorithm to sort the entire sequence. Show that any comparison based sorting needs at least $\Omega(n \log k)$ operations. (It is not rigorous enough to combine the lower bounds for the individual subsequences. )

Problem 5: 10 pts Program the quick sort algorithm as taught in class.