CS 381, Fall 2014; Instructor: Elena Grigorescu
TAs: Nabeel, Akash
Due Tue September 30, 12:01 pm sharp, in hard copy.

Problem set 1

You may discuss the problem sets with other students in the class, however you *must* write up the solutions yourself. If you collaborate, specify who you worked with. No other solution sources are allowed. The solutions may be hand-written.

1 Solve and turn in

Problem 1

Give the solution to the following recurrences by applying the Master theorem when possible, or by any other of the methods we've learned when the Master theorem doesn't apply. Assume $T(1) = 1$.

1. $T(n) = T(n/5) + T(7n/10) + 1$
2. $T(n) = 2T(n-1) + 5$
3. $T(n) = 6T(2n/5) + \log^2 n$
4. $T(n) = 2T(n/2) + n^{1.4}$
5. $T(n) = 12T(n/12) + 11n$
6. $T(n) = T(\sqrt{n}) + 12$.

Problem 2

Dan and Alex play the following game: Dan picks an integer from 1 to m and Alex is trying to figure it out by asking Dan as few questions as possible. Dan is willing to respond truthfully, but he announces that he will only answer 'yes' or 'no' to any of Alex's questions. Design a strategy such that Alex can always guess the number using only $o(m)$ questions.

Problem 3

You are given two n-digit numbers $a, b > 0$, and have to come up with a fast algorithm to multiply them.

1. What is a trivial upper bound on the number of steps it takes to compute the product ab ?
2. Design a divide-and-conquer procedure that uses $O(n^{1.58})$ digit-operations.

Problem 4 a) Consider the question of Selection in worst case linear time discussed in Chapter 9.3 and in class. In this problem the goal is to find the median element in an array of size n, by using the algorithm described in the book (page 220), with the following modification: in step 1, instead of partitioning the array into groups of size 5, we partition it into groups of size 11 and then perform the same steps according to the new grouping. Analyze the running time of the modified algorithm.

b) Consider the same problem, but where now in step 1 we group the elements into groups of size 3. Show an upper bound on the running time of the algorithm. Then show that your upper bound is also a lower bound, by providing a worst-case example.

Problem 5 Optional (do not turn in.)

Suppose you are given n points in the plane with coordinates $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, with $x_i \neq x_j$ and $y_i \neq y_j$ for all $i, j \in [n]$. Design an $O(n \log n)$ algorithm that finds the/a pair of closest points (under the Euclidean metric) (x_i, y_i) and (x_j, y_j) among the n points. (The Euclidean distance between points $p = (x_i, y_i)$ and $q = (x_j, y_j)$ is $d(p, q) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.)