CS 341, Fall 1999 due Friday Oct. 15

A. Lubiw, D. Schuurmans

ASSIGNMENT 2

Questions 1–4 are worth 10 points each. Do not copy from others, and acknowledge your sources.

1. Read section 7.6 of the text on matrix multiplication. Show how to multiply the complex numbers $a+bi$ and $c+di$ using only 3 real multiplications. Input is a, b, c and d; output is the real component and the imaginary component of the answer separately.

2. The k-th quantiles of an n-element set are the $k-1$ elements that divide the sorted set into k equal-sized sets (to within 1). For example, one set of 4-th quantiles of the set $\{1, \ldots, 10\}$ consists of $\{2, 5, 8\}$ which divides the set into $\{1\}, \{3, 4\}, \{6, 7\}, \{9, 10\}$. Give an $O(n \log k)$ algorithm to find the k-th quantiles. Hint: you may call a linear time selection algorithm.

3. The English coinage before decimalization included half-crowns (30 pence), florins (24 pence), shillings (12 pence), sixpences (6 pence), threepennies (3 pence), and pennies. Show that with these coins the greedy algorithm for making change does not necessarily produce an optimal solution, even when an unlimited supply of coins of each denomination is available.

4. Imagine you are attending a one-day event with many short workshops, each scheduled for some time interval. You would like to design a schedule for yourself, with the goal of attending the maximum number of workshops that don't overlap in time. A greedy approach would be to go through the list of workshops in some order, adding to your schedule any workshop that doesn't conflict with the ones you have already chosen. Prove or disprove that the greedy method with each of the following orderings of workshops gives the maximum:
   (a) Order the worshops by start time.
   (b) Order the workshops by duration.
   (c) Order the workshops by finish time.

5. PROGRAMMING. Read the programming guidelines on the web page. You are given a set of strings $\{S_1, S_2, \ldots, S_n\}$, and you would like to find the shortest string S' that contains each of the S_i's as a substring. This is called the "Shortest Common Superstring Problem", and is needed in DNA sequencing. For example, a solution for the set {ocelot, to, otter}, is 'ocelototter', but a better solution is 'tocelotter'.

   Unfortunately, this problem is NP-hard. A standard heuristic involves finding the two strings with the maximum overlap, replacing them by the single merged string, and repeating until the set contains only a single string. For the above example, we would first form 'ocelotter' with overlap 2, then form 'tocelotter' with further overlap 1.

   Write a program to implement this heuristic. Modularize your code and describe your design clearly. Analyze your implementation using O notation. You need not have the most efficient implementation, but you should demonstrate an awareness of where you are being efficient/inefficient.