Aalto University, Department of Computer Science
Pekka Orponen

CS-E3190 Principles of Algorithmic Techniques (5 cr)
Exam Thu 15 Feb 2018, 5–8 p.m.

Write down on each answer sheet:
- Your name, degree programme, and student number
- The text: "CS-E3190 Principles of Algorithmic Techniques 15.2.2018"
- The total number of answer sheets you are submitting for grading

Note: You can write down your answers in either Finnish, Swedish, or English.

1. Arrange the following functions according to their increasing order of growth:

$$n, \quad \sqrt{n}, \quad n^{1/n}, \quad n^{1/3}(\log n)^2, \quad n \log\log n, \quad n^{-1}(\log n)^3,$$
$$\sqrt{n}\log(n^2), \quad 42, \quad \log n, \quad n/\log n, \quad 2^{(\log n)^2}, \quad 2^n.$$

   (Notation $\log n$ denotes here logarithm in base 2.) You do not need to prove the correctness of your ordering. *12p*

2. The *square* of a matrix A is its product with itself, AA.

   (a) Show that five multiplications are sufficient to compute the square of a 2×2 matrix.

   (b) Professor Boondoggle suggests the following algorithm for computing the square of an $n \times n$ matrix, for n a power of 2.

   > Use a divide-and-conquer approach as in Strassen's algorithm, except that instead of getting 7 subproblems of size $n/2$, we now get 5 subproblems of size $n/2$, thanks to part (a). Using the same analysis as Strassen's algorithm, we find that the algorithm runs in time $O(n^{\log_2 5})$.

   What is wrong with the algorithm? *12p*

3. A sequence is *palindromic* if it is the same whether read left to right or right to left. For example the sequence

$$A, C, G, T, G, T, C, A, A, A, A, T, C, G$$

   contains palindromic subsequences A, C, G, C, A and A, A, A, A, as well as many others. (Note that the symbols in a subsequence do not need to appear contiguously in the longer sequence. Also note that the length of a palindromic sequence may be either odd or even.) Devise an algorithm that takes as input a sequence $x[1], x[2], \ldots, x[n]$ and returns the longest palindromic subsequence in it. The running time of your algorithm should be $O(n^2)$. For the present problem it suffices to determine the *length* of the longest palindromic subsequence, the actual sequence does not need to be presented. However, please justify the runtime of your algorithm. (*Hint:* Denote $P(i, j)$ = length of longest palindromic subsequence in $x[i..j]$.) *15p*

4. For the upcoming Student Guild annual gala dinner and party, n students $s_1, \ldots, s_n$ need to be assigned to n tasks $t_1, \ldots, t_n$, so that every task has a responsible person, and no student has two tasks. The constraints on who can take care of what are represented as an $n \times n$ matrix OK, where

$$\mathrm{OK}(i, j) = \begin{cases} 1, & \text{if student } s_i \text{ can do task } t_j, \\ 0, & \text{otherwise.} \end{cases}$$

   Design an efficient algorithm that finds a feasible allocation of students to tasks, if such exists, or indicates that not all the constraints given by matrix OK can be satisfied. You do not need to describe the details of your algorithm, it suffices to indicate the general algorithmic principles. Give a justified runtime estimate for your algorithm as a function of n. (The runtime should of course be polynomial in n.) *15p*