EXAMINER : Dr Michele Zito
DEPARTMENT : Computer Science Tel. No. 43705

THE UNIVERSITY
of LIVERPOOL

JANUARY 2003 EXAMINATIONS

Bachelor of Arts : Year 3
Bachelor of Science : Year 3

EFFICIENT SEQUENTIAL ALGORITHMS

TIME ALLOWED : Two Hours and a Half

INSTRUCTIONS TO CANDIDATES

Credits will be given for the best **FOUR** answers only.
Each Question is marked out of 25

If you attempt to answer more than the required number of questions (in any section), the marks awarded for the excess questions will be discarded (starting with your lowest mark).

THE UNIVERSITY
of LIVERPOOL

1. (a) State the two properties a greedy algorithm must have in order to solve optimally a given optimisation problem. **(6 marks)**

   (b) The travelling salesman problem (TSP) is defined as follows: a set of n cities is given $C = \{c_1, \ldots, c_n\}$ and for each pair of cities $\text{dist}(c_i, c_j)$ is the distance between city i and city j. A solution is a function $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ defining the order in which the cities are to be visited by the salesman. The cost of a solution is $\sum_{i=1}^{n} \text{dist}(c_{\pi(i)}, c_{\pi(i)+1})$. The aim is to find a minimum cost tour.

   Describe (using pseudo-code) a greedy algorithm that returns a solution for any given TSP instance. **(8 marks)**

   (c) Run the algorithm developed in the previous question on the following instance:

	Athens	Calcutta	Cork	London	Moscow	New York	Rome	Sidney
Athens	0	5000	2600	2000	2838	5200	1800	8000
Calcutta	5000	0	7300	6800	3500	9000	4055	2098
Cork	2600	7300	0	543	2391	2100	1690	8932
London	2000	6800	543	0	1800	2400	1400	8412
Moscow	2838	3500	2391	1800	0	3500	2600	7673
New York	5200	9000	2100	2400	3500	0	2824	10000
Rome	1800	4055	1690	1400	2600	2824	0	8500
Sidney	8000	2098	8932	8412	7673	10000	8500	0

   List the ordered sequence of cities returned by your algorithm. **(6 marks)**

   (d) Prove that your algorithm is not optimal by describing an instance on which it fails to return an optimal solution. **(5 marks)**

2. (a) Compute the total number of character-to-character comparisons performed by the brute force pattern matching algorithm for the pattern $P \equiv$ abccba and the text $T \equiv$ babccbababccbabcabbababc. **(4 marks)**

   (b) Define the string matching automaton for the pattern P in the previous question using the algorithm COMPUTE-TRANSITION-FUNCTION. **(9 marks)**

   (c) Describe how would you check that a string is a suffix of another string. **(5 marks)**

   (d) Would the algorithm COMPUTE-TRANSITION-FUNCTION perform fewer character-to-character comparisons on the given pattern P than the brute force pattern matching algorithm when run on P and T of part (a)?

   If you have solved the previous question you may assume that the check for "P_k suffix of $P_q x$" is performed using the algorithm developed as an answer to that question. Otherwise assume the check "P_k suffix of $P_q x$" costs $\min(|P_k|, |P_q x|)$ comparisons. **(2 marks)**

   (e) Use the algorithm FINITE-AUTOMATON-MATCHING and the automaton computed in part (b) to find the matchings of P in T. Your answer should contain the text T and the sequence of states the automaton is in when reading each character of the text. **(5 marks)**

3. (a) The *polar angle* of a point p with respect to an origin point p_0 is the angle of the vector $p - p_0$ in the usual polar coordinate system. For example, the polar angle of $(3, 5)$ with respect to $(2, 4)$ is the angle of the vector $(1, 1)$, which is $\pi/4$ radians. Write pseudocode to sort a sequence $(p_1, p_2, \ldots, p_n)$ of n points according to their polar angles with respect to a given origin point p_0. (Hints: start from your favourite sorting routine, define an appropriate $\leq$ relation, use cross-product). **(12 marks)**

   (b) Run GRAHAM-SCAN algorithm on the following list of points:

$$(0.22, 0.13), (0.53, 0.82), (0.08, 0.98), (0.57, 0.51), (0.24, 0.66),$$
$$(0.23, 0.68), (0.24, 0.89), (0.76, 0.89), (0.78, 0.69).$$

   List all intermediate stack configurations. **(13 marks)**

4. (a) State Hall's theorem. **(4 marks)**

   (b) Give a pseudo-code description of the matching algorithm implicit in the proof of Hall's theorem. **(6 marks)**

   (c) Use the algorithm referred to in part (b) to find a maximum matching in the bipartite graph $G = (V_1, V_2, E)$ with $V_1 = \{1, 2, 3, 4, 5\}$, $V_2 = \{6, 7, 8, 9, 10, 11, 12\}$, and

$$E(G) = \{\{1,7\}, \{1,8\}, \{1,10\}, \{2,6\}, \{3,7\}, \{3,8\}, \{3,12\}, \{4,6\}, \{4,9\},$$
$$\{4,11\}, \{5,9\}, \{5,11\}\}$$

   In your answer you should list the sequence of subproblems considered by the recursive algorithm and the list of edges in the matching returned by such algorithm. **(15 marks)**

5. Multiprocessor scheduling.

   (a) Define precisely the multiprocessor scheduling problem.
   Your answer should specify the four components of any optimisation problem in the case of multiprocessor scheduling. **(4 marks)**

   (b) Describe the list scheduling approximation algorithm for the multiprocessor scheduling problem. **(5 marks)**

   (c) Describe why the algorithm above is a $2 - 1/m$-approximation algorithm. **(6 marks)**

   (d) Define the term *polynomial time approximation scheme*. **(4 marks)**

   (e) Describe how the list scheduling approximation algorithm can be used to derive a polynomial time approximation scheme for the multiprocessor scheduling problem. **(6 marks)**