Examination in
SIF8010 Algorithms and Data Structures
Monday July 31 2000, 0900-1500

Contact during the examination: Arne Halaas, ph. 73 593442.
Tools: All types of calculators allowed. All printed and handwritten material allowed.
Answers: All answers must be given in the boxes. Do not submit extra pages.
Requirements: It is required that the student "passes" in both the section with ordinary questions (1-4) and the exercise-related question (5).
Remember: Enter your student number in the heading of each page.

Problem 1. (20%)

Give your evaluation of the following 10 statements. Explain your reasoning.

(a) $f(n) = 5n^2 - 64n + 256 = \Omega(n^2)$

Let S_1, S_2, S_3 be statements with running times $O(f_i(n))$, i = 1,2,3.

The following statements (b)-(j) apply to the compound statement

S: **If** S_1 **then** S_2 **else** S_3

(b) S is $O(\max(f_1(n), f_2(n), f_3(n)))$

(c) S is $O(\max(f_1(n)+f_2(n), f_1(n)+f_3(n)))$

(d) S is $O(\max(f_2(n), f_3(n)))$

(e) S is $\Omega(f_1(n))$

(f) S is $\Omega(\max(f_2(n), f_3(n)))$

(g) S is $\Omega(\min(f_2(n), f_3(n)))$

(h) S is $\Omega(\min(f_1(n), f_2(n), f_3(n)))$

(i) S is $\Omega(\min(f_1(n)+f_2(n), f_1(n)+f_3(n)))$

(j) S is $\Theta(f_2(n))$ if $f_2(n) = f_3(n)$

Answer: (Strike out "Yes" or "No". You must explain your reasoning. Each point counts 2 %)

a) Yes/no	Explanation
b) Yes/no	Explanation:
c) Yes/no	Explanation:
d) Yes/no	Explanation:

e) Yes/no Explanation:

f) Yes/no Explanation:

g) Yes/no Explanation:

h) Yes/no Explanation:

i) Yes/no Explanation:

j) Yes/no Explanation:

Problem 2. (20%)

Assume that you have analyzed the algorithm A for your program P, and concluded that the worst-case running time for P is O(f(n)).
You are uncertain about whether the analysis of A is correct, and you have therefore measured the running time of P, T(n), for several different, steadily increasing values of n.

(a) How would you proceed to find out if your worst-case analysis of A was correct?

Answer: (20 %) Explain your method concisely, in steps:

Problem 3. (15%)

Assume that you are to sort N persons after their age, written on the form dd.mm.yyyy, for instance 31.07.2000, 03.04.1979. The age is part of a big data object associated with each person. The persons are to be sorted in increasing order by their age, and it is vitally important to be able to do this as efficiently as possible.

(a) Which method would you suggest? (Discuss your choice if the answer is not simple.)

Answer: 10%

(b) Find the time complexity of the method(s) suggested in **(a)**.

Answer: 5% (Explain your reasoning.)

Problem 4. (20%)

A directed graph G=(V,E) is called "unipathic" if, for every pair of nodes u,v ϵ V there is at most 1 path from u to v. An edge in E can have a positive or negative weight.

(a) Construct an efficient algorithm to find the shortest path from a source node s to all the other nodes v ϵ V for a unipathic graph. The algorithm must detect whether G contains cycles with a negative total weight, and report this.

Answer (10%): Algorithm:

(b) Find the most useful O-time complexity of the method suggested in (a).

Answer: 10% (Explain your reasoning.)

Problem 5. (25%)

(This problem is related to the exercise material.)

Assume that we have a set of points in the plane. We want to group these into clusters of points that are close to each other.

In our analysis we are not interested in the exact position of each point, only the distances between them. We therefore represent the problem as a *complete graph* where each point is a node, and an edge between two nodes is marked with the distance between the corresponding points.

(a) We want to find out which points are closer to each other than a given threshold, that is, we want to remove all edges whose weights are larger than this threshold. What is the running time for the algorithm, expressed as a function of the number of points? Explain your reasoning.

Answer: 5%

(b) We decide to try another approach. We want to change the data structure from **(a)** so that only some of the edges remain. We require the following of our solution (the changed graph):

1) If we can find a path (via at least one other point, *z*) from point x to point y in our solution, then we do not want to include a direct edge from x to y.

2) we must be able to find a path from any given to any of the other nodes;

3) the sum of the distances of the edges we keep must be minimal.

What kind of data structure is this? How would you solve the problem? What is the running time? (Express the answer as a function of the number of points.) Explain your reasoning.

Answer: 5%

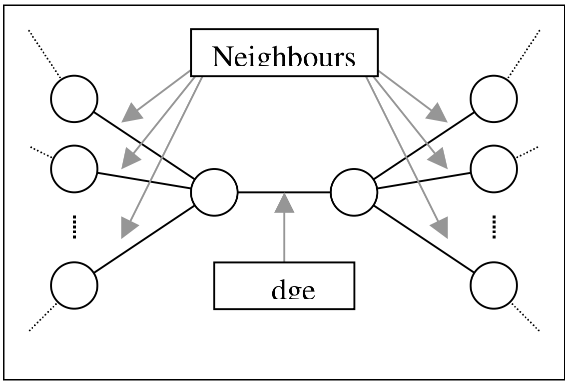
(c) After building the new data structure in **(b)** we are now ready to find the "clusters" of close points. We use the threshold-method from **(a)**, that is, we remove all edges in the new data structure that have edges longer than the threshold. What is the running time for this operation, expressed as a function of the number of points? Explain your reasoning.

Answer: 5%

(d) Does the method in **(c)** result in any time savings, relative to **(a)**, given that one first have to perform the transformation in **(b)**? Explain your reasoning.

Answer: 5%

We now want to use a more advanced cluster analysis. To find out whether an edge is to be "cut" or not, we also consider the neighbour edges (see figure 1) to see if the edge is "abnormally" long.



Figur 1

(e) Will the transformation in **(b)** result in any time savings if we for each edge must consider each of its neighbour edges (as mentioned above)? Explain your reasoning.

Answer: 5%