Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Institut für Theoretische Informatik
Peter Widmayer
Tobias Pröger
Thomas Tschager

Exam
Datenstrukturen und Algorithmen
D-INFK

January 21, 2015

Last name, first name: _____

Student number: _____

With my signature I confirm that I was able to participate in the exam under regular conditions, and that I read and understood the notes below.

Signature: _____

Please note:

- You may not use any accessories except for a dictionary and writing materials.
- Please write your student number on **every** sheet.
- **Immediately** report any circumstances that disturb you during the exam.
- Use a new sheet for every problem. You may only give one solution for each problem. Invalid attempts need to be clearly crossed out.
- Please write **legibly** with blue or black ink. We will only grade what we can read.
- You may use algorithms and data structures of the lecture without explaining them again. If you modify them, it suffices to explain your modifications.
- You have 180 minutes to solve the exam.

Good luck!

Student number: _____

problem	1	2	3	4	**Σ**
max. score	14	13	11	10	48
$\sum$ score					

Problem 1.

Please note:

1) In this problem, you have to provide **solutions only**. You can write them right on this sheet.
2) If you use algorithms and notation other than that of the lecture, you need to **briefly** explain them in such a way that the results can be understood and checked.
3) We assume letters to be ordered alphabetically and numbers to be ordered ascendingly, according to their values.

1 P a) Perform two iterations of *Selection sort* on the following array. The array has already been sorted by previous iterations up to the double bar.

1	2	4	6 ‖	11	9	20	7	15	12	14	8
1	2	3	4	5	6	7	8	9	10	11	12

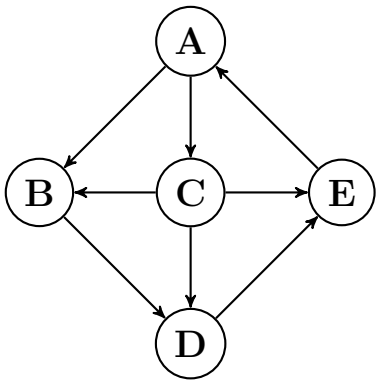
1	2	3	4	5	6	7	8	9	10	11	12

1	2	3	4	5	6	7	8	9	10	11	12

1 P b) Insert the keys 8, 10, 15, 9, 17 in this order into the hash table below. Use open hashing with the hash function $h(k) = k \bmod 7$, and resolve collisions with quadratic probing.

0	1	2	3	4	5	6

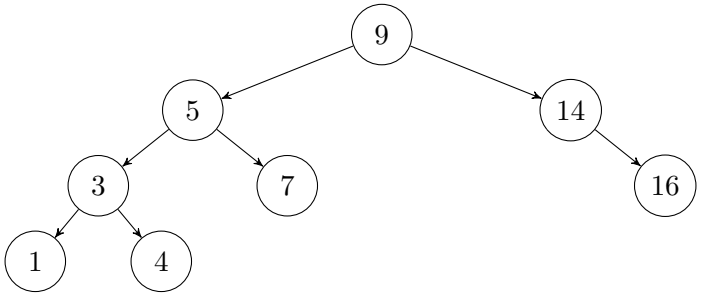
1 P c) Provide a sequence in which the vertices of the following graph are visited during a breadth-first search (BFS) starting at A, and a sequence in which the vertices are visited during a depth-first search (DFS) starting at A. The neighbors of a vertex are visited in alphabetical order.



BFS: _____

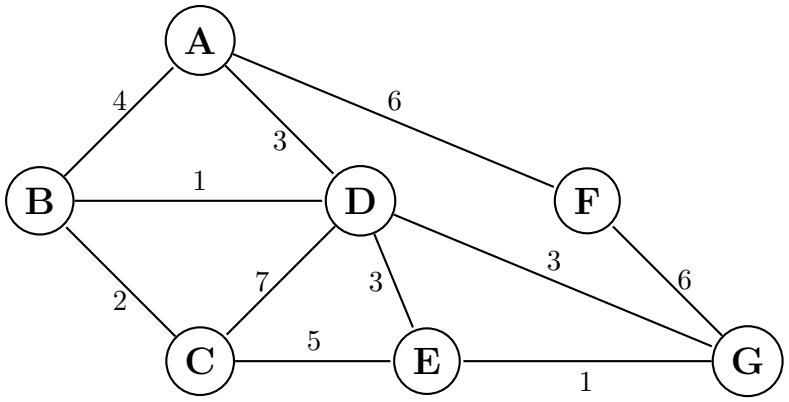
DFS: _____

1 P d) Insert the key 2 into the following AVL tree. *After that*, remove the key 14 from the resulting tree.



After insertion of 2:	After deletion of 14:

1 P e) Mark the first *three* edges in the following graph that are added to the minimal spanning tree by the algorithm of Jarník, Prim, and Dijkstra *starting from vertex A*.



1 P f) Draw the binary search tree that has postorder traversal 4, 7, 6, 10, 11, 9.

2 P g) Given a weighted graph G. Prove or disprove the following statements:

1. Let $P = \langle v_1, \ldots, v_k \rangle$ and $Q = \langle w_1, \ldots, w_l \rangle$ be shortest paths with $v_k = w_1$. Then, $\langle v_1, \ldots, v_k = w_1, \ldots, w_l \rangle$ is a shortest path from v_1 to w_l.
2. Let P be a shortest path from u to v and w be a internal vertex of P. The subpath of P from u to w is a shortest path from u to w. Likewise, the subpath from w to v is a shortest path in G.

1 P h) Specify an **order** for the functions below such that the following holds: If function f is left of function g, then $f \in \mathcal{O}(g)$.

Example: The three functions n^3, n^7, n^9 are already in a correct order, since $n^3 \in \mathcal{O}(n^7)$ and $n^7 \in \mathcal{O}(n^9)$.

- $\binom{n}{4}$
- $\log^2(n)$
- $n \cdot \sqrt{n}$
- $n!$
- $\log(n^5)$
- 7^{13}
- $\log(n^n)$
- $\sqrt{6^n}$

3 P i) Consider the following recursive formula:

$$T(n) := \begin{cases} 15 + 4T(n/4) & n > 1 \\ 1 & n = 1 \end{cases}$$

Specify a closed (i.e., non-recursive) form for $T(n)$ that is *as simple as possible*, and prove its correctness using mathematical induction.

Hints:

(1) You may assume that n is a power of 4.

(2) For $q \neq 1$, we have $\sum_{i=0}^{k} q^i = \frac{q^{k+1}-1}{q-1}$.

1 P j) Specify (as concisely as possible) the asymptotic running time of the following code fragment in Θ notation depending on $n \in \mathbb{N}$. You do not need to justify your answer.

```
1 for(int i = 1; i <= n; i += 3) {
2         for(int j = n; j > 1; j = j/3)
3                 ;
4         int k = 1;
5         while(k*k <= n)
6                 k = k + 2;
7 }
```

1 P k) Specify (as concisely as possible) the asymptotic running time of the following code fragment in Θ notation depending on $n \in \mathbb{N}$. You do not need to justify your answer.

```
1 for(int i = n; i > 0; i -= 1) {
2         for(int j = 0; j < i; j += 1) {
3                 ;
4         }
5 }
```

Problem 2.

A company got the job to produce a cable of length at least L. Since many earlier produced cable pieces are available in the warehouse, the company decides to build the requested cable by putting some of these pieces together. There exist n cable pieces where piece i has length l_i. If two cable pieces of lengths l_i and l_j are put together, we obtain a cable of length $l_i + l_j$. To avoid the resulting cable to be unnecessarily long, the goal is to produce a cable whose length is minimum among all possibilities to produce a cable of length $\geq L$. You can assume that the warehouse contains sufficiently many cable pieces, i.e. that the sum of the lengths of the cable pieces is at least L.

Example: We want to produce a cable of length $L = 6$, and the warehouse contains cable pieces with lengths $l_1 = 3$, $l_2 = 4$ and $l_3 = 5$. The best choice of cable pieces is $\{1, 2\}$, because these two cable pieces together have overall length 7. Notice that the choices $\{2, 3\}$ and $\{1, 2, 3\}$ also lead to a cable of length ≥ 6, but they are not optimal since the lengths of the produced cables is 9, or 12, respectively. Thus, we do not want to output them.

9 P a) Provide a dynamic programming algorithm for computing the minimum length of a cable that can be obtained by putting suitable cable pieces from $\{1, \ldots, n\}$ together, and that has overall length at least L. For the example above, the algorithm should return 7. Address the following aspects in your solution.

1) What is the meaning of a table entry, and which size does the DP table have?

2) How can an entry be computed from the values of previously computed entries?

3) In which order can the entries be computed?

4) How can the value of the minimum cable length be obtained from the DP table?

Hint: The trivial algorithm that simply inspects all possible solutions does **not** give any points, because it is not a dynamic programming algorithm.

2 P b) Describe in detail how you can recognize from the DP table which cable piece is contained in an optimal solution.

2 P c) Provide the running time of the algorithm developed in a) and in b), and justify your answer. Is the running time polynomial?

Problem 3.

A hospital wants to determine which doctor works on which day. This exercise is concerned with planning the daily working times during vacation periods. Let T be the set of all vacation days. There exist k vacation periods $\{1, \ldots, k\}$ consisting of one or more contiguous vacation days $T_j \subseteq T$ (for $j \in \{1, \ldots, k\}$). Every day in T is contained in exactly one vacation period, i.e., we especially have $T_i \cap T_j = \emptyset$ for $i \neq j$. In the hospital there are n doctors. Every doctor i specifies a set of vacation days $S_i \subseteq T$ on which he could be present. The goal is to design an operational plan that distributes the vacation days T among the doctors such that on every day there is exactly one doctor present. Additionally we require that no doctor works on more than $C \in \mathbb{N}$ many vacation days in overall. The problem now consists of deciding whether, for a given C, there exists an operational plan satisfying the above conditions, and to efficiently compute how such a plan looks like.

Example: Assume that C was set to 2, and that we had the following doctors and vacation periods:

i	Doctor	Possible days S_i
1	Dr. Cuddy	$\{24.12, 25.12, 26.12\}$
2	Dr. Foreman	$\{24.12\}$
3	Dr. House	$\{1.8\}$
4	Dr. Wilson	$\{1.8, 24.12\}$

j	Description	Associated days T_j
1	Christmas	$\{24.12, 25.12, 26.12\}$
2	Swiss National Day	$\{1.8\}$

Now there exist different possibilities: for example, Dr. Cuddy could be present on 25. and 26.12, Dr. House on 1.8 and Dr. Wilson on 24.12. If C was set to 1, then there would be no suitable operational plan.

4 P a) Model the above problem as a flow problem. For this purpose, describe the construction of an appropriate network $N = (V, E, c)$ with the vertex set V and the edge E, and describe which capacities the edges should have. How can you deduce from the value of a maximum flow whether an appropriate assignment exists, or not?

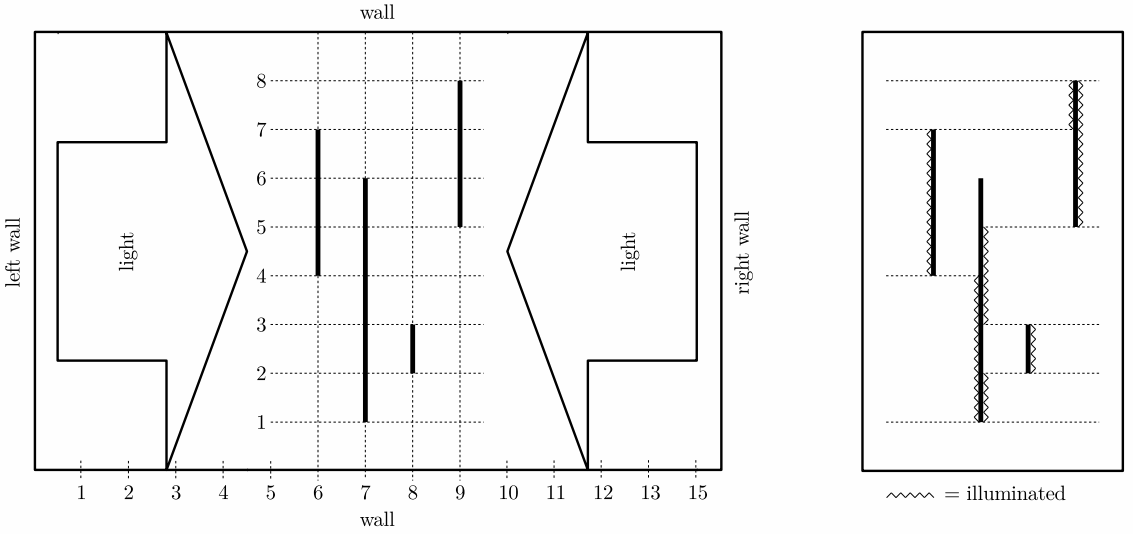
2 P b) Specify an algorithm that solves a) as efficient as possible. Provide the running time of the above-mentioned algorithm in dependency of the number of doctors n and the number of vacation days $m = |T|$. Justify your answer.

3 P c) Suppose that we already solved the flow problem described in a), i.e., we know the value of flow ϕ_e for each edge e, and an operational plan as described above really exists. Describe in detail an algorithm that uses the ϕ_e values and that computes such an operational plan. What is the running time of your algorithm if every ϕ_e can be accessed in time $\Theta(1)$?

2 P d) The current approach has the disadvantage that it might generate operational plans where some doctors have a lot of work while others have only very little. Thus, we now want to find an operational plan that satisfies the above conditions, and that, for every $j \in \{1, \ldots, k\}$, assigns at most one day of T_j to every doctor. For the christmas vacation period of the above example this means that a doctor has to present either on 24.12, or on 25.12, or on 26.12, or even not at all. Describe how you can modify the network constructed in a) to satisfy this additional requirement.

Problem 4.

An artist draws a ground plot of his artwork consisting of plates that are illuminated with lasers from both sides.



There are n plates with different positions and widths. The i-th plate is represented by a triple $P_i = (x_i, y_i, b_i)$, where x_i is the distance to the left wall, y_i is the distance to the lower wall in the picture above, and b_i is the width of the plate. For example, a possible input for the above ground plot is $P_1 = (7, 1, 5)$, $P_2 = (9, 5, 3)$, $P_3 = (8, 2, 1)$, and $P_4 = (6, 4, 3)$.

The plates are illuminated from extensive laser light sources at the left and the right wall. The plates absorb all incident light. As the plates are heated by the light, they need to be cooled proportionally to the amount of incident light. Therefore, the artist wants to compute the illuminated area (i.e. the sum of the area illuminated by the light source on the left wall and the light source on the right wall) *of each plate*. Instead of computing the illuminated area it is sufficient to compute the illuminated width, because all plates have the same height. Therefore, it is sufficient to consider the two-dimensional problem as shown in the above sketch.

For each plate i the total illuminated width B_i, i.e. the sum of the width illuminated from the left and the width illuminated from the right, should be computed (cf. the right sketch above). The result for the above instance is $B_1 = 6$, $B_2 = 4$, $B_3 = 1$, and $B_4 = 3$.

10 P Design an efficient sweepline algorithm for the above problem. Address the following aspects in your solution.

1) In which direction is the sweepline moving, and what are the stopping points?
2) Which objects have to be stored in the sweepline data structure, and what is an appropriate choice for it?
3) What happens if the sweepline encounters a new stopping point?
4) How can the illuminated width B_i of each plate i be computed?
5) Provide the running time of your algorithm depending on n, and justify your answer.

Hint: For the sake of simplicity we assume that no two plates are positioned directly on the top of each other and that no two plates start and end directly next to each other. Note that (as in the above example) the plates are not necessarily ordered by their x-coordinates.