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
Thomas Tschager

Exam
Datenstrukturen und Algorithmen
D-INFK

January 27, 2016

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.
- If not explicitly asked for, 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	17	11	8	12	48
$\sum$ score					

Problem 1.

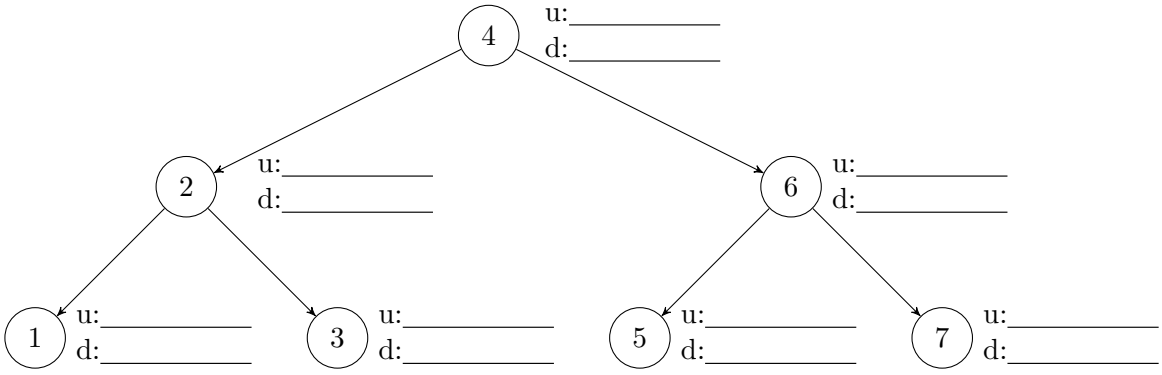
Please note:

1) In this problem, you have to provide **solutions only**. You can write them directly 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) Insert the keys 18, 4, 17, 7, 15, 29 in this order into the hash table below. Use double hashing with the hash function $h(k) = k \bmod 11$. Resolve collisions using $h'(k) = 1 + (k \mod 9)$ for probing (to the left).

0 1 2 3 4 5 6 7 8 9 10

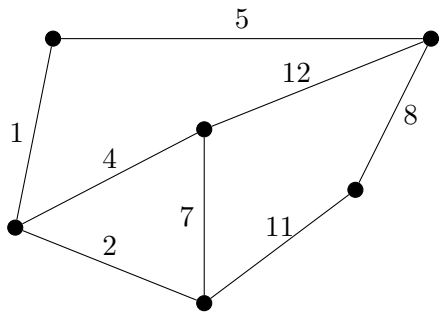
1 P b) Insert the closed intervals $[1,3], [1,5], [4,6], [2,4], [2,2]$ and $[1,2]$ in the following interval tree. Mark all nodes that are visited in a query for $x = 2.5$.



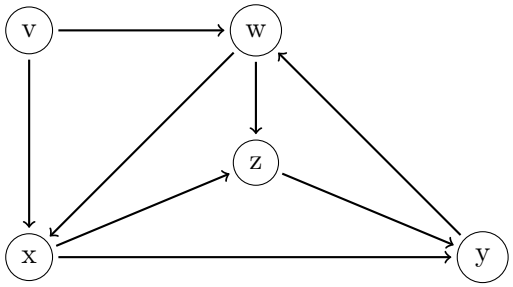
1 P c) Consider the following self-organizing list. Provide the number of key comparisons when the elements C, B, D and B are accessed in this order using the Move-to-Front rule.

$$A \to B \to C \to D \to E$$

1 P d) Mark in the following graph the first edge that is visited by the algorithm of Kruskal, but ***not*** added to the minimum spanning tree.



1 P e) Given the following graph $G = (V, E)$, provide a set $E' \subset E$ of smallest cardinality, such that $G' = (V, E \setminus E')$ can be sorted topologically. Provide also a topological ordering for G'.



$E' =$ _____

topological ordering for G':

_____

1 P f) Draw a connected graph on six vertices that has exactly two perfect matchings.

1 P g) The following array contains the elements of a min-heap stored in the usual fashion. Specify the array after the minimum has been removed and the heap condition has been reestablished.

2	8	10	9	12	15	11	13	14
1	2	3	4	5	6	7	8	9

1	2	3	4	5	6	7	8	9

1 P h) Draw an AVL tree with five vertices and the keys stored in these vertices, such that the insertion of a sixth key leads to a left-right double rotation. Indicate which key has to be added and draw the resulting AVL tree.

Before the insertion of the 6th key:	After the insertion of the 6th key:

Key that has to be inserted:_____

3 P i) Consider the following recursive formula:

$$T(n) := \begin{cases} 4T(n/4) + n + 6 & n > 1 \\ 1 & n = 1 \end{cases}$$

Specify a closed (i.e., non-recursive) form for $T(n)$ that is *as simple as possible*, and prove its correctness using mathematical induction. You may assume that n is a power of 4.

Hint: 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 = 2·i)
2         for(int j = n; j >= 0; j = j−1)
3                 ;
```

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

1 P l) 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*i < n; i = i+2) {
2         for(int j = 1; j/2 <= n; j = j + 4)
3                 ;
4 }
```

1 P g) 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 $n^7 \in \mathcal{O}(n^9)$.

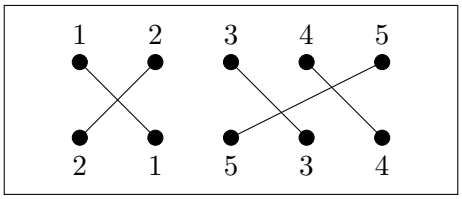
$$\log(n^{4n}),\ \binom{n}{5},\ \sqrt{16^n},\ \frac{n}{\log(n)},\ n+n^3,\ n!,\ 1052^{24}$$

2 P n) For each of the following statements, mark with a cross whether it is true or false. Every correct answer gives 0.5 points, for every wrong answer 0.5 points are removed. A missing answer gives 0 points. In overall the question gives at least 0 points. You do not have to justify your answers.

A binary search tree is balanced.	☐ True	☐ False
Let $G = (V, E)$ be a connected graph. Every spanning tree of G has exactly $\|V\| - 1$ edges.	☐ True	☐ False
Selection sort and insertion sort are in-situ *sorting algorithms.*	☐ True	☐ False
If you only insert keys into an empty binary search tree (and no key is removed), then the insertion order corresponds exactly to the preorder traversal.	☐ True	☐ False

Problem 2.

Problem definition. $2n$ components are placed on a circuit board in two layers with n components each. Every component of the upper layer is assigned to exactly one component of the lower layer and vice versa. These pairs of components should be connected by straight circuit paths. Develop an algorithm that computes how many components can by connected at most if no two circuit paths are allowed to intersect.



Example: In the figure above 10 components are placed on a circuit board. Components with the same numbering should be connected. In this example, there exist two solutions in which three pairs of components can be connected:

(i) components with numbers $1, 3$ and 4. (ii) components with numbers $2, 3$ and 4.

7 P a) Provide a dynamic programming algorithm that is as efficient as possible asymptotically for computing the maximum number of circuit paths that do not intersect pairwise. Address the following aspects in your solution.

1) What is the meaning of a table entry, and what size does the DP table have?
2) How can an entry be computed?
3) Describe the order of the computation steps.
4) How can the maximum number of non-intersecting circuit paths be obtained from the DP table?

Hint: The trivial algorithm that simply enumerates all possible solutions does **not** give any points (as is it not efficient).

2 P b) Describe in detail how the algorithm can be modified such that it additionally computes a set of non-intersecting circuit paths of maximum cardinality.

2 P c) Provide the running time of the algorithms developed in a) and in b), and justify your answer.

Problem 3.

6 P a) There are n assistants and m exercise groups that take place at different times. Every assistant specifies which exercise groups he can supervise. An assistant can supervise at most one exercise group. Develop an algorithm to compute how many exercise groups can be supervised by the assistants.

Example: An assignment of $n = 3$ assistants to $m = 3$ exercise groups should be computed (cf. table). In this example, every exercise can be supervised by an assistant: Assistant 1 supervises group 1, assistant 2 group 3 and assistant 3 group 2.

Assistant	*exercise groups*
assistant 1	group 1 or 3
assistant 2	group 3
assistant 3	group 1 or 2

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 edge set E, and describe which capacities c the edges should have. Name an efficient algorithm for computing the maximum flow in N. What is the time complexity of this algorithm depending on n and m? How can you deduce from the value of a maximum flow how many exercise groups can be supervised?

2 P b) Now you also need to assign rooms to the exercise groups, where k rooms are available. For every exercise group it is given, which rooms are available. Compute an assignment of as many assistants as possible to exercise groups and rooms, such that no room is occupied multiple times.

Example: An assignment for $n = 3$ assistants, $m = 3$ exercise groups and $k = 4$ rooms should be computed:

assistant	*exercise groups*
assistant 1	group 1 or 3
assistant 2	group 2
assistant 3	group 2

exercise group	*available rooms*
group 1	room 1
group 2	room 1
group 3	room 2, 3 or 4

In this example, not all groups can be supervised: For example, assistant 1 can supervise group 3 in room 1 and assistant 2 group 3 in room 4. Group 1 cannot be supervised in this case, because room 1 is already occupied by group 3.

Model this extended assignment problem as a flow problem. Describe the construction of an appropriate network $N' = (V', E', c')$ with the vertex set V' and edge set E', and describe which capacities c' the edges should have. How can you deduce from the value of a maximum flow how many exercise groups can be supervised?

Problem 4.

An arrangement of n roofs is represented by a set of n non-intersecting line segments: A roof is given as a line segment starting and ending at coordinates (x_l, y_l) and (x_r, y_r). All x-coordinates and all y-coordinates are pairwise different. It rains evenly everywhere and raindrops fall vertically on some roofs (i.e. those roofs that are not sheltered by other roofs). The rain flows to the lower ends of these roofs. Then it trickles vertically to the next roof or to the floor. Depending on the arrangement of the roofs some roofs get wet and others remain dry. You should determine the dry roofs.

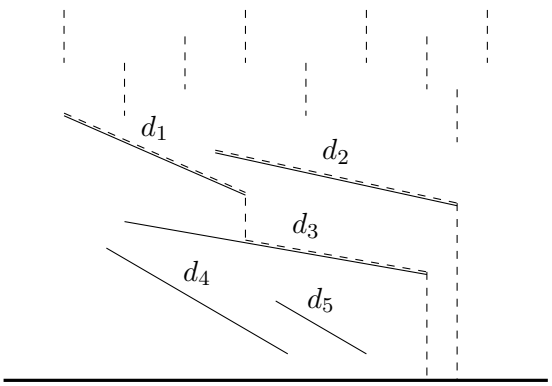
9 P a) All roofs are sloped to the right, i.e. for every roof it holds that $y_r < y_l$. Thus, the right end is also the lower end for every roof. Design an efficient scanline algorithm for computing the dry roofs. Address in particular the following aspects in your solution.

1) In which direction is the scanline moving, and what are the stopping points?
2) Which objects have to be stored in the data structure, and what is an appropriate choice for it?
3) What happens if the scanline encounters a new stopping point?
4) How can the set of dry roofs be obtained?
5) Provide the running time of the algorithm in dependency of n and justify your answer.

3 P b) Now we consider arbitrarily sloped roofs. For every given roof it holds that $y_l \neq y_r$. Describe how the algorithm can be modified such that it computes the set of dry roofs. Provide the running time of your algorithm depending on n and justify your answer.

Example: In the following figures, rain is depicted as dashed lines. In the left figure for exercise *a*) the roofs d_4 and d_5 remain dry. In the right figure for exercise *b*) only d_4 remains dry.

Exercise a)



Exercise b)