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
Holger Flier, Beat Gfeller

Prüfung
Datenstrukturen und Algorithmen
D-INFK

10. Februar 2010

English version

Family name, First name: _____

Student-Number: _____

With my signature, I confirm that I could take this exam under regular circumstances and that I have read and understood the remarks below.

Signature: _____

Remarks:

- Except for a dictionary, no auxiliary means are allowed.
- Please write your student number on **each** sheet.
- Please notify us **immediately**, if there is any disturbance which has an adverse effect on your work.
- Please use a separate sheet of paper for each task. Only one solution per task can be handed in. Invalid tentative solutions have to be crossed out clearly.
- Please write **legibly** with blue or black ink. We only give points for legibly written text.
- The exam takes 120 minutes. **Don't worry!** We don't expect that anyone can solve all the tasks in the given time! The maximum grade can be achieved with far less than the total of achievable points.

Good luck!

Student-Number: _____

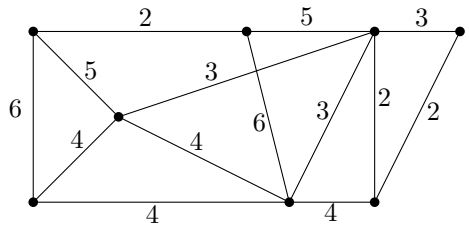
Task	1	2	3	4	5	**Σ**
Possible Points	9	7	9	9	9	43
Points						

Task 1:

Remarks:

1. In this task it suffices to write down **only the results**. These can be written directly below the task descriptions.
2. Provided that you use notations, algorithms and data structures from the lecture "Datenstrukturen & Algorithmen", no explanations or argumentation is required. If you use other methods, you should **briefly** explain your results to make them comprehensible and traceable.
3. We order letters by their alphabetic order, and numbers in increasing value.

1 P a) Mark the edges of a minimum spanning tree in the weighted graph below.



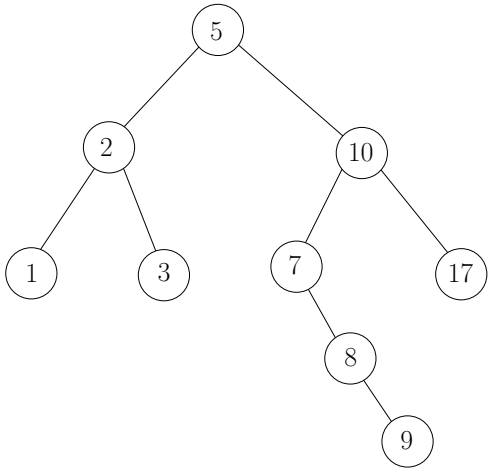
1 P b) Draw the binary search tree for which its preorder traversal results in the sequence $19, 9, 2, 7, 12, 17, 14, 13, 18, 21$.

1 P c) Execute the first pass of the sorting algorithm *natural 2-way merge sort* on the following array.

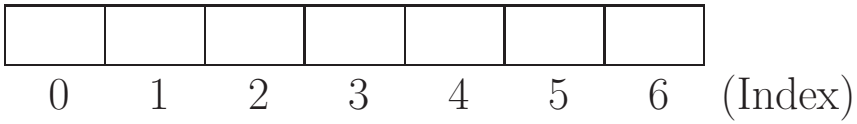
5	12	14	2	6	9	1	11	4	8	7

after the first pass:

1 P d) Draw the splay tree which results from the one shown below after node 9 is accessed (and after the corresponding splay operations have been executed).



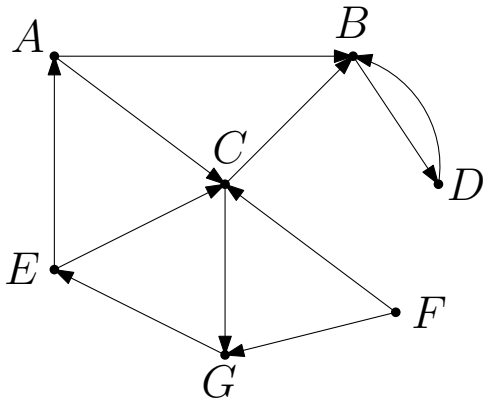
1 P e) Insert the keys $4, 8, 16, 14, 1, 9$ in this order into the following hash table via open hashing, and use double hashing. Use $h(k) = k \mod 7$ as the hash function. For probing, use the hash function $h'(k) = 1 + (k \mod 5)$.



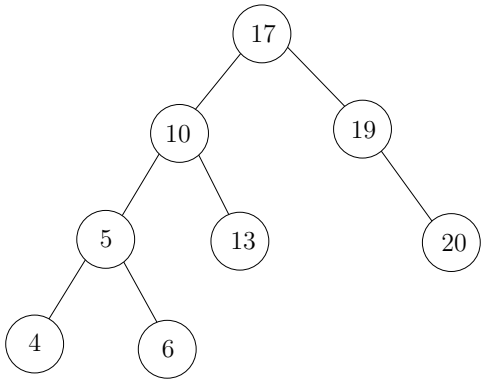
1 P f) The array below contains the elements of a min-heap, saved in the usual order. How will the array look like after the minimum is removed and and the heap property is restored?

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

1 P g) The following directed graph is traversed via depth-first-search. The search starts at node A. Write down a sequence in which the nodes could be traversed.



1 P h) Insert key 9 into the AVL-tree shown below, and delete key 17 from the resulting AVL-tree.



After inserting 9:	After deleting 17:

1 P i) Given are seven keys with their corresponding number of accesses. Using the algorithm of Huffman, generate an optimal coding tree and draw this tree.

Key	a	b	c	d	e	f	g
Number of accesses	6	2	4	3	13	3	12

Task 2:

1 P a) Provide an **order** of the functions given below, which has the following property: If the function f stands left of the function g then it holds that $f \in O(g)$.

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

- n^3
- $n\sqrt{n}$
- $n!$
- $n \log n$
- $3^{\sqrt{n}}$
- $\dfrac{n^2}{\log n}$

3 P b) Consider the following recursive equation:

$$T(n) := \begin{cases} 2T(\frac{n}{4}) + 2n + 1 & n > 1 \\ 3 & n = 1 \end{cases}$$

Give a closed (i.e., non-recursive) expression for $T(n)$ (simplified as far as possible) and prove its correctness using induction.

Hints:
(1) You may assume that n is a power of 4.
(2) For $q \neq 1$ it holds: $\sum_{i=0}^{k} q^i = \frac{q^{k+1}-1}{q-1}$.

1 P

c) Determine the asymptotic running time of the following code, depending on $n \in \mathbb{N}$, expressed in Theta-notation (you don't need to justify your answer):

```
from i := n until i < 2 loop
  from j := i until j < 1 loop
    j := j - 1
  end
  i := i // 2
end
```

1 P

d) Determine the asymptotic running time of the following code, depending on $n \in \mathbb{N}$, expressed in Theta-notation (you don't need to justify your answer):

```
from i := 1 until i > n // 8 loop
  from j := 2*i until j > n loop
    j := j + 1
  end
  i := i + 1
end
```

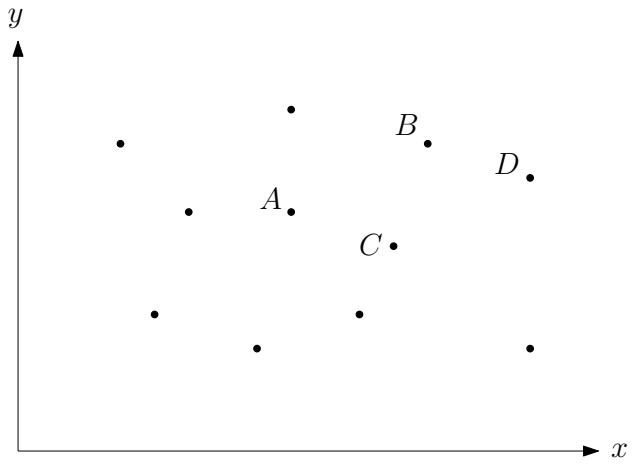
1 P

e) Determine the asymptotic running time of the following code, depending on $n \in \mathbb{N}$, expressed in Theta-notation (you don't need to justify your answer):

```
from i := 1 until i > n loop
  from k := 1 until k * k > n
    k := k + 1
  end
  i := i * 2
end
```

Task 3:

In this task we are looking at sets of points in the plane. We say that a point (x, y) *dominates* another point (x', y') if both $x > x'$ and $y > y'$ hold simultaneously. A point is *dominant*, if it is not dominated by any other point.
For example, in the figure below, point A is dominated by point B, but not by point C. Point D is dominant.



3 P a) Design an algorithm, as efficient as possible, that computes for a given set of n points a list of all dominant points. Describe your algorithm in words and/or pseudocode. Give the running time of your solution for the worst case.

6 P b) Now we consider a dynamic version of the problem solved in a), in which points may be added. No points may be removed, however. Hence, you should design a data structure that supports the following operations as efficiently as possible (with regard to the number of points in the current set):

- `Init`(): creates a data structure that represents an empty set of points
- `Insert`(x, y): adds a point (x, y) to the current set of points
- `ListEfficient`(): returns the set of dominant points in the current set of points

Describe your data structure in words and implement the three operations above in words and/or pseudocode.

Task 4:

In this task we are given a two-dimensional array A with n^2 different numbers, consisting of n rows and n columns. Two elements of the array are *neighbours* if they touch on one side, i.e, element $A[i,j]$ is a neighbour of the four elements $A[i-1,j]$, $A[i,j-1]$, $A[i+1,j]$ and $A[i,j+1]$, if these fields exist (hence, elements on the boundary of the array have less elements as neighbours). A series $x_1, x_2, \ldots, x_k$ of elements of the array is called a *sequence*, if the following conditions are met:

- the series of elements is sorted ascendingly, and
- for each $i \in 1, \ldots, k-1$ the elements x_i and x_{i+1} of the array are neighbours.

3 P a) Design an algorithm, as efficient as possible, that computes a longest sequence in the array which is completely contained in a single row or column. In the array of the example given below, a possible sequence would be $4, 6, 10$. This is not a longest sequence, however, since the sequence $6, 28, 29, 47$ is longer.

Describe your algorithm in words, and give its running time.

6 P b) Now, we are looking for a longest arbitrary sequence in the given two-dimensional array.

In the example given below, a possible sequence would be $4, 6, 28, 29, 47, 49$.

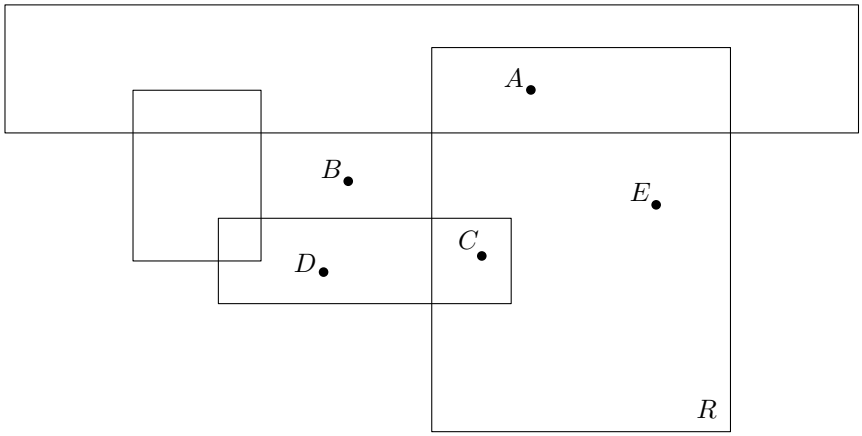
Design an algorithm, as efficient as possible, that finds such a longest sequence. Describe your algorithm in pseudocode, and give its running time. Your pseudocode has to be written in a notation similar to Java, Eiffel, or C++.

Example for an array:

9	27	42	41	48
35	39	8	3	5
12	49	2	38	4
15	47	29	28	6
19	1	25	33	10

Task 5:

We are given a set of points in the plane as well as a set of iso-oriented rectangles. We want to determine if there are points that lie within the area that is covered by the union of the rectangles. In the figure shown below, e.g., points A, C, D, and E lie within this area, whereas point B does not.



In the following, assume that a set of m iso-oriented rectangles and a set of n points are given as input. In particular, each point P_j, $j \in \{1, \ldots, n\}$, is described by coordinates (x_j, y_j) and each rectangle R_i, $i \in \{1, \ldots, m\}$, by its top-left corner (l_i, t_i) and its bottom right corner (r_i, b_i). You may assume that all coordinates are integer and are within the range $\{1, \ldots, m + n\}$.

5 P a) Design an algorithm, as efficient as possible, which decides (i.e., returns either "yes" or "no") whether or not (at least) one of the given points lies within the area covered by the union of the rectangles. Describe your algorithm in words and/or pseudocode.

1 P b) Which running time does your algorithm have in the worst case, depending on m and n?

3 P c) Next, we want to compute the maximum number of points which all lie within the same rectangle. Note that a point may lie within several rectangles at once. In the example above, the number we are looking for is three, since points A, C, and E all lie within rectangle R, and no other rectangle contains more points. Design an algorithm, as efficient as possible, which computes this number, and describe your algorithm in words and/or pseudocode.