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
Yann Disser

Exam
Datenstrukturen und Algorithmen
D-INFK

August 9, 2012

last name, first name: _____

stud.-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 record 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 120 minutes to solve the exam.

Good luck!

stud.-number: _____

problem	1	2	3	4	5	Σ
max. score	9	7	9	9	9	43
Σ 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 your 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) For each of the following data structures, mark with a cross, whether the tree it uses always has logarithmic height (balanced) or not (unbalanced):

(natural) search tree	☐ BALANCED	☐ UNBALANCED
AVL tree	☐ BALANCED	☐ UNBALANCED
optimum search tree	☐ BALANCED	☐ UNBALANCED
heap	☐ BALANCED	☐ UNBALANCED
interval tree	☐ BALANCED	☐ UNBALANCED
segment tree	☐ BALANCED	☐ UNBALANCED
range tree	☐ BALANCED	☐ UNBALANCED

1 P b) Execute one pivoting step of quicksort on the following array (in-situ, i.e., without auxiliary array). Use the leftmost element of the array as pivot.

32	13	19	8	69	34	77	52	17	49	37	26	5
1	2	3	4	5	6	7	8	9	10	11	12	13

1 P c) 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.

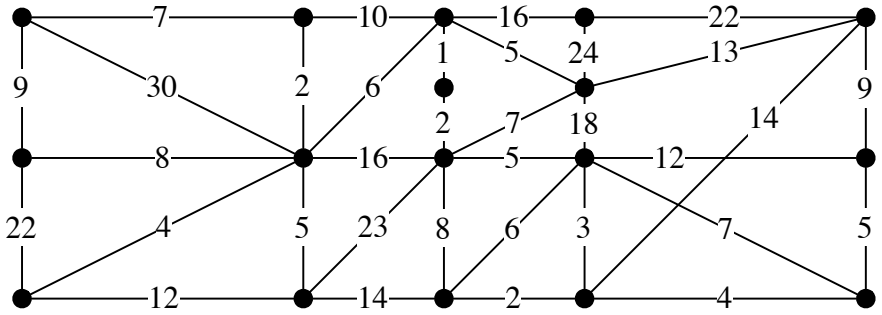
5	7	21	13	45	36	67	22	17	49	87	86	42
1	2	3	4	5	6	7	8	9	10	11	12	13

1 P d) Insert the keys 26,37,33,45,49,29,10,21,7 in this order into the following open-addressing hash table using double hashing. Use the hash function $h(k) = k \mod 13$, and the hash function $h'(k) = 1 + (k \mod 11)$ for probing.

0	1	2	3	4	5	6	7	8	9	10	11	12

1 P

e) Mark the edges of a minimum spanning tree in the following graph:

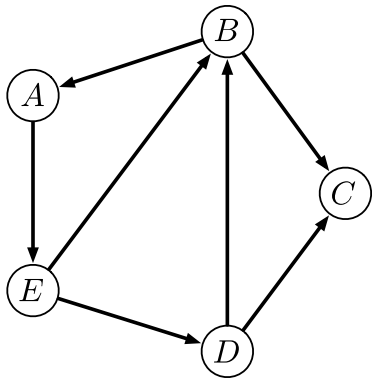


1 P

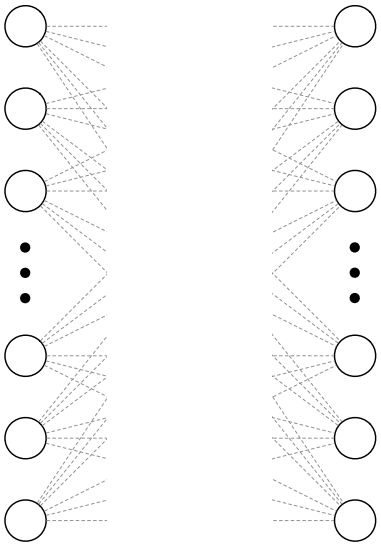
f) How many edges can a directed graph with n nodes have at most, while still having a topological sorting?

1 P

g) Provide a sequence in which the nodes of the following graph can be visited during a breadth-first search starting at A, and a sequence in which the nodes can be visited during depth-first search starting at A.

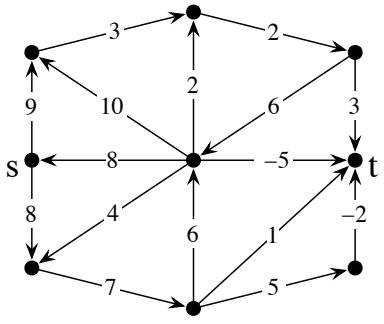


1 P h) We consider any bipartite graph G (the nodes on the left form one set, the nodes on the right form the other):



Add to G a source and a target node and additional edges with capacities, such that an integer flow in the resulting graph uses exactly the edges of a maximum matching in G. For the flow computation, you may assume that all edges of G have capacity 1 and are directed from left to right.

1 P i) Name an algorithm that can be used for finding a shortest path from s to t in the following graph as efficiently as possible. What is the running time of this algorithm in general for graphs with n nodes and m edges?



Problem 2

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

- n^{1000}
- $\log(n^2)$
- $(\log n)^2$
- $n!$
- $1000n$
- $5^{\sqrt{n}}$
- $\sqrt{n}$
- $n^2 + 1000$
- n^n

3 P b) Consider the following recursive formula:

$$T(n) := \begin{cases} 2 + 3T(\frac{n}{7}) & n > 1 \\ 2 & n = 1 \end{cases}$$

Specify a closed form (i.e., non-recursive) 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 7.
(2) For $q \neq 1$, we have $\sum_{i=0}^{k} q^i = \frac{q^{k+1}-1}{q-1}$.

1 P

c) 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.

```
for(int i = 0; i < n; ++i)
  for(int j = 0; j < i/2; ++j)
    ;
```

1 P

d) 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.

```
for(int i = 0; i < n*n; ++i)
  for(int j = 1; j <= i; j *= 3)
    ;
```

1 P

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

```
int f(int n)
{
  if(n <= 0)
    return 1;
  else
    return f(n-1) + f(n-1);
}
```

Problem 3

We consider a game in which coins lie in a row from left to right. Each coin has an integer value, and the initial number of coins n is divisible by 3. In every move, the player either may take (and keep) the leftmost coin or the rightmost coin, but must discard two coins from the other end. The discarded coins are lost for the player. For instance, if the player takes the leftmost coin, he has to discard the two rightmost coins. The goal of the player is to maximize the value of the coins that he takes.

Example: In the following example with $n = 21$, the player can take a total value of at most 803. To do this, he first needs to take one coin from the right end, then one from the left, then two from the right again, and finally three from the left end. Every other strategy (obviously) has the same number of moves ($n/3 = 7$), but yields a smaller total value.

20	20	500	20	20	20	20	100	100	100	5	5	5	5	5	5	1	1	1	1	1

5 P a) Design an algorithm using dynamic programming that, as efficiently as possible, computes the maximum value that the player can take. Describe your dynamic program and provide its running time.

1 P b) Briefly describe how you can adapt your algorithm if it has to not only compute the maximum possible value, but also a sequence of moves that achieves this value. (For the example above, this would be something of the type ,,RLRRLLL“.)

3 P c) We consider an extended version of the game with a player A and an opponent B. The players take turns alternatingly, and A starts the game. Whenever it is player A's turn, he takes the leftmost or the rightmost coin. Player B takes *two* coins in every move, either *both* from the left end or *both* from the right end.

We are interested in a good strategy for player A, independent of what B does. In other words, we want to determine what maximum value player A can **guarantee**, independent of B's strategy.

Describe an efficient algorithm that, given a row of coins, computes the maximum value that player A can always achieve. Provide the running time of your algorithm.

Hint: It is safe to assume that B always does a move that is worst-possible for A. The moves of B aim at *minimizing* the final value, while A's moves try to *maximize* it. In the example above, player A can guarantee a value of at most 632. In particular, player B cannot prevent A from getting the 500 (A can just take from the right end until the 500 becomes available).

Problem 4

You are to design an algorithm for optimizing multi-day bike tours with camping during the nights. A tour consists of n sections and $n+1$ sites, where the i-th section connects the sites i and $i+1$. You are given such a tour, a duration T (in days) of the tour, and a weather forecast for all areas of the tour. The weather forecast describes the precipitation (rain) along each section for every day and the precipitation for each night at each site. An instance with $n = 3$ and $T = 6$ might look as follows:

	section 1	section 2	section 3
day 1	2mm	3mm	1mm
day 2	1mm	4mm	0mm
day 3	3mm	0mm	8mm
day 4	3mm	7mm	5mm
day 5	10mm	1mm	2mm
day 6	5mm	5mm	5mm

	site 1	site 2	site 3	site 4
night $1 \to 2$	0mm	2mm	0mm	0mm
night $2 \to 3$	1mm	0mm	2mm	1mm
night $3 \to 4$	0mm	1mm	15mm	0mm
night $4 \to 5$	2mm	0mm	15mm	4mm
night $5 \to 6$	0mm	4mm	0mm	3mm

We are looking for a schedule that plans which sections should be completed on each day, such that the **sum** of the precipitations for each section and each overnight stay is minimized (according to the forecast). The sections have to be completed in order (i.e., section i before section $i+1$), but *any number of sections can be scheduled for a single day.* Also, *every* night has to be spent at some site, while multiple nights can be spent at the same site. This means that if the first section is planned for day j, then $j-1$ nights have to be spend at site 1. Similarly, $T-j$ nights need to be spent at site $n+1$ if the last section is scheduled for day j.

Example: The best solution for the example above achieves a precipitation of 8mm. To obtain this, first, one night at site 1 has to be planned (0mm), on day 2 section 1 is completed (1mm), then three nights are spent at site 2 (0+1+0=1mm), on day 5 the sections 2 and 3 are completed (1+2=3mm), and, finally, one night needs to be spent at site 4 (3mm). If, instead, we decide to complete all sections on day 1 (2+3+1=6mm), we need to stay at site 4 for five nights (0+1+0+4+3=8mm), and have a total precipitation of 14mm.

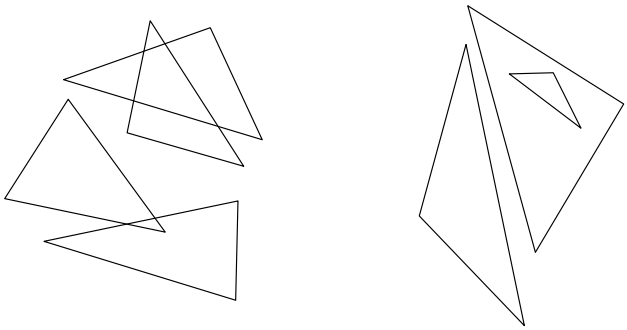
4 P a) Construct a directed graph G, such that a shortest path in G corresponds to a schedule with minimum precipitation. Introduce a node of G for every state of the system and define (directed and weighted) edges that correspond to transitions between states. Describe your construction and derive an efficient algorithm to schedule bike tours with minimum precipitation.

2 P b) What is the running time of your algorithm depending on n and T?

3 P c) Now, assume that no more than three sections can be completed each day. How can you adapt the construction of G, such that a schedule with minimum precipitation and at most 3 sections per day corresponds to a shortest path in G (and vice-versa)? How does this affect the running time of your algorithm?

Problem 5

You are given a set of triangles in the plane. Each triangle is provided by the coordinates of its corners. For simplicity, we assume that no two vertices share the same x-coordinate. We want to determine whether the set contains two triangles that overlap.



3 P a) We first determine whether there are triangles whos boundaries intersect (left figure). Describe a scanline algorithm that determines this as efficiently as possible. You may assume that no two triangle boundaries touch without crossing. What is the running time of your algorithm?

6 P b) If there are no triangle boundaries that intersect, we still need to determine whether there is a triangle that is contained in some other triangle (right figure). Describe a scanline algorithm that determines this as efficiently as possible. You may assume that no two triangle boundaries *touch or intersect*. What is the running time of your algorithm?