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
Beat Gfeller

Prüfung
Datenstrukturen und Algorithmen
D-INFK

4. Februar 2009

Name, Vorname: ______________________________

Stud.-Nummer: ______________________________

Ich bestätige mit meiner Unterschrift, dass ich diese Prüfung unter regulären Bedingungen ablegen konnte und dass ich die untenstehenden Hinweise gelesen und verstanden habe.

Unterschrift: ______________________________

Hinweise:

- Ausser einem Wörterbuch dürfen Sie keine Hilfsmittel verwenden.
- Bitte schreiben Sie Ihre StudentInnen-Nummer auf **jedes** Blatt.
- Melden Sie sich bitte **sofort**, wenn Sie sich während der Prüfung in irgendeiner Weise bei der Arbeit gestört fühlen.
- Bitte verwenden Sie für jede Aufgabe ein neues Blatt. Pro Aufgabe kann nur eine Lösung angegeben werden. Ungültige Lösungsversuche müssen klar durchgestrichen werden.
- Bitte schreiben Sie **lesbar** mit blauer oder schwarzer Tinte. Wir werden nur bewerten, was wir lesen können.
- Die Prüfung dauert 120 Minuten. **Keine Angst!** Wir rechnen nicht damit, dass irgendjemand alles löst! Sie brauchen bei weitem nicht alle Punkte, um die Bestnote zu erreichen.

Viel Erfolg!

Stud.-Nummer: ______________________

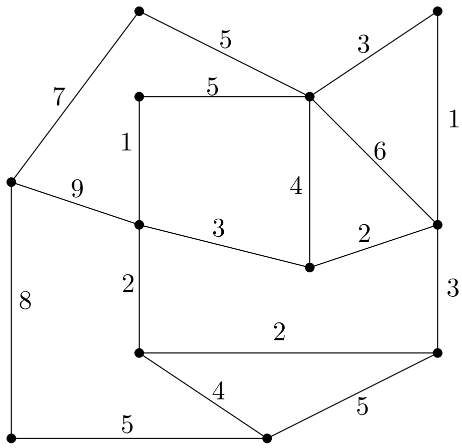
Aufgabe	1	2	3	4	5	$\boldsymbol{\Sigma}$
Mögl. Punkte	9	7	7	10	10	43
Punkte						

Aufgabe 1:

Hinweise:

1. In dieser Aufgabe sollen Sie **nur die Ergebnisse** angeben. Diese können Sie direkt bei den Aufgaben notieren.
2. Sofern Sie die Notationen, Algorithmen und Datenstrukturen aus der Vorlesung "Datenstrukturen & Algorithmen" verwenden, sind Erklärungen oder Begründungen nicht notwendig. Falls Sie jedoch andere Methoden benutzen, müssen Sie diese **kurz** soweit erklären, dass Ihre Ergebnisse verständlich und nachvollziehbar sind.
3. Als Ordnung verwenden wir für Buchstaben die alphabetische Reihenfolge, für Zahlen die aufsteigende Anordnung gemäss ihrer Grösse.

1 P a) Markieren Sie im untenstehenden gewichteten Graphen die Kanten eines minimalen Spannbaums.



1 P b) Führen Sie auf der untenstehenden Folge von Zahlen den ersten Schritt von *Sortieren durch Fachverteilung* aus, d.h. eine Verteilungsphase gefolgt von einer Sammelphase. Zeichnen Sie nur den Zustand der Folge nach der Sammelphase ein.

Zu sortierende Zahlenfolge: $12, 07, 19, 94, 27, 99, 14, 10$.
Nach der ersten Sammelphase:

1 P

c) Das untenstehende Array soll in einen Min-Heap in der üblichen Darstellung umgewandelt werden. Verwenden Sie dazu den Linearzeit-Algorithmus aus der Vorlesung, der den Heap von unten nach oben aus Teil-Heaps aufbaut.

11	9	10	3	7	5	1	8	
1	2	3	4	5	6	7	8	(Index)

1 P

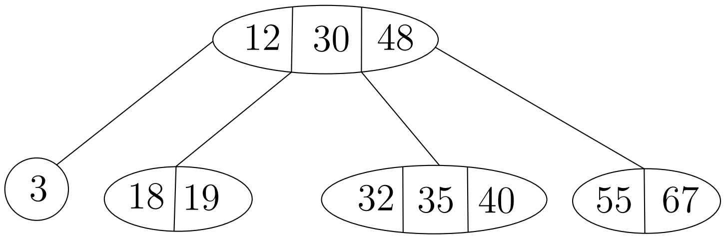
d) Zeichnen Sie den binären Suchbaum, dessen Postorder-Traversierung die Folge $1, 4, 3, 7, 8, 5, 14, 15, 10, 9$ ergibt.

1 P

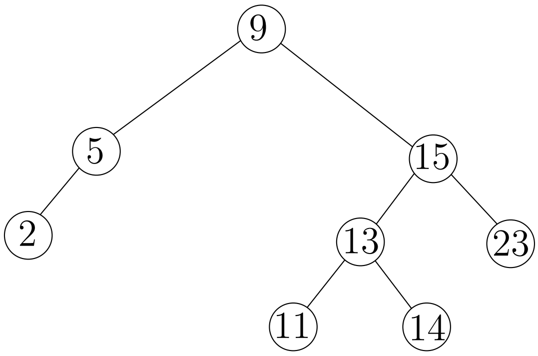
e) Fügen Sie die Schlüssel $19, 23, 30, 12, 27$ in dieser Reihenfolge mittels Offenem Hashing in die folgende Hashtabelle ein (die bereits ein paar Schlüssel enthält), und benutzen Sie dabei Double Hashing. Die zu verwendende Hash-Funktion ist $h(k) = k \mod 11$, und für das Sondieren soll die Hashfunktion $h'(k) = 1 + (k \mod 9)$ benutzt werden.

99			14		16					10	
0	1	2	3	4	5	6	7	8	9	10	(Index)

1 P f) Gegeben sei der folgende B-Baum der Ordnung 4. Fügen Sie in diesen den Schlüssel 36 ein.

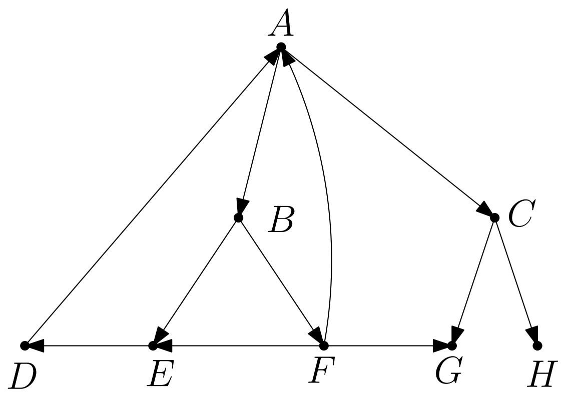


1 P g) Fügen Sie in den untenstehenden AVL-Baum den Schlüssel 10 ein, und löschen Sie im entstandenen AVL-Baum den Schlüssel 15.

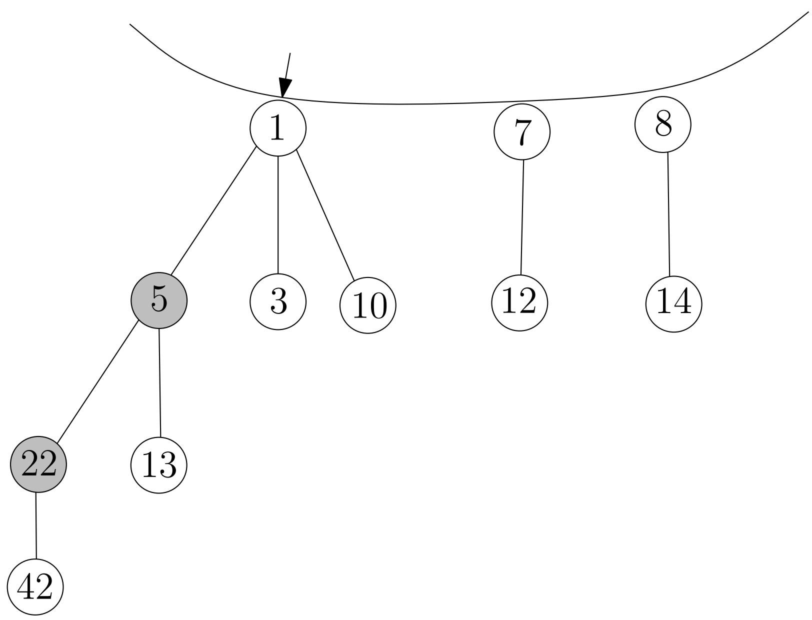


Nach Einfügen von 10:	Nach Löschen von 15:

1 P h) Der folgende gerichtete Graph wird mit Breitensuche traversiert. Die Suche startet beim Knoten A. Geben Sie eine Reihenfolge an, in der die Knoten erreicht werden können.



1 P i) Führen Sie auf dem unten stehenden Fibonacci-Heap die Operation *Decrease-Key*$(42, 21)$ aus, welche den Schlüssel 42 auf 21 herabsenkt. Beachten Sie, dass die Knoten 5 und 22 markiert sind.



Aufgabe 2:

1 P a) Geben Sie für die untenstehenden Funktionen eine **Reihenfolge** an, so dass folgendes gilt: Wenn Funktion f links von Funktion g steht, so gilt $f \in O(g)$.

Beispiel: Die drei Funktionen n^3, n^7, n^9 sind bereits in der entsprechenden Reihenfolge, da $n^3 \in O(n^7)$ und $n^7 \in O(n^9)$ gilt.

- $\dfrac{n}{\log(n)}$
- $\sqrt{n}$
- $(\log(n))^3$
- $2^{\sqrt{n}}$
- $\log(n^5)$
- $2^{3\log_2(n)}$

3 P b) Gegeben ist die folgende Rekursionsgleichung:

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

Geben Sie eine geschlossene (d.h. nicht-rekursive) Formel für $T(n)$ an und beweisen Sie diese mit vollständiger Induktion.

Hinweise:
(1) Sie können annehmen, dass n eine Potenz von 2 ist.
(2) Für $q \neq 1$ gilt: $\sum_{i=0}^{k} q^i = \frac{q^{k+1}-1}{q-1}$.

1 P c) Geben Sie die asymptotische Laufzeit in Abhängigkeit von $n \in \mathbb{N}$ für folgenden Algorithmus in Theta-Notation an:

```
from i := 1; j := 0 until j >= n loop
    j := j + i
    i := i * 2
end
```

1 P d) Geben Sie die asymptotische Laufzeit in Abhängigkeit von $n \in \mathbb{N}$ für folgenden Algorithmus in Theta-Notation an:

```
r := 1
from i := n until i <= 0 loop
    if i \\ 2 = 0 then            -- \\ ist der modulo-Operator in Eiffel
        r := r * r
        i := i // 2
    else
        r := r * 3
        i := i - 1
    end
end
```

1 P e) Geben Sie die asymptotische Laufzeit in Abhängigkeit von $n \in \mathbb{N}$ für folgenden Algorithmus in Theta-Notation an:

```
from i := 1 until i > n loop
    from j := n until j < 4*i loop
        j := j - 1
    end
    i := i * 3
end
```

Aufgabe 3:

In dieser Aufgabe geht es um den Median einer Folge von Zahlen. Sie erinnern sich: der Median einer Folge von n Zahlen ist diejenige Zahl, welche in der sortierten Reihenfolge dieser Zahlen an Stelle $\lfloor n/2 \rfloor$ steht. Zum Beispiel ist der Median der Folge $\{12, 2, 10, 15, 1, 7\}$ die Zahl 7.
Im Folgenden meinen wir mit $A \bigodot B$ die Folge, welche entsteht, wenn man die Folgen A und B in dieser Reihenfolge hintereinanderfügt. Zudem nehmen wir an, dass alle Zahlen *eindeutig* sind, d.h. dass in A und B jeweils keine Zahl mehrfach vorkommt, und dass es keine Zahl gibt, die sowohl in A als auch in B vorkommt. Die Folgen sind als zwei Arrays `A: ARRAY[INTEGER]` und `B: ARRAY[INTEGER]` gegeben, in denen die Zahlen in sortierter Reihenfolge vorliegen.

Ein Beispiel der entsprechenden Arrays von zwei Folgen mit Länge 6 sieht so aus:

`A:`

4	5	8	13	17	22

`B:`

1	3	6	9	18	30

2 P a) Gegeben seien zwei sortierte Folgen A und B von je n verschiedenen Zahlen, sowie ein Index i mit $1 \leq i \leq n$. Finden Sie einen Weg, die Frage "Ist $A[i]$ der Median der Folge $A \bigodot B$?" für einen gegebenen Index i möglichst effizient zu beantworten (dabei bezeichnet $A[i]$ die i-te Zahl in der Folge A). Beschreiben Sie Ihre Lösung in Worten oder Pseudocode, und geben Sie die asymptotische Laufzeit Ihrer Lösung an.

3 P b) Gegeben seien zwei sortierte Folgen A und B von je n verschiedenen Zahlen. Die Frage "Enthält die Folge A den Median der Folge $A \bigodot B$?" soll möglichst effizient beantwortet werden. Entwerfen und beschreiben Sie einen entsprechenden Algorithmus in Worten oder Pseudocode, und geben Sie die asymptotische Laufzeit Ihrer Lösung an.

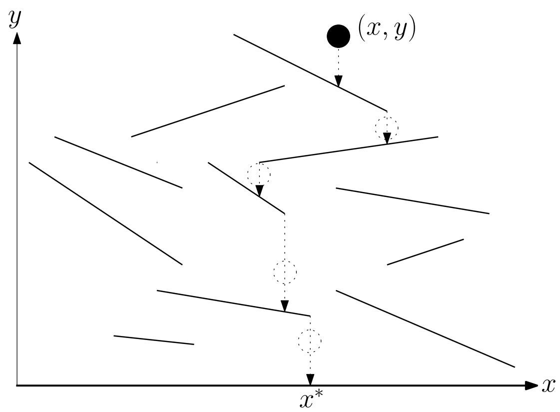
2 P c) Gegeben seien zwei sortierte Folgen A und B von je n verschiedenen Zahlen. Der Median der Folge $A \bigodot B$ soll möglichst effizient bestimmt werden. Beschreiben Sie Ihren Algorithmus in Worten, und geben Sie die asymptotische Laufzeit Ihrer Lösung an.

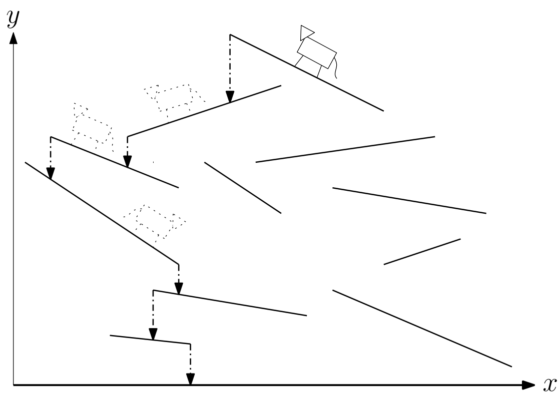
Aufgabe 4:

Wir betrachten eine Kugel, die über eine Anordnung von Holzrinnen hinunter rollt. Die Kugel wird an einem bestimmten Punkt losgelassen. Von dort fällt sie gerade herunter, bis sie auf eine Rinne trifft. Ab da rollt sie diese Rinne hinunter, bis ans untere Ende der Rinne, und fällt dann wieder gerade nach unten. Dies wiederholt sich, bis die Kugel auf dem Boden auftrifft. Im Folgenden repräsentieren wir die Rinnen als Liniensegmente. Wir nehmen an, dass sich keine zwei Segmente berühren oder schneiden. Alle Endpunkte haben paarweise verschiedene x- und y-Koordinaten. Zudem ist kein Segment horizontal.
Jedes Segment ist als Objekt der Klasse `SEGMENT` repräsentiert, und alle Segmente der Anordnung sind im Array `rinnen: ARRAY[SEGMENT]` gespeichert.

```
class SEGMENT
feature -- ACCESS
    xl, yl: INTEGER
    xr, yr: INTEGER
end -- class SEGMENT
```

Die Startposition der Kugel ist durch die Koordinate (x, y) gegeben, wobei `x,y: INTEGER`.





3 P a) Beschreiben Sie kurz in Worten einen Algorithmus, der möglichst effizient ermittelt, an welcher x-Position x^* die Kugel auf den Boden trifft.

1 P b) Geben Sie die Laufzeit Ihres Algorithmus in Abhängigkeit der Anzahl Liniensegmente an.

3 P c) Beschreiben Sie Ihre Lösung für Teilaufgabe a) in Pseudocode. Ihr Pseudocode muss sich an eine der folgenden Sprachen anlehnen: Eiffel, Java, C++. Sie dürfen dabei grundlegende Algorithmen und Datenstrukturen (Sortieren, Balancierter Suchbaum, etc.) verwenden, ohne deren Code aufzuschreiben.

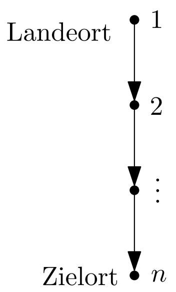
3 P d) Wir betrachten nun statt einer Kugel eine Katze. Diese steht ursprünglich auf einem bestimmten Liniensegment. Sie kann dann wählen, ob sie am oberen oder am unteren Ende des Segments herunterspringt, und fällt vom gewählten Ende aus gerade herunter, bis sie wieder auf ein Segment trifft. Sie getraut sich aber nur herunter zu springen, wenn der Sprung eine Maximalhöhe H nicht überschreitet. Beschreiben Sie in Worten einen Algorithmus, der möglichst effizient berechnet, ob die Katze von einem gegebenen Start-Segment aus den Boden erreichen kann, so dass jeder einzelne Sprung höchstens Höhe H beträgt.

Aufgabe 5:

Für eine Marsmission wird die Fahrt eines Roboters vom Landeort zum Zielort geplant. Der Roboter soll dabei geradlinig zum Zielort fahren, und hat dafür genau T Tage Zeit. Jeden Tag kann der Roboter entweder an seinem aktuellen Ort bleiben, oder genau einen Ort geradewegs weiter in Richtung Zielort fahren. Er darf jedoch nie rückwärts fahren. Jeder Aufenthaltsort birgt eine gewisse Gefahr, die von Tag zu Tag variiert. Das *Gesamtrisiko* einer Fahrt ist die Summe der T Gefahren, denen der Roboter während der Fahrt ausgesetzt ist. Wenn er bereits am Tag $t < T$ beim Zielort ankommt, wird daher für die Tage nach Tag t jeweils die Gefahr des Zielorts zum Gesamtrisiko addiert. Nun soll eine Fahrt geplant werden, die genau T Tage dauert, so dass das Gesamtrisiko minimal ist. Eine solche Fahrt heisst *sicherste Fahrt.*

Im Detail ist folgendes gegeben: Die Anzahl der Orte n, die Fahrtdauer T, sowie ein $n \times T$ grosses zweidimensionales Array `G: ARRAY2[INTEGER]`, welches die Gefahr jedes Orts an jedem Tag enthält: Die Gefahr des Orts i am Tag j ist als `G[i,j]` gegeben. Der Roboter startet an Ort 1 und muss zum Ort n gehen.

Ein Beispiel mit $n = 4$ und $T = 6$ ist in der untenstehenden Abbildung gezeigt. In diesem Beispiel ist die sicherste Fahrt mit Kreisen eingezeichnet. Diese Fahrt erreicht den Zielort bereits am Tag 5, und hat das Gesamtrisiko $1 + 4 + 1 + 6 + 2 + 1 = 15$.



Gefahr / Ort \ Tag	1	2	…			T
1	1	5	2	7	3	9
2	1	4	1	1	2	2
⋮	3	1	8	6	8	4
n	9	7	2	2	2	1

3 P a) Erstellen Sie ein rekursives Programm in Pseudocode, welches das Gesamtrisiko einer sichersten Fahrt, die genau T Tage dauert, berechnet. Geben Sie die Laufzeit des rekursiven Algorithmus an.

5 P b) Beschreiben Sie einen Algorithmus nach dem Muster der dynamischen Programmierung, der das Gesamtrisiko einer sichersten Fahrt, die genau T Tage dauert, berechnet. Geben Sie die Laufzeit Ihres Algorithmus an.

2 P c) Sie erfahren, dass die Maximal-Geschwindigkeit des Roboters erhöht wurde: er kann nun bis zu k Orte an einem Tag weiterreisen (statt einem Ort wie bisher). Das Risiko für die Teilfahrt eines Tages ist nun gegeben durch die Gefahr des letzten an diesem Tag besuchten Ortes. (Wenn der Roboter z.B. am $i$ten Tag von Ort a zum Ort b fährt, wird nur die Gefahr von Ort b am Tag i zum Gesamtrisiko hinzugezählt.) Unter dieser Bedingung soll erneut das Gesamtrisiko einer sichersten Fahrt, die genau T Tage dauert, berechnet werden.

Beschreiben Sie in Worten, wie Sie die Lösung von b) für diese Variante anpassen können, und geben Sie die Laufzeit des angepassten Algorithmus an.