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

Prüfung
Datenstrukturen und Algorithmen
D-INFK

7. August 2014

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 Studierenden-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.
- Sie dürfen alle Algorithmen und Datenstrukturen aus der Vorlesung verwenden, ohne sie noch einmal zu beschreiben. Wenn Sie sie modifizieren, reicht es, die Modifikationen zu beschreiben.
- Die Prüfung dauert 180 Minuten.

Viel Erfolg!

Stud.-Nummer: _____

Aufgabe	1	2	3	4	**Σ**
Mögl. Punkte	16	13	6	10	45
$\sum$ 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) Führen Sie auf dem folgenden Array zwei Iterationen des Sortieralgorithmus *Sortieren durch Einfügen* aus. Das zu sortierende Array ist durch vorherige Iterationen bereits bis zum Doppelstrich sortiert worden.

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

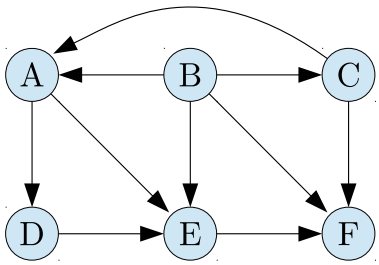
1	2	3	4	5	6	7	8	9	10	11	12

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

1 P b) Fügen Sie die Schlüssel 9, 11, 17, 25, 31, 20 in dieser Reihenfolge in die untenstehende Hashtabelle ein. Benutzen Sie Double Hashing mit der Hashfunktion $h(k) = k \bmod 11$, und benutzen Sie $h'(k) = 1 + (k \bmod 9)$ zur Sondierung.

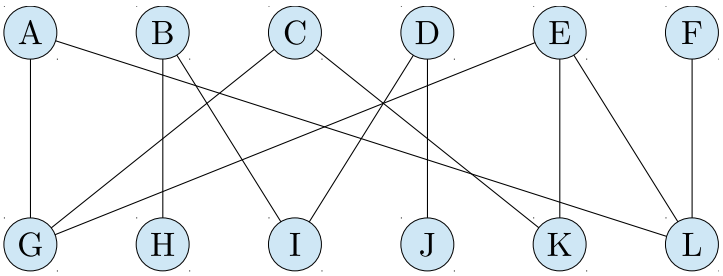
0	1	2	3	4	5	6	7	8	9	10

1 P c) Geben Sie eine topologische Sortierung des untenstehenden Graphen an.



Topologische Sortierung: _____, _____, _____, _____, _____, _____

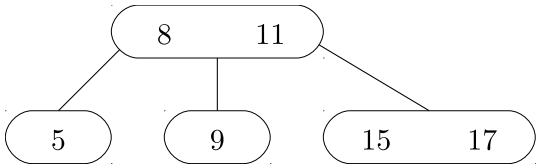
1 P d) Geben Sie eine *möglichst kleine* Teilmenge der Knoten des folgenden Graphen an, die mit dem Satz von Hall beweist, dass der Graph kein perfektes Matching besitzt.



1 P e) Geben Sie an, wieviele Schlüsselvergleiche benötigt werden, wenn in der folgenden selbstanordnenden Liste mit der Move-to-Front-Regel auf die Elemente 'L', 'A', 'M', 'A', 'H', 'A', 'A', 'R' in dieser Reihenfolge zugegriffen wird.

$$\text{A} \longrightarrow \text{L} \longrightarrow \text{G} \longrightarrow \text{O} \longrightarrow \text{R} \longrightarrow \text{I} \longrightarrow \text{T} \longrightarrow \text{H} \longrightarrow \text{M} \longrightarrow \text{U} \longrightarrow \text{S}$$

1 P f) Fügen Sie in den untenstehenden 2-3-Baum (B-Baum der Ordnung 3) zuerst den Schlüssel 7 und in den entstehenden Baum den Schlüssel 20 ein. Führen Sie auch die zugehörigen Strukturänderungen durch.



Nach Einfügen von 7:	Nach Einfügen von 20:

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

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

- 2^n
- $\binom{n}{2}$
- $n \cdot (\log n)^5$
- 15^7
- $3^{n/2}$
- $n!$
- $\frac{n}{(\log n)^2}$

3 P h) Gegeben ist die folgende Rekursionsgleichung:

$$T(n) := \begin{cases} 12 + 7T(n/7) & n > 1 \\ 3 & n = 1 \end{cases}$$

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

Hinweise:

(1) Sie können annehmen, dass n eine Potenz von 7 ist.

(2) Für $q \neq 1$ gilt: $\sum_{i=0}^{k} q^i = \frac{q^{k+1}-1}{q-1}$.

1 P i) Geben Sie die asymptotische Laufzeit in Abhängigkeit von $n \in \mathbb{N}$ für den folgenden Algorithmus (so knapp wie möglich) in Θ-Notation an. Sie müssen Ihre Antwort nicht begründen.

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

1 P j) Geben Sie die asymptotische Laufzeit in Abhängigkeit von $n \in \mathbb{N}$ für den folgenden Algorithmus (so knapp wie möglich) in Θ-Notation an. Sie müssen Ihre Antwort nicht begründen.

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

1 P k) Zeigen oder widerlegen Sie: Der minimale Spannbaum eines ungerichteten gewichteten Graphen $G = (V, E, w)$ ist genau dann eindeutig, wenn G keine zwei Kanten mit dem gleichen Gewicht besitzt.

3 P l) Gegeben ist ein AVL-Baum der Höhe h (zur Erinnerung: Ein AVL-Baum der Höhe 1 besteht aus genau einem Knoten). Wir definieren $\phi := (1+\sqrt{5})/2$. Sei $N(h)$ die minimale Knotenanzahl eines AVL-Baums mit der Höhe h. Zeigen Sie durch allgemeine Induktion über h, dass jeder AVL-Baum der Höhe h mindestens ϕ^{h-1} viele Knoten hat, d.h., dass $N(h) \geq \phi^{h-1}$ gilt. Betrachten Sie dazu im Induktionsschritt einen AVL-Baum mit der Höhe h und minimaler Knotenanzahl, und skizzieren Sie die Struktur eines solchen Baums.

Hinweis: Es gilt $\phi^2 = 1 + \phi$.

Aufgabe 2.

Ein Student möchte heute und morgen jeweils eine Portion Bratkartoffeln kochen und hat dazu n Kartoffeln $\{1, \ldots, n\}$ mit einem Gesamtgewicht von $G \in \mathbb{N}$ Gramm zur Verfügung. Die Kartoffel i wiegt $g_i \in \mathbb{N}$ Gramm. Es sollen nun *alle* n Kartoffeln so auf die zwei Portionen A und B verteilt werden, dass diese annähernd gleich schwer sind. Da die Portionen für unterschiedliche Tage bestimmt sind, muss eine Kartoffel entweder komplett für die Portion A oder komplett für die Portion B verwendet werden (jede Kartoffel muss entweder sofort benutzt werden oder *vollständig* für morgen aufbewahrt werden). Konkret suchen wir also zwei Mengen A und B von Kartoffeln mit $A \cap B = \emptyset$, $A \cup B = \{1, \ldots, n\}$, deren Gewichtsdifferenz minimal ist.

8 P a) Geben Sie einen Algorithmus an, der nach dem Prinzip der dynamischen Programmierung arbeitet und die kleinstmögliche Gewichtsdifferenz zweier Mengen A und B (wie oben beschrieben) berechnet. Gehen Sie in Ihrer Lösung auf die folgenden Aspekte ein.

1) Was ist die Bedeutung eines Tabelleneintrags, und welche Grösse hat die DP-Tabelle?
2) Wie berechnet sich ein Tabelleneintrag aus früher berechneten Einträgen?
3) In welcher Reihenfolge können die Einträge berechnet werden?
4) Wie kann aus der DP-Tabelle der Wert der kleinstmöglichen Gewichtsdifferenz ausgelesen werden?

Hinweis: Der triviale Algorithmus, der einfach alle möglichen Lösungen aufzählt, gibt **keine** Punkte.

2 P b) Beschreiben Sie detailliert, wie aus der DP-Tabelle abgelesen werden kann, welche Kartoffel an welchem Tag benutzt wird.

3 P c) Geben Sie die Laufzeit des in a) und b) entwickelten Verfahrens an und begründen Sie Ihre Antwort. Ist die Laufzeit polynomiell?

Aufgabe 3.

Im Devisenhandel bezeichnet *Arbitrage* das Ausnutzen von Preisunterschieden, um durch mehrfaches Wechseln von Währungen einen Gewinn zu erzielen. Am 2. Juni 2009 zum Beispiel konnte 1 US-Dollar in 95.729 Yen gewechselt werden, 1 Yen in 0.00638 Britische Pfund und 1 Britisches Pfund in 1.65133 US-Dollar. Hätte ein Händler also 1 US-Dollar in Yen gewechselt, den erhaltenen Betrag in Britische Pfund und schliesslich diesen Betrag dann zurück in US-Dollar getauscht, dann hätte er $95.729 \cdot 0.00638 \cdot 1.65133 \approx 1.0086$ US-Dollar erhalten, was einem Gewinn von 0.86% entspricht.

3 P a) Gegeben seien n Währungen $\{1, \ldots, n\}$ und eine $(n \times n)$-Wechselkursmatrix $R \in (\mathbb{Q}^+)^2$. Für zwei Währungen $i, j \in \{1, \ldots, n\}$ kann eine Einheit der Währung i in $R(i, j) > 0$ Einheiten der Währung j getauscht werden. Es soll entschieden werden, ob ein Arbitrage-Geschäft möglich ist, d.h., ob es eine Folge von k verschiedenen Währungen $W_1, \ldots, W_k \in \{1, \ldots, n\}$ mit $R(W_1, W_2) \cdot R(W_2, W_3) \cdots R(W_{k-1}, W_k) \cdot R(W_k, W_1) > 1$ gibt.

Modellieren Sie das Problem als Graphproblem. Konstruieren Sie aus der obigen Eingabe einen gerichteten, gewichteten Graphen $G = (V, E, w)$, der *genau dann* einen Kreis mit negativem Gewicht besitzt, wenn ein Arbitrage-Geschäft möglich ist. Begründen Sie, dass G genau dann einen Kreis negativer Länge enthält, wenn ein Arbitrage-Geschäft möglich ist.

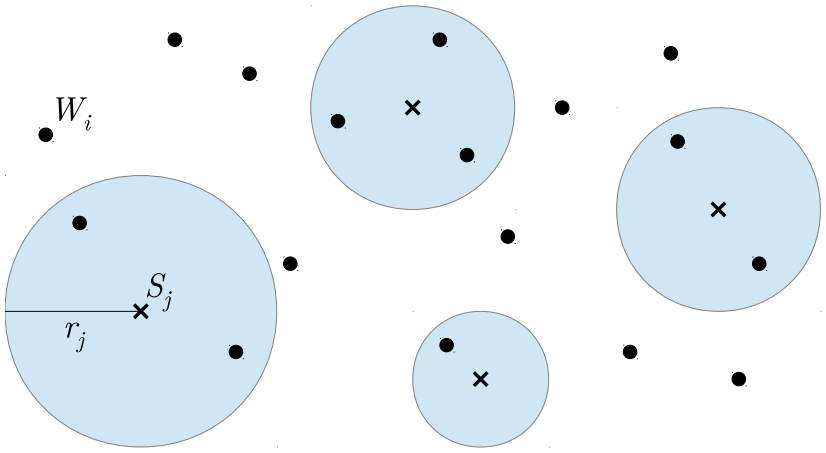
Hinweis: Die Benutzung des Logarithmus kann sinnvoll sein, denn es gilt $\ln(a \cdot b) = \ln(a) + \ln(b)$.

3 P b) Welcher Kürzeste-Wege-Algorithmus kann benutzt werden, um in einem Graphen Kreise negativer Länge zu erkennen? An welchem Knoten kann der Algorithmus beginnen? Welche Laufzeit (in Abhängigkeit von n) erreicht der gewählte Algorithmus, wenn er auf den Graphen in a) angewendet wird?

Aufgabe 4.

Gegeben sei eine zweidimensionale Landkarte, die einen Ausschnitt eines Gebirges zeigt. Auf dieser Karte sind die Positionen von n Wanderern eingetragen. Der Wanderer i befindet sich an der Position $W_i = (x_i^W, y_i^W) \in \mathbb{Q}^2$. Im Gebirge sind m Mobilfunk-Sendemasten aufgestellt. Der Sendemast j hat die Position $S_j = (x_j^S, y_j^S) \in \mathbb{Q}^2$ und eine Reichweite von $r_j \in \mathbb{Q}^+$. Konkret heisst dies, dass die Natels aller Personen, die sich innerhalb des Kreises mit Mittelpunkt S_j und Radius r_j befinden, Empfang haben. Die Aufgabe besteht nun darin, alle Wanderer zu identifizieren, deren Natel *keinen* Empfang hat.

Beispiel: In der untenstehenden Karte sind die Positionen der Wanderer als dicke schwarze Punkte eingetragen. Die Positionen der Sendemasten entsprechen den als Kreuzen eingezeichneten Mittelpunkten der Kreise, und die Reichweiten korrespondieren mit den entsprechenden Radien der Kreise.



9 P a) Entwerfen Sie einen möglichst effizienten Scanline-Algorithmus, der als Eingabe die Positionen $W_1, \ldots, W_n$ von n Wanderern, die Positionen $S_1, \ldots, S_m$ von m Sendemasten sowie die entsprechenden Reichweiten $r_1, \ldots, r_m$ erhält, und der die Menge aller Wanderer berechnet, deren Natel keinen Empfang hat. Gehen Sie in Ihrer Lösung auf die folgenden Aspekte ein.

1) In welche Richtung verläuft die Scanline, und was sind die Haltepunkte?

2) Welche Objekte muss die Datenstruktur verwalten, und was ist eine angemessene Datenstruktur?

3) Was passiert, wenn die Scanline auf einen neuen Haltepunkt trifft?

4) Wie können schliesslich alle Wanderer ohne Natel-Empfang identifiziert werden?

Hinweis: Der Einfachheit halber nehmen wir an, dass sich die Reichweiten keiner zwei Sendemasten überschneiden. Ebenso dürfen Sie wie immer davon ausgehen, dass keine degenerierten Fälle auftreten, das heisst zum Beispiel, dass keine zwei Wanderer die gleiche x- oder y-Koordinate besitzen.

1 P b) Geben Sie die Laufzeit Ihres in a) entwickelten Algorithmus an und begründen Sie Ihre Antwort.