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
Thomas Tschager

Prüfung
Datenstrukturen und Algorithmen
D-INFK

21. Januar 2015

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	14	13	11	10	48
$\sum$ 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 Auswahl* aus. Das zu sortierende Array ist durch vorherige Iterationen bereits bis zum Doppelstrich sortiert worden.

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

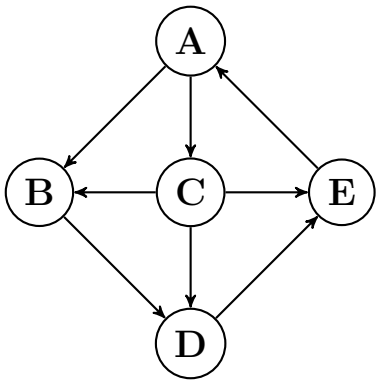
1	2	3	4	5	6	7	8	9	10	11	12

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

1 P b) Fügen Sie die Schlüssel 8, 10, 15, 9, 17 in dieser Reihenfolge in die untenstehende Hashtabelle ein. Benutzen Sie offenes Hashing mit der Hashfunktion $h(k) = k \bmod 7$ und lösen Sie Kollisionen mittels quadratischem Sondieren auf.

0	1	2	3	4	5	6

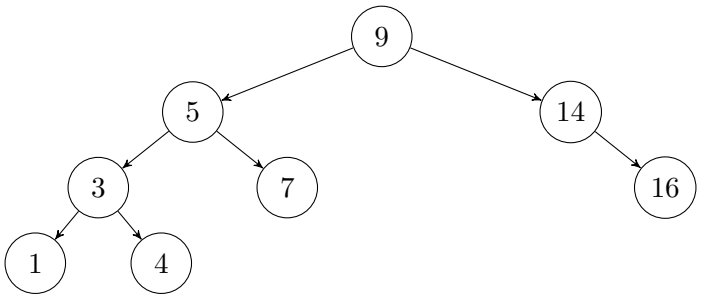
1 P c) Geben Sie jeweils eine Reihenfolge an, in der die Knoten des folgenden Graphen von einer Breitensuche (BFS) bzw. Tiefensuche (DFS) mit Startknoten A besucht werden, wenn die Nachbarn eines Knotens in alphabetischer Reihenfolge abgearbeitet werden.



BFS: _____

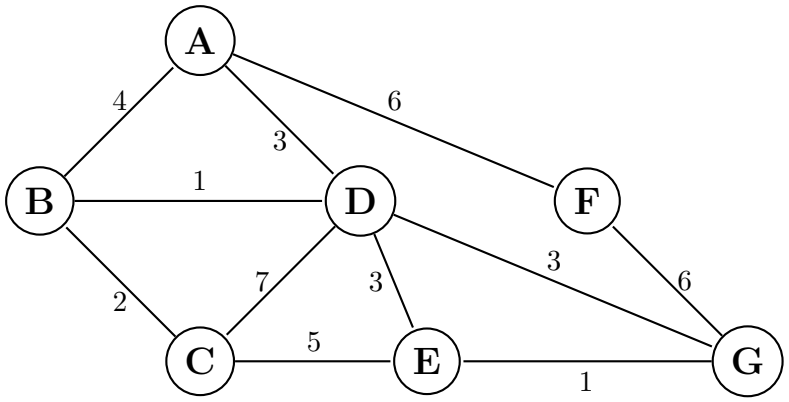
DFS: _____

1 P d) Fügen Sie in den folgenden AVL-Baum *zuerst* den Schlüssel 2 ein und entfernen Sie *danach* den Schlüssel 14.



Nach Einfügen von 2:	Nach Löschen von 14:

1 P e) Markieren Sie in der untenstehenden Abbildung die ersten *drei* Kanten, die der Algorithmus von Jarník, Prim und Dijkstra *ausgehend von Knoten A* in den minimalen Spannbaum aufnimmt.



1 P f) Zeichnen Sie denjenigen binären Suchbaum, bei dem in Postorder-Reihenfolge die Schlüssel 4, 7, 6, 10, 11, 9 angetroffen werden.

2 P g) Gegeben sei ein gewichteter Graph G. Zeigen oder widerlegen Sie die folgenden Aussagen:

1. Wenn $P = \langle v_1, \ldots, v_k \rangle$ und $Q = \langle w_1, \ldots, w_l \rangle$ kürzeste Pfade mit $v_k = w_1$ sind, dann ist $\langle v_1, \ldots, v_k = w_1, \ldots, w_l \rangle$ ein kürzester Pfad von v_1 nach w_l.
2. Sei P ein kürzester Pfad von u nach v und sei w ein innerer Knoten von P. Der Teilpfad von P von u nach w ist ein kürzester Pfad von u nach w in G. Ebenso ist der Teilpfad von w nach v ein kürzester Pfad in G.

1 P h) 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.

- $\binom{n}{4}$
- $\log^2(n)$
- $n \cdot \sqrt{n}$
- $n!$
- $\log(n^5)$
- 7^{13}
- $\log(n^n)$
- $\sqrt{6^n}$

3 P i) Gegeben ist die folgende Rekursionsgleichung:

$$T(n) := \begin{cases} 15 + 4T(n/4) & n > 1 \\ 1 & n = 1 \end{cases}$$

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 4 ist.

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

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

1 P k) 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 = n; i > 0; i -= 1) {
2         for(int j = 0; j < i; j += 1) {
3                 ;
4         }
5 }
```

Aufgabe 2.

Eine Firma hat den Auftrag bekommen, ein Kabel der Länge mindestens L zu produzieren. Da es im Lager viele bereits früher produzierte Kabelstücke gibt, soll das gewünschte Kabel aus diesen Teilstücken zusammengesetzt werden. Konkret gibt es im Lager n Kabelstücke. Kabelstück i hat Länge l_i. Werden zwei Kabelstücke mit Längen l_i und l_j zusammengesetzt, entsteht ein Kabel der Länge $l_i + l_j$. Um das entstehende Kabel nicht unnötig lang zu machen, soll es unter allen Möglichkeiten, ein Kabel der Länge $\geq L$ herzustellen, minimale Länge haben. Sie dürfen annehmen, dass der Lagerbestand ausreichend gross ist, d.h., dass die Summe der Längen aller Kabelstücke mindestens L beträgt.

Beispiel: Es soll ein Kabel der Länge $L = 6$ produziert werden, und im Lager gibt es Kabelstücke der Längen $l_1 = 3$, $l_2 = 4$ und $l_3 = 5$. Die beste Möglichkeit ist die Wahl der Kabelstücke $\{1, 2\}$, denn diese haben zusammen Länge 7. Auch die Wahlen $\{2, 3\}$ und $\{1, 2, 3\}$ führen zu einem Kabel der Länge ≥ 6, aber da ihre Gesamtlängen 9 bzw. 12 betragen, sind sie nicht optimal und daher auch nicht die gesuchte Lösung.

9 P a) Geben Sie einen Algorithmus an, der nach dem Prinzip der dynamischen Programmierung arbeitet und die minimale Länge eines Kabels berechnet, sodass dieses Kabel aus geeigneten Kabelstücken aus $\{1, \ldots, n\}$ zusammengesetzt werden kann und mindestens Länge L hat. Für das Beispiel oben soll also 7 ausgegeben werden. 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 minimalen Kabellänge ausgelesen werden?

Hinweis: Der triviale Algorithmus, der einfach alle möglichen Lösungen inspiziert, gibt **keine** Punkte, weil er nicht nach dem Prinzip der dynamischen Programmierung arbeitet.

2 P b) Beschreiben Sie detailliert, wie aus der DP-Tabelle abgelesen werden kann, welches Kabelstück in einer optimalen Lösung benutzt wird.

2 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.

In einem Krankenhaus soll bestimmt werden, welcher Arzt an welchem Tag arbeitet. Diese Aufgabe befasst sich mit der Arbeitszeitplanung zu Ferientagen. Sei T die Menge aller Ferientage. Es gibt k Ferienzeiten $\{1, \ldots, k\}$, die aus einem oder mehreren zusammenhängenden Tagen $T_j \subseteq T$ bestehen (für $j \in \{1, \ldots, k\}$). Jeder Tag in T kommt in genau einer Ferienzeit vor, d.h. insbesondere ist $T_i \cap T_j = \emptyset$ für $i \neq j$. Im Krankenhaus arbeiten n Ärzte. Jeder Arzt i gibt eine Menge von Tagen $S_i \subseteq T$ an, an denen er anwesend sein kann. Die Aufgabe besteht nun darin, einen Einsatzplan zu entwerfen, der die Ferientage T so unter den Ärzten verteilt, dass an jedem Tag genau ein Arzt anwesend ist. Ausserdem soll kein Arzt an mehr als $C \in \mathbb{N}$ Ferientagen insgesamt arbeiten müssen. Die Aufgabe besteht nun darin, effizient zu entscheiden, ob für ein gegebenes C ein geeigneter Einsatzplan existiert, und falls ja, effient zu berechnen, wie dieser aussieht.

Beispiel: Sei $C = 2$ gegeben. Die Ärzte und Ferienzeiten seien die folgenden:

i	Arzt	Mögliche Tage S_i
1	Dr. Cuddy	$\{24.12, 25.12, 26.12\}$
2	Dr. Foreman	$\{24.12\}$
3	Dr. House	$\{1.8\}$
4	Dr. Wilson	$\{1.8, 24.12\}$

j	Beschreibung	Zugehörige Tage T_j
1	Weihnachten	$\{24.12, 25.12, 26.12\}$
2	Nationalfeiertag	$\{1.8\}$

Nun existieren verschiedene Möglichkeiten, z.B. könnte Dr. Cuddy am 25. und am 26.12 anwesend sein, Dr. House am 1.8 und Dr. Wilson am 24.12. Wäre stattdessen $C = 1$ gegeben gewesen, dann existierte kein geeigneter Einsatzplan.

4 P a) Modellieren Sie das o.g. Problem als Flussproblem. Beschreiben Sie dazu die Konstruktion eines geeigneten Netzes $N = (V, E, c)$ mit der Knotenmenge V sowie der Kantenmenge E, und geben Sie an, welche Kapazitäten die Kanten besitzen sollen. Wie kann aus dem Wert eines maximalen Flusses abgelesen werden, ob ein geeigneter Einsatzplan existiert oder nicht?

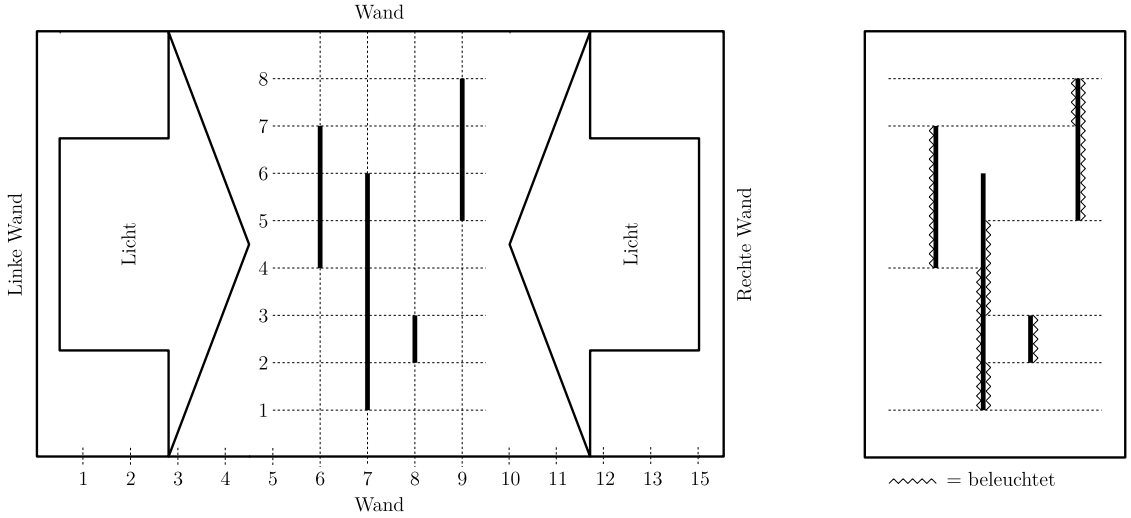
2 P b) Nennen Sie einen Algorithmus, der das Problem aus a) möglichst effizient löst, und geben Sie die Laufzeit im schlimmsten Fall in Abhängigkeit von der Anzahl der Ärzte n und der Anzahl der Ferientage $m = |T|$ an. Begründen Sie Ihre Antwort.

3 P c) Nehmen Sie an, das Flussproblem aus a) wurde bereits gelöst, d.h. zu einem maximalen Fluss ϕ kennen Sie den Fluss ϕ_e auf jeder Kante e. Nehmen Sie weiter an, ein geeigneter Einsatzplan wie oben beschrieben existierte tatsächlich. Beschreiben Sie detailliert einen Algorithmus, der aus den ϕ_e einen solchen Einsatzplan berechnet. Welche Laufzeit hat Ihr Verfahren, wenn auf jedes ϕ_e in Zeit $\Theta(1)$ zugegriffen werden kann?

2 P d) Das bisherige Verfahren hat den Nachteil, dass es u.U. Einsatzpläne generiert, bei denen manche Ärzte sehr viel und andere Ärzte sehr wenig eingesetzt werden. Wir wollen daher einen Einsatzplan finden, der die o.g. Bedingungen erfüllt und zusätzlich jedem Arzt für jedes $j \in \{1, \ldots, k\}$ maximal einen Tag in T_j zuweist. Für das obige Beispiel heisst dies, dass ein Arzt zu Weihnachten entweder am 24.12, am 25.12, am 26.12 oder überhaupt keinen Dienst hat. Beschreiben Sie, wie das in a) konstruierte Netz modifiziert werden muss, um diese zusätzliche Anforderung zu erfüllen.

Aufgabe 4.

Ein Künstler fertigt einen Grundrissplan seines Kunstwerks an, bei dem vertikale Platten mit Laserlicht von der Seite bestrahlt werden.



Es gibt n Platten mit verschiedenen Positionen und Breiten. Platte i werde dabei durch ein Tripel $P_i = (x_i, y_i, b_i)$ repräsentiert, wobei x_i der Abstand zur linken Wand, y_i der Abstand zur unten gezeichneten Wand und b_i die Breite der Platte angeben. Eine mögliche Eingabe für die Skizze oben wäre beispielsweise $P_1 = (7, 1, 5)$, $P_2 = (9, 5, 3)$, $P_3 = (8, 2, 1)$ sowie $P_4 = (6, 4, 3)$.

Die Platten werden von flächenförmigen Laserlichtquellen (an den Wänden links und rechts) bestrahlt, welche über die gesamte Breite und Höhe des Kunstwerks horizontale Lichtstrahlen aussenden. Licht, welches auf eine Platte trifft, wird vollständig absorbiert. Weil das Licht die Platten erhitzt, muss jede bestrahlte Platte gekühlt werden, und zwar proportional zum Betrag des einfallenden Lichts. Deshalb möchte der Künstler nun *für jede Platte* die gesamte bestrahlte Fläche (als Summe der von links bestrahlten Fläche und der von rechts bestrahlten Fläche) berechnen. Da die Platten vom Fussboden bis zur Decke reichen, ist die Höhe aller Platten gleich und es genügt, anstatt der bestrahlten Fläche die bestrahlte Breite der Platten zu berechnen. Daher liegt wie in der obigen Skizze ein zweidimensionales Problem vor.

Für jede Platte i soll die gesamte bestrahlte Breite B_i als Summe der von links bestrahlten Breite und der von rechts bestrahlten Breite berechnet werden (siehe Abbildung rechts). Für die obige Eingabe soll also $B_1 = 6$, $B_2 = 4$, $B_3 = 1$ sowie $B_4 = 3$ ausgegeben werden.

10 P Entwerfen Sie einen möglichst effizienten Scanline-Algorithmus für das obige Problem. 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 Scanline-Datenstruktur verwalten, und was ist eine angemessene Datenstruktur?
3) Was passiert, wenn die Scanline auf einen neuen Haltepunkt trifft?
4) Wie kann für jede Platte die bestrahlte Breite B_i berechnet werden?

5) Welche Laufzeit in Abhängigkeit von n hat Ihr Algorithmus? Begründen Sie Ihre Antwort.

Hinweis: Der Einfachheit halber nehmen wir an, dass sich keine zwei Platten direkt übereinander befinden oder direkt nebeneinander starten oder enden. Beachten Sie aber, dass wie im obigen Beispiel die Platten der Eingabe nicht notwendigerweise nach x-Koordinate sortiert sind.