



Klausur Effiziente Algorithmen WS 04/05

1. Februar 2005

NICHT MIT BLEISTIFT ODER ROTSTIFT SCHREIBEN!

Heften Sie die Blätter bei Abgabe zusammen, und tragen Sie auf jedem Blatt Ihren Namen, Vornamen, Studiennummer und Matrikel ein. Es sind keine Hilfsmittel, insbesondere Taschenrechner oder Mobiltelefone, zugelassen.

Name, Vorname:

Studiennummer und Matrikel:

Code

abgegeben: **2 Aufgabenblätter**
... eigene Blätter

Einsichtnahme
Datum, Unterschrift

Aufgabe	1	2	3	4	5	6	Ges.
erreichbare Punktzahl	15	9	8	6	8	4	50
erreichte Punktzahl							

Aufgabe 1

[15 Punkte]

Bitte kreuzen Sie für jede der folgenden 5 Fragen entweder „JA“ oder „NEIN“ in jeder Zeile an.

Bewertung: Für jede korrekte Antwort erhalten Sie einen Punkt. Für jede falsche Antwort wird Ihnen ein halber Punkt abgezogen. Wurde kein Kreuz gemacht, erhalten Sie keinen Punkt.

(a) Sind folgende Aussagen richtig?

JA NEIN

- Der Algorithmus QuickSelect findet das i -t kleinste Element im schlechtesten Fall in linearer Zeit.
- Eine Folge von Union-Find-Operationen mit Pfadkompression über n Elementen kostet höchstens $O(m \log^*(n))$ Schritte.
- Die Größe der Teilmengen im Select-Algorithmus hat keinen Einfluss auf die Laufzeit.

(b) Welche der folgenden Aussagen sind korrekt?

JA NEIN

- Aus jedem vergleichsbasierten Sortierverfahren lässt sich ein stabiles Verfahren konstruieren.
- Mit BucketSort können beliebige Schlüssel sortiert werden.
- Die vergleichsbasierte Sortierung von n Schlüsseln benötigt mindestens $\Omega(n \log n)$ Vergleiche.

(c) Welche der folgenden Aussagen über die Sortierung von n Schlüsseln sind korrekt?

JA NEIN

- MaxSort benötigt **im schlechtesten Fall** eine Laufzeit von $O(n \log n)$
- HeapSort benötigt **im Mittel** eine Laufzeit von $O(n \log n)$.
- QuickSort benötigt **im schlechtesten Fall** eine Laufzeit von $O(n \log n)$.

(d) Welche der folgenden Aussagen sind korrekt?

JA NEIN

- Fibonacci-Heaps enthalten nur Binomialbäume.
 Die amortisierten Kosten eines DeleteMin in einem Fibonacci-Heap mit n Schlüsseln sind $O(\log n)$.
 Der Algorithmus von Dijkstra mit Fibonacci-Heaps benötigt $O(|V| + |V| \log |V|)$ Schritte.

(e) Welche der folgenden Aussagen sind korrekt?

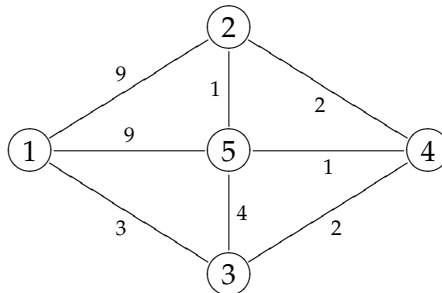
JA NEIN

- Der Algorithmus von Kruskal berechnet einen minimalen Spannbaum in $O(|V| \log |V|)$.
 Die transitive Hülle eines Graphen kann mittels Matrizenmultiplikation berechnet werden.
 Der Algorithmus von Floyd berechnet die Distanzen zwischen allen Paaren von Knoten in einer Laufzeit von $O(|V|^3)$.

Aufgabe 2 (Dijkstra)

(9 Punkte)

Berechnen Sie mittels des Dijkstra-Algorithmus auf dem gegebenen Graphen kürzeste Wege vom Knoten 1 zu jedem anderen Knoten. Geben Sie den Inhalt der Prioritäts-Warteschlange und die bislang konstruierten Wege nach jeder Runde an.



Aufgabe 3 (BucketSort)

(8 Punkte)

Sortieren Sie die folgenden Strings über dem Alphabet $\{e, f, g, h\}$ mittels BucketSort:

$efghh, fefgg, ehgfe, gggeh, heggf, fehgh, efghh.$

Geben Sie den Inhalt aller Buckets nach jedem Durchlauf an.

Aufgabe 4 (Zahlenmengen)

(6 Punkte)

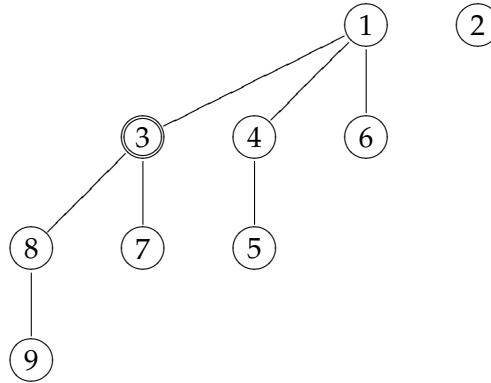
Gegeben sei eine Menge S von n reellen Zahlen x_i mit $0 \leq x_i \leq 1$. Die Zahlen seien gleichmäßig verteilt, d.h. die Wahrscheinlichkeit, dass eine der Zahlen in einem gegebenen Teilintervall von $[0, 1]$ liegt ist gleich der Länge des Intervalls.

Die Zahlen sollen so in einer Datenstruktur organisiert werden, dass die Frage, ob eine gegebene Zahl y in S liegt in erwarteter Zeit $O(1)$ beantwortet werden kann. Beschreiben Sie eine solche Datenstruktur und machen Sie plausibel, warum Ihre Lösung korrekt ist. Wie lange dauert das Einordnen aller Zahlen in die Datenstruktur?

Aufgabe 5 (Fibonacci Heaps)

(8 Punkte)

Geben Sie die Struktur des untenstehenden Fibonacci Heaps nach jedem der Ausführung der Operation `delete(7)` an. Gehen Sie dabei davon aus, dass Knoten 3 markiert ist.



Aufgabe 6 (Algebraische Wegeprobleme)

(4 Punkte)

Sei $G = (\{1, \dots, n\}, E)$ ein gerichteter Graph mit Kantenlängen $l(i, j)$ für $(i, j) \in E$. Es seien folgender Halbring $H = (\mathbb{R}_{\geq 0} \cup \{\infty\}, \min, +, \infty, 0)$ und die $n \times n$ -Matrix $A = (a_{ij})_{1 \leq i, j \leq n}$ gegeben, mit

$$a_{ij} = \begin{cases} l(i, j) & \text{falls } (i, j) \in E \\ \infty & \text{sonst.} \end{cases}$$

- (a) Wie berechnet sich der Eintrag $a_{ij}^{(k)}$ an Position (i, j) der Matrix $A^k = \underbrace{A \cdot \dots \cdot A}_{k\text{-mal}}$ für $k > 1$ aus den Einträgen von A und A^{k-1} ?
- (b) Was geben die Einträge der Matrizen A^k für $k \geq 1$ an?

Viel Spaß und viel Erfolg!

