Prüfung zur Lehrveranstaltung

Datenstrukturen und Algorithmen WS 2000/2001

Es sind keinerlei Unterlagen oder Hilfsmittel erlaubt. Es dürfen nur einzelne, lose Blätter verwendet werden! Auf jedem Blatt muß der Name und die Matrikelnummer angegeben werden! Zeit: 90 Minuten.

(1) (5 Punkte). Betrachten Sie die folgende Aussage: Es existiert eine Datenstruktur S, welche folgende Operationen für n Elemente unterstützt:

1. Ein gegebenes Element in S einfügen in $O(1)$ Zeit.
2. Ein gegebenes Element aus S löschen in $O(log\ n)$ Zeit.
3. Das kleinste Element in S finden und löschen in $O(1)$ Zeit.
4. Das größte Element in S finden in $O(log\ n)$ Zeit.

Beweisen oder widerlegen Sie die Korrektheit dieser Aussage indem Sie entweder eine Datenstruktur und die dazugehörigen Operationen angeben oder beweisen, daß eine derartige Datenstruktur nicht existieren kann.

(2) (5 Punkte). Definieren Sie kurz das Spiel 'Türme von Hanoi'. Geben Sie einen rekursiven Algorithmus an, der dieses Spiel in optimaler Zeit löst und beweisen Sie dessen Korrektheit.

Analysieren Sie dessen Zeit- und Speicherverhalten (nicht raten, sondern rekursive Zeit- und Speichergleichung aufstellen und lösen!). Geben Sie die exakte Anzahl benötigter Scheibenbewegungen bei optimaler Strategie und n Scheiben an.

(3) (5 Punkte). Erklären Sie die Begriffe 'präfixfreie' und 'optimale' Codierung. Warum spielt die Eigenschaft 'präfixfrei' bei optimaler Codierung eine Rolle? Was ist ein Blockcode? Ist ein Blockcode immer präfixfrei? Ist eine präfixfreie Codierung immer eindeutig? Ist eine optimale Codierung immer eindeutig? Ist ein eindeutiger, präfixfreier Code immer eindeutig dekodierbar? Begründen Sie kurz jede Ihrer Antworten!

Beispiel: 7 Zeichen mit der absoluten Häufigkeit [6,4,3,2,1,1,1]. Konstruieren Sie grafisch den zugehörigen Codebaum einer optimalen Codierung. Was kann aus diesem Codebaum jetzt alles abgelesen bzw. sehr leicht berechnet werden?

(4) (5 Punkte). Gegeben sind $2n$ paarweise verschiedene natürliche Zahlen $A = [a_1, \ldots, a_{2n}]$. Geben Sie einen Algorithmus in Pseudocode an, der die Differenz aus der Summe der Kuben der n größeren Zahlen und der Summe der Quadrate der n kleineren Zahlen berechnet.

Analysieren Sie detailliert dessen Laufzeit und Speicherbedarf! Volle Punkte nur für Algorithmen mit Laufzeit in $O(n)$.

Viel Erfolg!