Name: **Vorname**: **Matrikelnr.**:
Tutor: □ Claudia □ Lina □ Maria □ Tobias □ Yvonne

---

1. Klausur zur Vorlesung
Entwurf und Analyse von Algorithmen

Helmut Alt, Claudia Klost

Wintersemester 2003/2004 **19. Dezember 2003**

1.	2.	3.	4.	Σ
/12	/12	/13	/13	/50

Außer Schreibutensilien sind keine Hilfsmittel erlaubt!

Alle Antworten sind ausführlich zu begründen, und die Laufzeit aller entwickelten Algorithmen ist zu analysieren.

Auf diesem Klausurbogen ist genügend Platz, um die Lösungen der Aufgaben aufzuschreiben. Auch die Rückseiten der Blätter können verwendet werden (bitte auf der Vorderseite anmerken). Bitte geheftet lassen! **Zusätzliche lose Blätter** sind von der Aufsicht erhältlich und müssen mit der Matrikelnummer, Namen und Aufgabennummer versehen werden. Auf einem Zusatzblatt jeweils nur eine Aufgabe bearbeiten. Nicht mit Bleistift und nicht mit Rot schreiben. *Der Klausurbogen ist auf jeden Fall abzugeben!*

Aufgabe 1

/12

Gegeben sei ein linear geordnetes Universum $(\mathcal{U}, \leq)$.

Angenommen, die Elemente einer Menge $S \subseteq \mathcal{U}$ sind

(a) in einem binären Suchbaum

(b) in einem (a, b)-Baum

(c) in einer Hashtabelle

gespeichert.

Geben Sie für jede der oben genannten Datenstrukturen einen effizienten Algorithmus an, der die Elemente der Menge S aufsteigend geordnet ausgibt.

Aufgabe 2

/12

Sei $(\mathcal{U}, \leq)$ ein linear geordnetes Universum, und $S \subseteq \mathcal{U}$. Mit vielen Datenstrukturen ist es möglich, die Operation $\textsc{Select}(k,S)$ effizient zu implementieren. Dabei bestimmt $\textsc{Select}(k,S)$ das k-t-kleinste Element der Menge S.

Geben Sie einen effizienten Algorithmus an, der $\textsc{Select}(k,S)$ berechnet und als Datenstruktur $(2,3)$-Bäume verwendet.

Erläutern Sie, wie für Ihren Algorithmus ein $(2,3)$-Baum modifiziert werden muss.

Aufgabe 3 /13

Seien $A = a_1 a_2 \dots a_n$ und $B = b_1 b_2 \dots b_m$ zwei Zeichenfolgen.

$C = c_1 c_2 \dots c_k$ ist eine Teilfolge von A, falls es eine streng monoton wachsende Folge von Indizes $i_1, i_2, \dots i_k$ von A gibt, so dass für alle $j \in \{1, \dots k\}$ $a_{i_j} = c_j$.

Beispiel: Falls $A = w\mathbf{xy}x\mathbf{z}w\mathbf{x}$ und $C = \mathbf{xyzx}$ dann ist C eine Teilfolge von A.

Geben Sie einen Algorithmus an, der die Länge einer längsten gemeinsamen Teilfolge zweier Folgen A und B berechnet. Verwenden Sie für Ihre Lösung das Prinzip der dynamischen Programmierung.

Hinweis: Ihr Algorithmus sollte Werte $w_{i,j}$ für wachsende i und j bestimmen, wobei die $w_{i,j}$ die Länge der längste gemeinsame Teilfolge von $a_1 \dots a_i$ und $b_1 \dots b_j$ ist.

Um die rekursive Definition der $w_{i,j}$ zu finden, ist es hilfreich, die beiden Fälle $a_i = b_j$ und $a_i \neq b_j$ zu unterscheiden.

Aufgabe 4 /13

Seien die ganzzahligen Koeffizienten $a_0, a_1, \ldots a_n$ und die Zahl $c \in \mathbb{Z}$ gegeben.

Das Polynom $P(x) = a_0 + a_1 x + \ldots + a_n x^n$ kann wie folgt in Polynome $P_1(x)$ und $P_2(x)$ mit Grad kleiner oder gleich $\frac{n}{2}$ zerlegt werden:

$$P(x) = P_1(x^2) + xP_2(x^2)$$

Dabei wird P_1 durch die Koeffizienten mit geradem Index und P_2 durch die Koeffizienten mit ungeradem Index definiert.

Sie dürfen für diese Aufgabe annehmen, dass $n + 1$ eine Potenz von 2 ist.

(a) Geben Sie einen effizienten Divide & Conquer-Algorithmus an, der $P(c)$ mit Hilfe obiger Formel berechnet.

(b) Stellen Sie eine Rekursionsgleichung auf, die die Laufzeit Ihres Algorithmus beschreibt und lösen sie diese.

Lösen Sie die Rekursionsgleichung direkt, d.h. greifen Sie nicht auf in der Vorlesung vorgestellte Ergebnisse zurück und benutzen Sie **nicht** das Master-Theorem.