Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
Kennzahl: ______________	**Frühjahr 2011**	**66115**
Kennwort: ______________		
Arbeitsplatz-Nr.: __________		

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen
— Prüfungsaufgaben —

Fach: **Informatik (vertieft studiert)**

Einzelprüfung: **Theoret. Informatik, Algorithmen**

Anzahl der gestellten Themen (Aufgaben): **2**

Anzahl der Druckseiten dieser Vorlage: **12**

Bitte wenden!

Thema Nr. 1

Aufgabe 1:

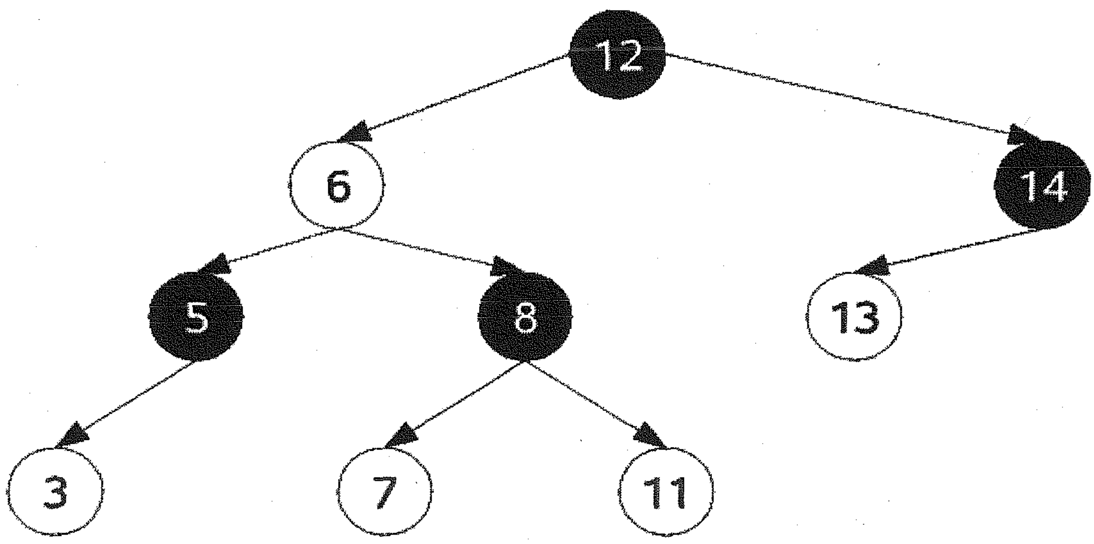
Bestimmen Sie mit Hilfe des Master-Theorems für die folgenden Rekursionsgleichungen möglichst scharfe asymptotische untere und obere Schranken, falls das Master-Theorem anwendbar ist!

Geben Sie andernfalls eine kurze Begründung, warum das Master-Theorem nicht anwendbar ist!

a) $T(n) = 16T(n/2) + 40n - 6$
b) $T(n) = 27T(n/3) + 3n^3 \log n$
c) $T(n) = 4T(n/2) + 3n^2 + \log n$
d) $T(n) = 4T(n/16) + 100 \log n + \sqrt{2n} + n^{-2}$

Fortsetzung nächste Seite!

Aufgabe 2:



a) Zeichnen Sie die beiden Rot-Schwarz-Bäume (die roten Knoten sind weiß gekennzeichnet), die entstehen, wenn man nacheinander die Schlüssel 2 und 1 in den obigen Baum gemäß dem Einfügealgorithmus für Rot-Schwarz-Bäume einfügt.

b) Zeichnen Sie die beiden Rot-Schwarz-Bäume, die entstehen, wenn man nacheinander die Schlüssel 7 und 6 aus dem in Teilaufgabe a) angegebenen Rot-Schwarz-Baum gemäß dem Löschalgorithmus für Rot-Schwarz- Bäume entfernt.

Hinweis: Falls Sie die Zwischenschritte geeignet dokumentieren, können auch für teilweise richtige Lösungen entsprechend Punkte vergeben werden.

Fortsetzung nächste Seite!

Aufgabe 3

Ein Weinkellner benutzt das folgende System zur Lagerung seiner Weine. Die n Flaschen ($n \geq 8$) werden in 3 Kategorien A, B und C eingeteilt:

Kategorie A soll möglichst wenige Flaschen enthalten, die zusammen mindestens 60% des Wertes ausmachen - falls dies möglich ist: Diese werden dann in einem speziellen Weinkühlschrank gelagert, dessen Kapazität $n/\log n$ Flaschen fasst. Falls die $n/\log n$ teuersten Flaschen also zusammen einen Wert von weniger als 60% haben, fallen eben die $n/\log n$ teuersten Flaschen in Kategorie A.

Kategorie C enthält die 60% der Flaschen, d. h. $\lfloor 0{,}6n \rfloor$ Stück, die den geringsten Wert haben. Diese sind zum alltäglichen Genuss bestimmt und werden aufrecht stehend im Vorratsschrank gelagert.

Die restlichen Flaschen bilden die **Kategorie B**, diese lagern im Keller im Weinregal.

Hinweis: $\lfloor x \rfloor$ bezeichnet die größte ganze Zahl, die kleiner oder gleich x ist.

Ein möglicher Algorithmus zur Klassifikation der Weinflaschen besteht darin, diese zuerst (möglichst effizient) nach ihrem Wert zu sortieren. Dann wird solange die jeweils wertvollste Flasche zur Kategorie A hinzugefügt, bis 60% des Wertes oder $n/\log n$ Flaschen erreicht sind. Anschließend wird $\lfloor 0{,}6n \rfloor$ mal die Flasche mit dem geringsten Wert in die Kategorie C eingereiht. Die verbleibenden Flaschen bilden Kategorie B.

a) Konstruieren Sie ein Beispiel mit $n = 8$ Flaschen, denen Sie Werte so zuordnen, dass mehr als 60% des Gesamtwertes in Kategorie A fallen.

b) Geben Sie einen Algorithmus zur Klassifikation an, der eine asymptotisch bessere Laufzeit im worst-case hat, und begründen Sie dies. Sie können bekannte Algorithmen und Datenstrukturen verwenden.

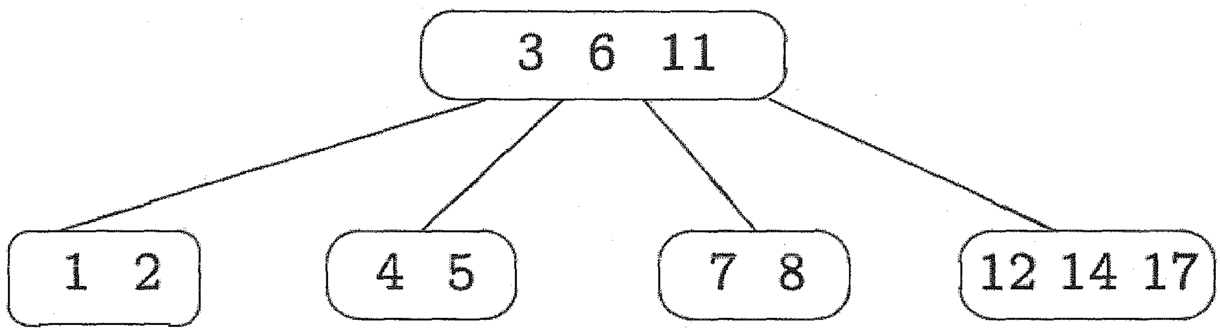
Fortsetzung nächste Seite!

Aufgabe 4: B-Bäume

a) Was ist ein B-Baum vom Grad m? Geben Sie alle Bedingungen an, die an so einen Baum gestellt werden.

b) Wofür werden B-Bäume verwendet?

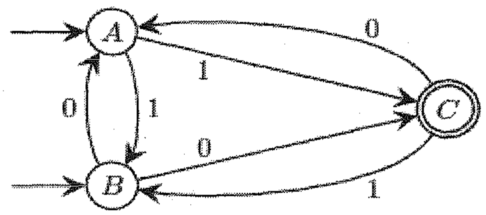
c) Fügen Sie in folgenden B-Baum vom Grad 2 das Element 42 ein und zeichnen Sie den entstehenden Baum.



d) Löschen Sie aus obigem Baum die 4 und die 5 und zeichnen Sie das Resultat.

Fortsetzung nächste Seite!

Aufgabe 5



Der endliche Automat N mit diesem Zustandsgraphen ist nichtdeterministisch. Konstruieren Sie einen deterministischen endlichen Automaten M, der dieselbe Sprache $L \subseteq \{0, 1\}^*$ akzeptiert wie N.

Vergessen Sie nicht, Start- und Endzustände von M *anzugeben.*

Fortsetzung nächste Seite!

Aufgabe 6

Für $z \in \Sigma^*$ sei $|z|_a$ die Anzahl der a und $|z|_b$ die Anzahl der b und $|z|_c$ die Anzahl der c im Wort z.

Sei $\Sigma = \{a, b, c\}$ und $L = \{ z \in \Sigma^* \mid |z|_a < |z|_b + |z|_c \}$.

Beweisen Sie, dass L nicht vom Typ 3 (regulär) ist.

Hinweis: Es gibt verschiedene Möglichkeiten für den Beweis.

Fortsetzung nächste Seite!

Aufgabe 7

Sei G = (V, Σ, P, S) mit Startsymbol S und V = {S, T, A, B} und Σ = {a, b} und

P = {S → AB, S → BT, A → BA, B → TT, A → a, B → b,

T → AB, T → a}

Sei L = L(G) die von dieser Grammatik erzeugte Sprache.

Sei w = baaab

a) Wenden Sie den Algorithmus von Cocke, Younger und Kasami (CYK) auf w an.

b	a	a	a	b
B	A, T	A, T	A, T	B

Es gilt w ∈ L. Woran sieht man das?

b) Mit dem Verfahren kann man zeigen, ob es einen Syntaxbaum mit Wurzel S für ein gegebenes Wort gibt. Tatsächlich hat w sogar mehrere verschiedene Syntaxbäume mit Wurzel S. Das Verfahren kann so modifiziert werden, dass es solche Fälle erkennt.

Fortsetzung nächste Seite!

Beschreiben Sie

i) eine *sehr kleine* dafür geeignete Modifikation des Verfahrens,

ii) das Kriterium, mit dem man im modifizierten Verfahren sieht, ob es *mehrere* Syntaxbäume mit Wurzel S für ein gegebenes Wort gibt.

iii) Geben Sie alle echten Teilwörter von w an, die mindestens zwei Syntaxbäume mit Wurzel S haben.

Aufgabe 8

Gegeben sei

	Multiplikation	absolute Differenz
$\begin{array}{llll} h: & \mathbb{N}\times\mathbb{N} & \rightarrow & \mathbb{N} \\ & (y,x) & \mapsto & \mathit{absdiff}(x, \mathit{mult}(y,y)) \end{array}$	$\begin{array}{llll} \mathit{mult}: & \mathbb{N}\times\mathbb{N} & \rightarrow & \mathbb{N} \\ & (x,y) & \mapsto & x\cdot y \end{array}$	$\begin{array}{llll} \mathit{absdiff}: & \mathbb{N}\times\mathbb{N} & \rightarrow & \mathbb{N} \\ & (x,y) & \mapsto & \lvert x-y\rvert \end{array}$

Sie können voraussetzen, dass mult *und* absdiff *primitiv rekursiv sind.*

Zeigen Sie:

h ist primitiv rekursiv.

Geben Sie zum Nachweis eine Definition von h an, die strikt nach den syntaktischen

Vorgaben des Kompositions- und/oder Rekursionsschemas für primitive Rekursion aufgebaut ist.

Thema Nr. 2

1. Aufgabe (reguläre Sprachen und endliche Automaten)

Die Elemente einer regulären Sprache können durch deterministische oder nicht-deterministische endliche Automaten erkannt werden.
Betrachten Sie folgenden nicht-deterministischen endlichen Automaten $A_1 = (\{q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \delta, q_1, \{q_4\})$ mit Zustandsmenge $\{ q_1, q_2, q_3, q_4, q_5 \}$, Eingangsalphabet $\{a, b\}$, Anfangszustand q_1 und Endzustandsmenge $\{q_4\}$.
Die Übergangsfunktion δ sei durch folgende Tabelle definiert, wobei $\in$ das leere Wort bezeichnet:

δ	q_1	q_2	q_3	q_4	q_5
a	$\{q_1, q_2\}$	$\{q_4\}$	$\emptyset$	$\{q_4\}$	$\emptyset$
b	$\{q_1\}$	$\emptyset$	$\{q_5\}$	$\{q_4\}$	$\{q_4\}$
$\in$	$\{q_3\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

a) Zeichnen Sie das Übergangsdiagramm des Automaten mit Zuständen und Übergangskanten.

b) Beschreiben Sie die von A_1 erkannte reguläre Sprache L_1, indem Sie eine mathematisch exakte Definition der Menge der erkannten Worte über $\{a, b\}$ angeben. Geben Sie einen möglichst kurzen regulären Ausdruck an, der die Sprache L_1 beschreibt.

c) Wandeln Sie den nicht-deterministischen endlichen Automaten A_1 in einen deterministischen endlichen Automaten A_2 um, indem Sie die Teilmengenkonstruktion anwenden.

d) Geben Sie eine Definition der Äquivalenz von Zuständen in deterministischen endlichen Automaten.

e) Bestimmen Sie alle äquivalenten Zustände von A_2. Bauen Sie dazu die vollständige Tabelle mit Zustandspaaren schrittweise auf und markieren Sie, ob die jeweiligen Zustände unterscheidbar sind. Erläutern Sie jeden durchgeführten Schritt. Fassen Sie anschließend die äquivalenten Zustände zusammen und konstruieren Sie den resultierenden deterministischen endlichen Automaten A_3, indem Sie für A_3 ein Übergangsdiagramm und eine tabellenförmige Darstellung der Übergangsfunktion angeben.

Fortsetzung nächste Seite!

2. Aufgabe (kontextfreie Sprachen und Kellerautomaten)

Betrachten Sie die kontextfreie Sprache $L = \{ww^R; w \in \{0,1\}^*\}$. Dabei bezeichnet w^R die Umkehrung des Wortes w, d.h. für $w = a_1 \ldots a_n$ ist $w^R = a_n \ldots a_1$.

(a) Beweisen Sie durch Anwendung des Pumping-Lemmas für reguläre Sprachen, dass die Sprache L nicht regulär ist. Begründen Sie die jeweiligen Schritte.

(b) Geben Sie eine kontextfreie Grammatik G mit Terminalsymbolen, Nichtterminalsymbolen und Produktionen an, die L erzeugt, d.h. $L = L(G)$. Geben Sie die Schritte zur Ableitung des Wortes $011110 \in L$ mit dieser Grammatik an.

(c) Konstruieren Sie einen nicht-deterministischen Kellerautomaten $K = (Q, \Sigma, \Gamma, \delta, q_0, z_0, f)$, der L erkennt, d.h. $L = L(K)$. Geben Sie eine genaue Definition aller Elemente des Kellerautomaten mit einer mathematisch exakten Definition der Übergangsrelation δ.
Erläutern Sie die Arbeitsweise des Kellerautomaten und begründen Sie, warum K alle Worte aus L erkennt.

(d) Erläutern Sie den Unterschied zwischen nicht-deterministischen und deterministischen Kellerautomaten durch Angabe der exakten Definitionen. Welche Unterschiede in den Verarbeitungsschritten gibt es?

(e) Gibt es Sprachen, für die ein nicht-deterministischer Kellerautomat K_1 konstruiert werden kann, der die jeweilige Sprache erkennt, für die aber kein deterministischer Kellerautomat existiert, der die Sprache erkennt?
Verwenden Sie die Sprache L für ihre Argumentation. Begründen Sie Ihre Antwort ausführlich.

(f) Konstruieren Sie einen deterministischen Kellerautomaten für die Sprache $L' = \{wcw^R] \; w \in \{0,1\}\}$ mit separatem Markierungszeichen $c \notin \{0,1\}$. Geben Sie eine mathematisch exakte Definition der Übergangsfunktion und erläutern Sie die Arbeitsweise des Kellerautomaten.

3. Aufgabe (Turingmaschinen)

(a) Konstruieren Sie eine deterministische Turingmaschine, die eine Eingabe $x \in \{0,1\}^*$ als Binärzahl interpretiert und die Binärdarstellung der Zahl produziert, die durch Addition von 1 entsteht.
Erläutern Sie die Rolle der Zustände der von Ihnen konstruierten Turingmaschine und geben Sie die Übergangsfunktion in Tabellenform an.
Illustrieren Sie die Arbeitsweise der von Ihnen konstruierten Maschine, indem Sie die Berechnungsschritte für die Eingabe $x = 101$ als Konfigurationsübergänge angeben.

(b) Erkennen deterministische Turingmaschinen dieselbe Sprachklasse wie nicht-deterministische Turingmaschinen oder gibt es Sprachen, die zwar von nicht-deterministischen, nicht aber von deterministischen Turingmaschinen erkannt werden können? Geben Sie eine ausführliche Begründung Ihrer Antwort. Geben Sie entweder eine Sprache an, die zwar von nicht-deterministischen, nicht aber von deterministischen Turingmaschinen erkannt werden kann, oder beschreiben Sie, wie eine beliebige nicht-deterministische Turingmaschine durch eine deterministische Turingmaschine simuliert werden kann.

Fortsetzung nächste Seite!

4. Aufgabe (Algorithmen und Datenstrukturen)

Rot-Schwarz-Bäume sind balancierte Bäume, die im Gegensatz zu unbalancierten Bäumen effiziente Sucheigenschaften auch für beliebige Einfügereihenfolgen garantieren.

(a) Geben Sie eine genaue Definition von Rot-Schwarz-Bäumen an, aus der insbesondere hervorgeht, wann ein Suchbaum ein Rot-Schwarz-Baum ist.

(b) Beweisen Sie, dass in einem Rot-Schwarz-Baum folgende Eigenschaft gilt: Ein Rot-Schwarz-Baum mit n inneren Knoten hat höchstens die Höhe $2 \cdot \log_2(n + 1)$.

(c) Erläutern Sie das effiziente Einfügen von Elementen in einem Rot-Schwarz-Baum mit Rotationen, indem Sie eine Prozedur in Pseudocodenotation angeben, die in einen beliebigen Rot-Schwarz-Baum ein Element einfügt, so dass der resultierende Baum weiterhin ein Rot-Schwarz-Baum ist.

(d) Illustrieren Sie die Arbeitsweise Ihrer in (c) entworfenen Einfügemethode, indem Sie in einen anfangs leeren Rot-Schwarz-Baum schrittweise die Zahlen 41, 38, 31, 12, 19, 8 in dieser Reihenfolge einfügen. Geben Sie den nach jedem Schritt resultierenden Rot-Schwarz-Baum an und erläutern Sie den Ablauf des Einfügens.

(e) Analysieren Sie die Laufzeit der von Ihnen angegebenen Einfügemethode.