Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
Kennzahl: ______ Kennwort: ______ Arbeitsplatz-Nr.: ______	**Frühjahr 2011**	**46115**

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen
— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

Einzelprüfung: **Theoret. Informatik, Algorith./Datenstr.**

Anzahl der gestellten Themen (Aufgaben): **1**

Anzahl der Druckseiten dieser Vorlage: **6**

Bitte wenden!

Thema Nr. 1

Teilaufgabe I Wir fixieren das Alphabet $\Sigma = \{\binom{0}{0}, \binom{0}{1}, \binom{1}{0}, \binom{1}{1}\}$ und definieren $L \subseteq \Sigma^*$ als die Sprache aller Wörter w mit der folgenden Eigenschaft:

- Es sei $w_1 \in \{0,1\}^*$ das Wort bestehend aus den „oberen" Einträgen von w und es sei $w_2 \in \{0,1\}^*$ das aus den „unteren" Einträgen von w bestehende Wort. Wenn etwa $w = \binom{0}{0}\binom{1}{1}\binom{1}{0}$, dann ist $w_1 = 011$ und $w_2 = 010$.
- Das Wort w ist in L genau dann, wenn w_1 als Binärzahl aufgefasst, um eins größer als der Wert von w_2 ist. So ist also das obige Beispielwort in L, da 011 den Wert 3 hat und 010 den Wert 2 hat.

1. Konstruieren Sie einen endlichen Automaten (egal, ob deterministisch oder nicht) für L^{rev}, also für die Sprache aller Wörter über Σ, welche von rechts nach links gelesen in L liegen. Hinweis: Man addiert eins zu einer Binärzahl, indem man die erste Null von rechts gesehen zur Eins umändert und alle davor (von rechts her gesehen) liegenden Einsen zu Nullen ändert.
2. Konstruieren Sie nun einen nichtdeterministischen endlichen Automaten für L selbst. Es bietet sich an, den Automaten aus Aufgabenteil 1 zu verwenden, Sie müssen das aber nicht tun.
3. Beschreiben Sie die Potenzmengenkonstruktion zur Konversion eines nichtdeterministischen in einen deterministischen Automaten.
4. Konstruieren Sie einen deterministischen endlichen Automaten für L.
5. Ermitteln Sie, ob Ihr Automat minimal ist. Falls ja, begründen Sie die Minimalität, falls nein, konstruieren Sie den Minimalautomaten. Wenn Sie Aufgabenteil 4 nicht lösen konnten, beschreiben Sie allgemein, wie man den Minimalautomaten konstruiert und wie man die Minimalität eines vorgelegten Automaten zeigen kann.

Teilaufgabe II

1. Begründen Sie, dass folgende Menge unentscheidbar ist:

   $$M = \{e \mid \text{die durch } e \text{ bezeichnete Turingmaschine hält bei Eingabe 37 nicht.}\}$$

2. Ist M partiell entscheidbar (Synonym: semi-entscheidbar, rekursiv aufzählbar)? (Begründung!)
3. Zeigen Sie durch Angabe eines Gegenbeispiels, dass folgende Aussage unwahr ist: Wenn L_1 und L_2 unentscheidbar sind, dann ist auch $L_1 \cup L_2$ unentscheidbar.

Fortsetzung nächste Seite!

Teilaufgabe III

1. Fügen Sie in einen anfangs leeren 2-3-4 Baum (B-Baum der Ordnung 4) der Reihe nach die folgenden Schlüssel ein: $1, 2, 3, 5, 7, 8, 9, 4, 11, 12, 13, 6$. Dokumentieren Sie die Zwischenschritte so, dass die Entstehung des Baumes und nicht nur das Endergebnis nachvollziehbar ist.

2. Zeichnen Sie einen Rot-Schwarz-Baum *oder* einen AVL-Baum, der dieselben Einträge enthält.

3. Geben Sie eine möglichst gute untere Schranke (in Ω-Notation) für die Anzahl der Schlüssel in einem 2-3-4-Baum der Höhe h an. Hinweis: Überlegen Sie sich, wie ein 2-3-4 Baum mit Höhe h und möglichst wenigen Schlüsseln aussieht.

4. Geben Sie eine möglichst gute obere Schranke (in O-Notation) für die Anzahl der Schlüssel in einem 2-3-4-Baum der Höhe h an.

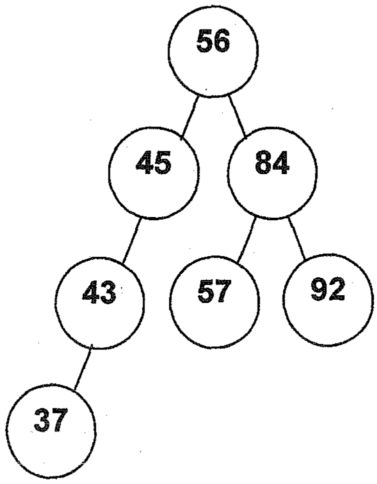
5. Für welche $a \in \mathbb{R}$ gilt: $4^{an} = O(2^n)$?

6. Für welche $a \in \mathbb{R}$ gilt: $n^2 = \Omega(44an^a)$?

Thema Nr. 2

Aufgabe 1

Als spezielle binäre Bäume werden in der Informatik oft AVL-Bäume eingesetzt. Gegeben sei folgender AVL-Baum, in den zuletzt der Knoten mit dem Wert 37 eingefügt wurde.



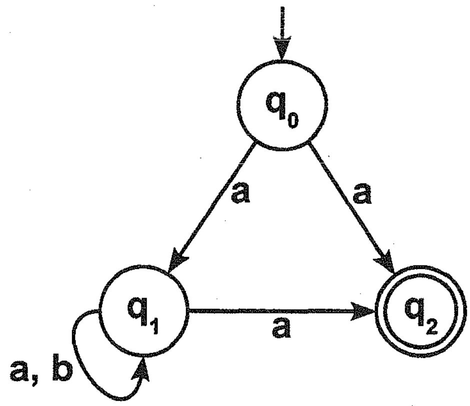
a) Erläutern Sie, warum jetzt die AVL-Eigenschaft des Baumes verletzt ist, und stellen Sie diese Eigenschaft durch geeignete Maßnahmen wieder her. Erläutern Sie Ihr Vorgehen.

b) Fügen Sie nun schrittweise Knoten mit den Werten 21, 42 und 41 in dieser Reihenfolge in den AVL-Baum ein und stellen Sie gegebenenfalls jeweils die AVL-Eigenschaft durch geeignete Maßnahmen wieder her.

Fortsetzung nächste Seite!

Aufgabe 2

Gegeben sei folgender endlicher Automat:



a) Überführen Sie diesen nichtdeterministischen Automaten in einen deterministischen endlichen Automaten.

b) Geben Sie die Sprache, die dieser Automat erkennt, als möglichst kurzen regulären Ausdruck an.

Aufgabe 3

Sei $L = \{a^{2n}b^{3n} | n > 0\}$ eine Sprache über dem Alphabet $\Sigma = \{a,b\}$.

a) Geben Sie eine kontextfreie Grammatik G an, die L erzeugt.

b) Geben Sie eine Ableitung des Wortes aaaabbbbbb in der Grammatik G an.

c) Begründen Sie ausführlich, dass es keine reguläre Grammatik gibt, die L erzeugt.

Betrachten Sie die Sprachen $L_1 = \{ww^R \mid w \in \Sigma^*\}$ und $L_2 = \{ww \mid w \in \Sigma^*\}$ über dem Alphabet Σ.

d) Ordnen Sie die Sprachen L_1 und L_2 möglichst exakt in die Chomsky Hierarchie ein.

e) Beschreiben Sie informell die Arbeitsweise der L_1 und L_2 akzeptierenden Automaten.

f) Begründen Sie insbesondere, warum L_1 und L_2 nicht in die jeweils niedrigere Hierarchiestufe eingeordnet werden können (Typ 3 sei die niedrigste, Typ 0 die höchste Stufe).

Begründen oder widerlegen Sie die folgenden Aussagen:

g) Die Schnittmenge zweier semi-entscheidbarer Sprachen ist ebenfalls semi-entscheidbar.

h) Die Vereinigung zweier semi-entscheidbarer Sprachen ist ebenfalls semi-entscheidbar.

i) Das Komplement einer semi-entscheidbaren Sprache ist ebenfalls semi-entscheidbar.

j) Die Differenz zweier semi-entscheidbarer Sprachen ist ebenfalls semi-entscheidbar.

Fortsetzung nächste Seite!

Aufgabe 4

Eine Datenstruktur zu *augmentieren* bedeutet, in der Datenstruktur zusätzliche Information zu speichern, um neue Typen von Anfragen effizient beantworten zu können. In dieser Aufgabe geht es darum, Ihnen bekannte Datenstrukturen so zu augmentieren, dass man schnell den Median der gespeicherten Menge von Zahlen bestimmen kann. Zur Erinnerung: Der Median einer Menge $\{a_1, a_2, \ldots, a_n\}$ von Zahlen mit $a_1 < a_2 < \cdots < a_n$ ist das Element $a_{\lfloor n/2 \rfloor}$.

Gehen Sie in beiden Teilaufgaben der Einfachheit halber davon aus, dass eine Zahl zu jedem Zeitpunkt höchstens einmal in der Liste gespeichert ist.

a) Gehen Sie davon aus, dass Sie eine korrekte Implementierung einer *sortierten*, doppelt verketteten Liste vorliegen haben. Die Liste besitzt einen Zeiger *head*, der auf das erste Element der Liste zeigt. Jedes Element verfügt über einen Schlüssel *key* – die zu speichernde Zahl – und über zwei Zeiger. Der Zeiger *next* zeigt auf das folgende Element, falls ein solches existiert; ansonsten ist *next* der Nullzeiger *nil*. Entsprechend zeigt *prev* auf das vorhergehende Element, falls ein solches existiert; ansonsten ist *prev* = *nil*.

Die Liste sucht beim Einfügen einer Zahl ihren Platz in der Sortierung und gibt dann einen Zeiger auf das neu angelegte Listenelement zurück, in dem die Zahl gespeichert wurde. Mithilfe dieses Zeigers kann der Benutzer die Zahl später in konstanter Zeit aus der Liste löschen.

Ihre Aufgabe besteht darin anzugeben, welche zusätzliche Information wo und wie aufrechterhalten werden soll, wenn der Benutzer Zahlen in die Liste einfügt oder aus der Liste löscht. Geben Sie dazu in Pseudocode-Schreibweise an, welche Anweisungen im Anschluss an die Einfüge- bzw. die Löschoperation der zugrundeliegenden Liste ausgeführt werden sollen. Stellen Sie sicher, dass eine Löschoperation auch mit Ihrer Augmentierung nur konstante Zeit dauert. Das Einfügen darf lineare Zeit dauern.

b) Zeigen Sie, wie man eine andere Datenstruktur augmentieren kann, um nach wie vor schnellen Zugriff auf den Median einer sich ändernden Menge von Zahlen zu haben – nun aber so, dass die Einfügeoperation asymptotisch schneller ist als bei der Liste in Teilaufgabe (a). Genauer gesagt soll die augmentierte Datenstruktur die drei Operationen Einfügen, Löschen und Zugriff auf den Median in jeweils sublinearer Zeit ermöglichen.