Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
Kennzahl: ________ Kennwort: ________ Arbeitsplatz-Nr.: ________	**Herbst 2014**	**46115**

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

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

Anzahl der gestellten Themen (Aufgaben): 2

Anzahl der Druckseiten dieser Vorlage: 8

Bitte wenden!

Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1:

Sei L die Menge der nicht-leeren Wörter über dem Alphabet $\{a, b, c\}$, bei denen das erste und das letzte Zeichen verschieden sind.

a) Geben Sie einen regulären Ausdruck an, der L beschreibt.

b) Geben Sie einen deterministischen endlichen Automaten A an mit $L(A) = L$. Hier reicht z. B. das Zustandsdiagramm.

Aufgabe 2:

Sei $L = \{a^m b^n c^n d^m \mid n, m \geq 1\}$.

a) Geben Sie eine textuelle Beschreibung für L der Form „L besteht aus allen Wörtern, die...“.

b) Zeigen oder widerlegen Sie, dass L regulär ist.
Begründen (Beweisen) Sie Ihre Antwort durch die Angabe einer entsprechenden Beschreibung für L oder zeigen Sie, dass L nicht regulär sein kann.

Aufgabe 3:

Zeigen Sie, dass es zu jedem deterministischen endlichen Automaten (DFA) A einen DFA B gibt mit $L(B) = \Sigma^* - L(A)$, d.h., B erkennt das Komplement von A.

Aufgabe 4:

Zeigen Sie, dass die Sprache $L = \{a^k b^k a^m b^m a^n b^n \mid k, n, m \geq 1\}$ kontextfrei ist.

Aufgabe 5:

Geben Sie eine Turingmaschine M an, die die Sprache $L = \{a^{n^2} \mid n \geq 1\}$ erkennt und beschreiben Sie in Worten, wie Ihre Turingmaschine arbeitet.
Tipp: n^2 ist die Summe der ungeraden Zahlen von 1 bis $2n - 1$.

Aufgabe 6:

Fügen Sie nacheinander die Zahlen 10, 4, 20, 12, 5, 7, 1, 2, 8, 9

a) in einen leeren binären Suchbaum ein und zeichnen Sie den Suchbaum.

b) in einen leeren AVL Baum ein.
Beschreiben (zeichnen) Sie den AVL-Baum nach dem Einfügen von 5 und am Ende.
Beschreiben Sie, wann ggf. rotiert werden muss, und geben Sie die Rotationen an.

Fortsetzung nächste Seite!

Aufgabe 7:

In einen bzgl. $\leq$ angeordneten leeren Min-Heap werden nacheinander

a) die Zahlen 1, 18, 3, 20, 22, 7, 30, 26, 16 eingefügt.
Geben Sie den Min-Heap vor und nach dem Einfügen von 16 an.

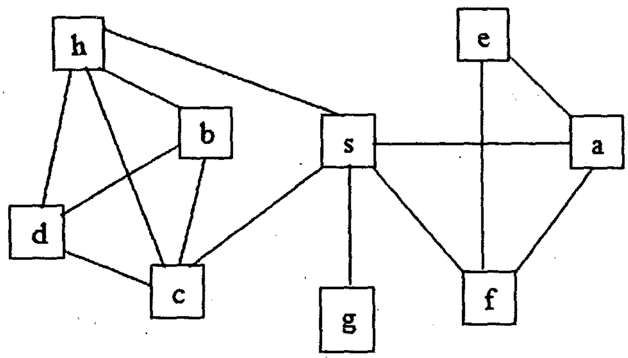
b) Anschließend wird mit deletemin() die 1 gelöscht.
Beschreiben Sie, wie das realisiert wird, und geben Sie den Heap danach an.

Aufgabe 8:

Führen Sie auf dem folgenden ungerichteten Graphen G eine Tiefensuche ab dem Knoten s aus. Unbesuchte Nachbarn eines Knotens sollen dabei in alphabetischer Reihenfolge abgearbeitet werden.

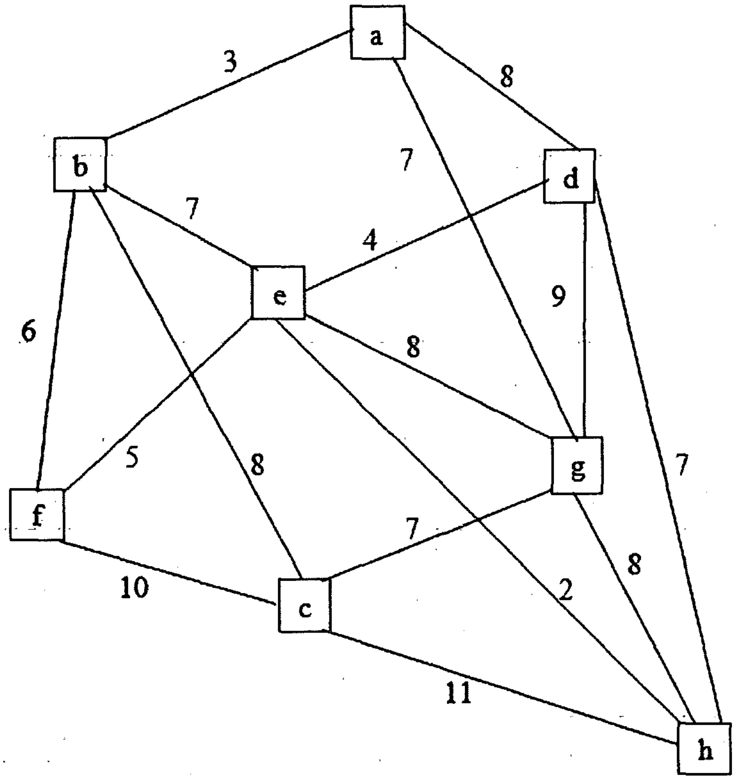
Die Tiefensuche soll auf Basis eines Stacks implementiert werden.

Geben Sie die Reihenfolge der besuchten Knoten, also die dfs-number der Knoten, und den Inhalt des Stacks in jedem Schritt an.



Aufgabe 9:

Berechnen Sie einen minimalen Spannbaum im Graphen G.
Markieren Sie die zum Spannbaum gehörenden Kanten.
Welchen Wert hat der Spannbaum?



Thema Nr. 2
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

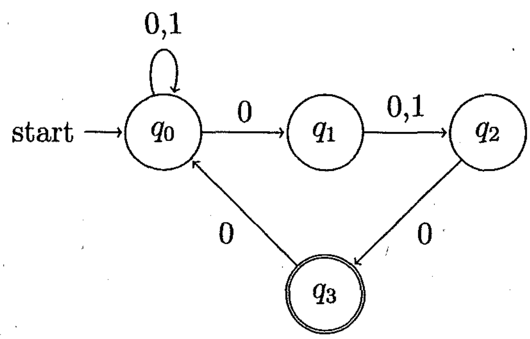
Aufgabe 1:
Reguläre Sprachen I

a) Sei $\Sigma = \{a, b\}$. Für ein $w \in \Sigma^*$ und $x \in \Sigma$ bezeichnen wir mit $|w|_x$ die Anzahl der x in w. Sei $L = \{w \in \Sigma^* \mid |w|_a \text{ ungerade}\}$.

   i) Geben Sie einen regulären Ausdruck α mit $L(\alpha) = L$ an.

   ii) Eine Grammatik heißt *linkslinear*, wenn es für jede Produktion Nichtterminale A, B und ein Terminalsymbol x gibt, so dass die Produktion die Form $A \to Bx$ oder $A \to \varepsilon$ hat. Geben Sie eine linkslineare Grammatik für die Sprache L an.

b) Geben Sie einen deterministischen endlichen Automaten an, der die gleiche Sprache erkennt, wie der unten aufgeführte nichtdeterministische Automat. Erläutern Sie, wie Ihre Konstruktion mit dem ursprünglichen Automaten zusammenhängt.



Aufgabe 2:
Reguläre Sprachen II

a) Erläutern Sie kurz, warum reguläre Sprachen unter Schnitt abgeschlossen sind!

Sei nun $\Sigma = \{a, b, c\}$. Sie dürfen als gegeben annehmen, dass die Sprache $L_0 = \{a^i b^i \mid i \in \mathbb{N}\}$ nicht regulär ist.

b) Ist $L_b = \{a^i b^i c^j \mid i, j \in \mathbb{N}_0\}$ regulär? Zeigen Sie, warum (nicht)!

c) Ist $L_c = \{a^i b^j c^k \mid i, j, k \in \mathbb{N}_0\}$ regulär? Zeigen Sie, warum (nicht)!

Fortsetzung nächste Seite!

Aufgabe 3:
Kontextfreie Sprachen

a) Geben Sie eine kontextfreie Grammatik für die folgende Sprache an:

$$L = \{(ba)^n b^m a^{2n} \mid n \geq 0, m \text{ gerade}\}$$

b) Betrachten Sie die Grammatik $G_2 = (N_2, \Sigma_2, P_2, S)$ mit $N_2 = \{S, T, U, V, W, A, B, C, D\}$ und $\Sigma_2 = \{a, b, c\}$, wobei P_2 aus den folgenden Produktionen besteht:

$$\begin{array}{lllll} S \to SS \mid AV & T \to UU \mid CW \mid d & U \to UU \mid d & V \to TB & W \to TD \\ A \to a & B \to b & C \to c & D \to d & \end{array}$$

Verwenden Sie den CYK-Algorithmus, um zu zeigen, dass das Wort $w_2 = adbb$ nicht in $L(G_2)$ enthalten ist.

Aufgabe 4:
Komplexitätstheorie

In einem ungerichteten Graphen $G = (V, E)$ ist eine *Clique* der Größe k eine Teilmenge $V' \subseteq V$ mit $|V'| = k$, sodass alle $u, v \in V'$ benachbart sind, d.h. $\{u, v\} \in E$.

Wir betrachten das 2-Cliquenproblem 2-CLIQUE, welches für einen ungerichteten Graphen G und natürliche Zahl k die Frage stellt, ob G zwei disjunkte Cliquen der Größe mindestens k enthält.

Formal ist das Problem 2-CLIQUE wie folgt definiert:

Gegeben: Ungerichteter Graph $G = (V, E)$ und natürliche Zahl $k \in \mathbb{N}$.
Problem: Existieren Cliquen $V_1, V_2 \subseteq V$ mit $|V_1| \geq k$, $|V_2| \geq k$ und $V_1 \cap V_2 = \emptyset$

a) Begründen Sie: 2-CLIQUE liegt in NP.

b) Geben Sie eine polynomielle Reduktion von CLIQUE auf 2-CLIQUE an. Das Problem CLIQUE ist die Frage, ob ein Graph eine Clique der Größe mindestens k besitzt.

c) Zeigen Sie: 2-CLIQUE ist NP-vollständig.

Hinweis: Sie dürfen verwenden, dass CLIQUE *NP-hart ist.*

Aufgabe 5:
Berechen- und Entscheidbarkeit

Sei $\Sigma = \{0, 1\}$. Wir nehmen an, dass jedes $w \in \Sigma^*$ eine Turingmaschine kodiert und bezeichnen diese Turingmaschine mit M_w. Die von M_w berechnete Funktion bezeichnen wir mit φ_w.

Welche der folgenden Mengen sind entscheidbar? Begründen Sie kurz.

a) $L_1 = \{w \in \Sigma^* \mid M_w \text{ besitzt mindestens 13 Zustände}\}$

b) $L_2 = \{w \in \Sigma^* \mid \text{es gibt } x, \text{ so dass } \varphi_w(x) = 01\}$

c) $L_3 = \{w \in \Sigma^* \mid \exists n \in \mathbb{N} : w = 0^n 1^n 0^n\}$

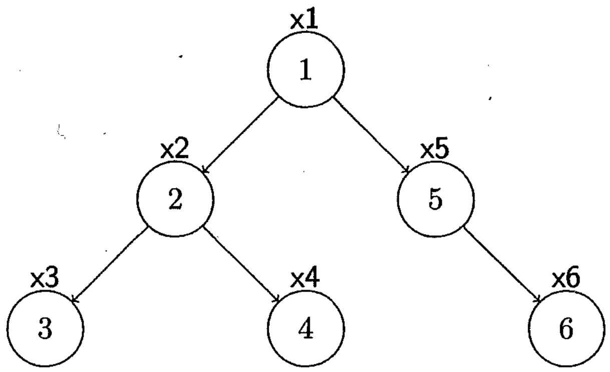
Fortsetzung nächste Seite!

Aufgabe 6:
Algorithmen und Datenstrukturen

Für diese Aufgabe nehmen wir eine Datenstruktur für Knoten in Binärbäumen an: Ein Knoten **node** besitzt Zeiger auf sein linkes bzw. rechtes Kind **node.left** bzw. **node.right** sowie einen Wert node.value $\in \mathbb{N}$. Zeiger auf **nil** markieren Blätter.

Wir zeichnen Knoten als Kreise, in welche wir die Werte schreiben. Optional geben wir den Namen des Knoten über dem Kreis an. Pfeile symbolisieren Zeiger auf Kinder. Der Einfachheit halber ignorieren wir Blätter.

Im nachfolgend abgebildeten Baum gilt zum Beispiel x5.left = **nil**, x5.right = x6 und x5.value = 5.



Die Methode **print**(n) gibt die natürliche Zahl n aus. Nachfolgend abgebildeter Pseudocode liefert mit Aufruf von traverse(x1) die Ausgabe 1, 2, 3, 4, 5, 6.

```
procedure traverse(node) {
  print node.value
  if (node.left ≠ nil)
    traverse (node.left)
  if (node.right ≠ nil)
    traverse (node.right)
}
```

a) Binäre Suchbäume:

i) Modifizieren Sie die Werte im oben abgebildeten Baum so, dass er einen binären Suchbaum für die Werte $\{1, 2, 3, 4, 5, 6\}$ darstellt. (Zeichnen Sie diesen Baum!)

ii) Geben Sie eine Pseudocode-Implementierung für **procedure** contains(x, n) an, welche **true** zurückliefert, falls sich der Wert n in dem binären Suchbaum mit Wurzelknoten x befindet, und **false** andernfalls.

iii) Wie hängt die Laufzeit Ihrer Implementierung mit der Höhe des Suchbaums zusammen? *Hinweis: Die Höhe ist die Anzahl der Knoten auf einem längsten Pfad zu einem Blatt, im obigen Beispielbaum* 3

iv) Wie viele Knoten kann ein binärer Suchbaum der Höhe n maximal enthalten? Wie viele minimal?

Fortsetzung nächste Seite!

b) Baumtraversierung:

i) Zeichnen Sie einen Baum, der sich strukturell vom oben abgebildeten Baum unterscheidet, für dessen Wurzel allerdings traverse (wie im oben gegebenen Beispiel) ebenfalls die Ausgabe $1, 2, 3, 4, 5, 6$ liefert.

ii) Modifizieren Sie traverse so zu traverse_sorted, dass die Werte in einem binären Suchbaum *sortiert* ausgegeben werden. Verwenden sie dabei *keinen* der Vergleichsoperatoren $\leq$ und $\geq$!