Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Frühjahr
2019

66115

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:	8

Bitte wenden!

Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1

Antworten Sie auf die folgenden Behauptungen mit Wahr/Falsch und geben Sie eine kurze Begründung an.

(a) Wenn L_2 regulär ist und $L_1 \subseteq L_2$ gilt, dann ist L_1 auch regulär.

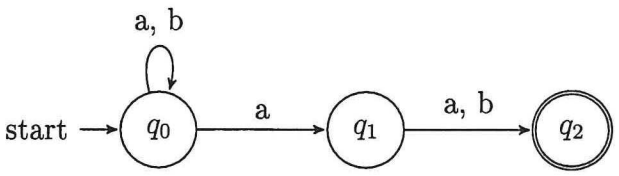
(b) $Q = \{a^q \mid \exists i \in \mathbb{N}.\ q = i^2\}$ ist bekanntlich nicht regulär.
Behauptung: Q^* ist ebenfalls nicht regulär.

(c) Wenn $L \subseteq \Sigma^*$ entscheidbar ist, dann ist auch das Komplement $\bar{L} = \Sigma^* \setminus L$ entscheidbar.

(d) Jedes $\mathcal{NP}$-vollständige Problem ist entscheidbar.

Aufgabe 2

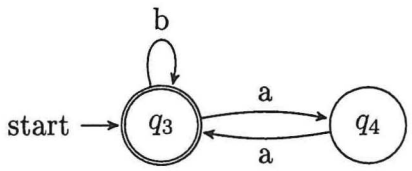
(a) Gegeben sei der nichtdeterministische endliche Automat A über dem Alphabet $\Sigma = \{a, b\}$ wie folgt:



Konstruieren Sie einen deterministischen endlichen Automaten, der das Komplement $\overline{L(A)} = \{w \in \Sigma^* \mid w \notin L(A)\}$ der von A akzeptierten Sprache $L(A)$ akzeptiert.

Fortsetzung nächste Seite!

(b) Gegeben sei zudem der nichtdeterministische Automat B über dem Alphabet $\Sigma = \{a, b\}$:



Konstruieren Sie einen endlichen Automaten (möglicherweise mit ε-Übergängen), der die Sprache $(L(A)L(B))^* \subseteq \Sigma^*$ akzeptiert (A aus der vorigen Aufgabe). Erläutern Sie auch Ihre Konstruktionsidee.

Aufgabe 3

Gegeben sei die kontextfreie Grammatik $G = (V, \Sigma, P, S)$ mit Sprache $L(G)$, wobei $V = \{S, T, U\}$ und $\Sigma = \{a, b\}$. P bestehe aus den folgenden Produktionen:

$$S \to TUUT \qquad T \to aT \mid \varepsilon \qquad U \to bUb \mid a$$

(a) Geben Sie fünf verschiedene Wörter $w \in \Sigma^*$ mit $w \in L(G)$ an.

(b) Geben Sie eine explizite Beschreibung der Sprache $L(G)$ an.

(c) Bringen Sie G in Chomsky-Normalform und erklären Sie Ihre Vorgehensweise.

Aufgabe 4

Wir betrachten die Turingmaschine $M = (Q, q_0, F, \Sigma, \Gamma, \square, \delta)$. Hierbei ist die Zustandsmenge $Q = \{q_0, q_a, q_b, q_L, q_f\}$ mit Startzustand q_0 und akzeptierenden Zuständen $F = \{q_f\}$. Das Eingabealphabet ist $\Sigma = \{a, b, \$\}$, das Bandalphabet ist $\Gamma = \Sigma \cup \{\square\}$ mit Blank-Zeichen $\square$ für leeres Feld. Die Übergangsfunktion $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$, wobei der Schreib-Lese-Kopf mit L nach links, mit N nicht und mit R nach rechts bewegt wird, ist durch folgende Tabelle gegeben (bspw. ist $\delta(q_0, a) = (q_a, \square, R)$):

	a	b	\$	$\square$
q_0	$(q_a, \square, R)$	$(q_b, \square, R)$	$(q_f, \$, N)$	$(q_f, \square, N)$
q_a	(q_a, a, R)	(q_a, b, R)	$(q_a, \$, R)$	(q_L, a, L)
q_b	(q_b, a, R)	(q_b, b, R)	$(q_b, \$, R)$	(q_L, b, L)
q_L	(q_L, a, L)	(q_L, b, L)	$(q_L, \$, L)$	$(q_0, \square, R)$

Fortsetzung nächste Seite!

(a) Die Notation (v, q, aw) beschreibt eine Konfiguration der Turingmaschine: der interne Zustand ist q, der Schreib-Lesekopf steht auf einem Feld mit $a \in \Gamma$, rechts vom Schreib-Lesekopf steht $w \in \Gamma^*$, links vom Schreib-Lesekopf steht $v \in \Gamma^*$.

Vervollständigen Sie die Folge von Konfigurationen, die die Turingmaschine bei Eingabe $ab\$$ bis zum Erreichen des Zustands q_f durchläuft. Sie können auch Ihre eigene Notation zur Darstellung von Konfigurationen verwenden.

$$(\varepsilon, q_0, ab\$) \longrightarrow$$
$$(\varepsilon, q_a, b\$) \longrightarrow$$
$$(b, q_a, \$) \longrightarrow$$
$$\ldots$$

(b) Sei $w \in \{a, b\}^*$ beliebig. Mit welchem Bandinhalt terminiert die Turingmaschine bei Eingabe von $w\$$? Geben Sie auch eine kurze Begründung an.

Aufgabe 5

Wir betrachten das Behälterproblem BEHAELTER. Gegeben ist eine Menge von $k \in \mathbb{N}$ Behältern, die jeweils ein Fassungsvermögen der Größe $b \in \mathbb{N}$ haben. Gegeben sind weiterhin n Objekte mit jeweiligen Größen $a_1, \ldots, a_n$. Gesucht ist eine Zuordnung der n Objekte auf die k Behälter, sodass keiner der Behälter überläuft.

Formal sind Instanzen des Behälterproblems BEHAELTER durch Tupel $(k, b, a_1, \ldots a_n)$ gegeben, die wie folgt zu interpretieren sind:

- $k \in \mathbb{N}$ steht für eine Anzahl von Behältern.
- Jeder Behälter hat ein Fassungsvermögen von $b \in \mathbb{N}$.
- Die a_i stehen für die jeweiligen Größen von n Objekten.

Zuordnungen von Objekten zu Behältern geben wir durch eine Funktion v an, wobei $v(j) = i$ wenn das j-te Objekt (mit Größe a_j) dem i-ten Behälter zugeordnet wird.

$(k, b, a_1, \ldots a_n)$ ist eine JA-Instanz von BEHAELTER, wenn es eine Zuordnung v von Objekten auf Behälter $(v : [1; n] \to [1; k])$ gibt, die sicherstellt, dass kein Behälter überläuft:

$$(k, b, a_1, \ldots a_n) \in \text{BEHAELTER} \iff (\exists v : [1; n] \to [1; k].\ \forall i \leq k.\ \sum_{i=v(j)} a_j \leq b)$$

Wir betrachten auch das modifizierte Problem GERADEBEHAELTER. Instanzen von GERADEBEHAELTER tragen die zusätzliche Einschränkung, dass alle a_j gerade (durch zwei teilbar) sein müssen.

Fortsetzung nächste Seite!

(a) Warum ist sowohl BEHAELTER $\in \mathcal{NP}$ als auch GERADEBEHAELTER $\in \mathcal{NP}$?

(b) Beweisen Sie, dass das Problem BEHAELTER auf das Problem GERADEBEHAELTER in polynomieller Zeit reduzierbar ist.

(c) BEHAELTER ist $\mathcal{NP}$ -vollständig. Begründen Sie, was obige Reduktion für die Komplexität von GERADEBEHAELTER bedeutet.

Aufgabe 6 (Algorithmen und Datenstrukturen)

Aus dem Känguru-Wettbewerb 2017 – Klassenstufen 3 und 4.

> Luna hat für den Kuchenbasar Muffins mitgebracht: 10 Apfelmuffins, 18 Nussmuffins, 12 Schokomuffins und 9 Blaubeermuffins. Sie nimmt immer 3 verschiedene Muffins und legt sie auf einen Teller. Welches ist die kleinste Zahl von Muffins, die dabei übrig bleiben können?
>
> A: 1, B: 3, C: 4, D: 7, E: 8

(a) Geben Sie die richtige Antwort auf die im Känguru-Wettbewerb gestellte Frage und begründen Sie sie.

(b) Lunas Freundin empfiehlt den jeweils nächsten Teller immer aus den drei aktuell häufigsten Muffinsorten zusammenzustellen. Leiten Sie aus dieser Idee einen effizienten Greedy-Algorithmus her, der die Fragestellung für beliebige Anzahlen von Muffins löst (nach wie vor soll es nur vier Sorten und je drei pro Teller geben). Skizzieren Sie in geeigneter Form, wie Ihr Algorithmus die Beispielinstanz von oben richtig löst.

(c) Beschreiben Sie eine mögliche und sinnvolle Verallgemeinerung Ihrer Lösung auf n Muffinsorten und k Muffins pro Teller für $n > 4$ und $k > 3$.

(d) Diskutieren Sie, wie man die Korrektheit des Greedy-Algorithmus zeigen könnte, also dass er tatsächlich immer eine optimale Lösung findet. Ein kompletter, rigoroser Beweis ist nicht verlangt.

Thema Nr. 2
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1 (k-kleinste Elemente)

Gegeben sei eine unsortierte Liste von n verschiedenen natürlichen Zahlen. Das k-kleinste Element ist das Element, das größer als genau $k-1$ Elemente der Liste ist.

(a) Geben Sie einen Algorithmus mit Laufzeit $\mathcal{O}(n \log n)$ an, um das k-kleinste Element zu berechnen.

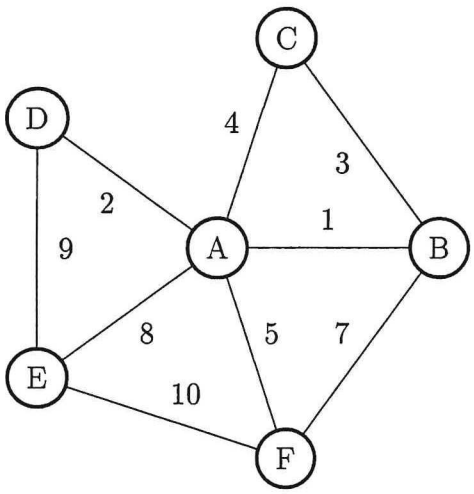
(b) Gegeben sei nun ein Algorithmus $\mathcal{A}$, der den Median einer unsortierten Liste von n Zahlen in $\mathcal{O}(n)$ Schritten berechnet. Nutzen Sie Algorithmus $\mathcal{A}$ um einen Algorithmus $\mathcal{B}$ anzugeben, welcher das k-kleinste Element in $\mathcal{O}(n)$ Schritten berechnet.

Argumentieren Sie auch, dass der Algorithmus die gewünschte Laufzeit besitzt.

(c) Geben Sie einen Algorithmus an, der für alle $i = 1, \ldots, \lfloor n/k \rfloor$ das $i \cdot k$-kleinste Element berechnet. Die Laufzeit Ihres Algorithmus sollte $\mathcal{O}(n \log(n/k))$ sein. Sie dürfen weiterhin Algorithmus $\mathcal{A}$, wie in Teilaufgabe (b) beschrieben, nutzen.

Aufgabe 2 (Spannbäume)

(a) Berechnen Sie zu dem angegebenen Graph einen minimalen Spannbaum. Nutzen Sie den Algorithmus von Kruskal. Geben Sie tabellarisch mit jedem Schritt des Algorithmus an, welche Kanten dem Baum bereits hinzugefügt wurden.



Fortsetzung nächste Seite!

(b) Der *Durchmesser* eines Spannbaums ist die Länge des längsten einfachen Pfads im Baum.

Geben Sie ein möglichst effizientes Verfahren an, den minimalen Spannbaum mit Durchmesser 2 zu bestimmen, falls ein solcher existiert. Analysieren Sie die asymptotische worst-case Laufzeit Ihres Algorithmus in Abhängigkeit der Kanten- und Knotenanzahl.

(c) Geben Sie nun ein effizientes Verfahren an, den minimalen Spannbaum mit Durchmesser 3 zu bestimmen, falls ein solcher existiert. Analysieren Sie die asymptotische worst-case Laufzeit Ihres Algorithmus.

Aufgabe 3 (Rotierende Bäume)

(a) Ein Binärbaum ist eine rechtslineare Kette, falls kein Knoten im Baum ein linkes Kind besitzt.

Zeigen Sie, wie ein beliebiger Binärbaum durch höchstens n (n=Anzahl der Elemente im Baum) Rotationen in eine rechtslineare Kette umgewandelt werden kann.

(b) Gegeben sei ein Algorithmus wie in Teil (a) gefordert. Zeigen Sie, dass ein Binärbaum durch $\mathcal{O}(n)$ Rotationen in einen beliebigen anderen Binärbaum gleicher Knotenanzahl umgewandelt werden kann.

(c) Zeigen Sie, dass es nicht möglich ist, einen binären Heap in $\mathcal{O}(n)$ Operationen, die nicht notwendigerweise Rotationen sind, in einen binären Suchbaum umzuwandeln.

Hinweis: Ihre Lösung sollte nicht länger als 4 Sätze sein.

Aufgabe 4 (O-Notation)

Zeigen oder widerlegen Sie die folgenden Aussagen zur O-Notation.

(a) $2^n \in \Theta(3^n)$.

(b) $2^{2\log_2 n} \in \mathcal{O}(n)$.

Fortsetzung nächste Seite!

Aufgabe 5 (Formale Sprachen und Komplexität)

Wie Sie wissen, ist der Schnitt zweier kontextfreier Sprachen nicht notwendigerweise selbst kontextfrei und es ist unentscheidbar, ob der Schnitt zweier durch Grammatiken gegebener kontextfreier Sprachen leer ist.

Das Shuffle-Produkt $L_1 \star L_2$ zweier Sprachen L_1, L_2 über dem Alphabet Σ ist bekanntlich definiert durch

$$L_1 \star L_2 = \{u_1 v_1 u_2 v_2 \dots u_n v_n \mid u_1 u_2 \dots u_n \in L_1, v_1 v_2 \dots v_n \in L_2 \text{ mit } u_1, ..., u_n, v_1, ..., v_n \in \Sigma^*\}$$

Die Sprache $L_1 \star L_2$ enthält also alle Wörter, die man durch Verzahnen eines Wortes in L_1 mit einem Wort in L_2 erhalten kann.

Beispiel:

$$\{aab, abab\} \star \{aa\} = \{aaaab, aaaba, aabaa, aaabab, aabaab, aababa, abaaab, abaaba, ababaa\}$$

Shuffle-Produkte spielen bei der Verifikation nebenläufiger Programme eine wichtige Rolle.

(a) Geben Sie einen regulären Ausdruck für das Shuffle-Produkt der (selbst durch reguläre Ausdrücke gegebenen) Sprache a^*b^* mit der Sprache c^*d^* an.

(b) Zeigen Sie unter Verwendung des o.a. Resultats über Schnitte kontextfreier Sprachen, dass folgendes Problem unentscheidbar ist: Gegeben sind zwei kontextfreie Sprachen L_1, L_2 über dem Alphabet $\Sigma = \{a, b\}$ durch Grammatiken, sowie eine reguläre Sprache L über Σ durch regulären Ausdruck. Gefragt ist, ob $(L_1 \star L_2) \cap L$ leer ist.

Es bietet sich an, zu einer Sprache L über Σ die Sprache L' aller Wörter über dem von Σ disjunkten Alphabet $\Sigma' = \{a', b'\}$ zu betrachten, die man durch Ersetzen von a durch a' und b durch b' erhält.

(c) Dieses Resultat legt nahe, dass das Shuffle-Produkt zweier kontextfreier Sprachen nicht selbst kontextfrei sein muss. Hier soll ein rigoroser Beweis erbracht werden:
Seien $L_1 = \{a^n b^n | n \geq 1\}$ und $L_2 = \{c^n d^n | n \geq 1\}$. Zeigen Sie mithilfe des Pumpinglemmas für kontextfreie Sprachen, dass $L_1 \star L_2$ nicht kontextfrei ist.