Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Herbst 2019

46115

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

Einzelprüfung: **Theoretische Informatik/Algorithmen/Datenstrukturen**

Anzahl der gestellten Themen (Aufgaben): 2

Anzahl der Druckseiten dieser Vorlage: 12

Bitte wenden!

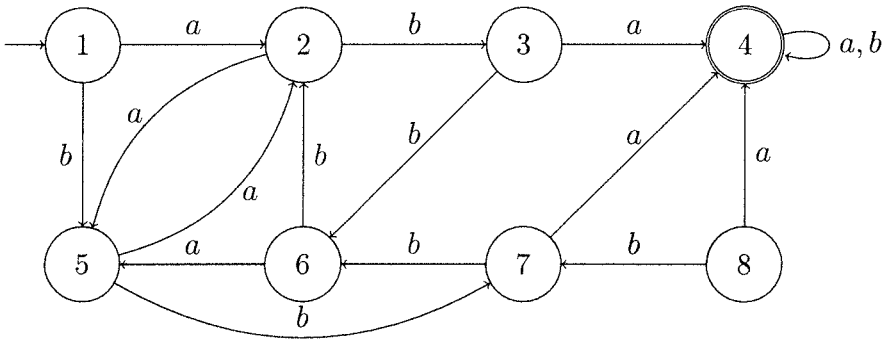
Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1 (Reguläre Sprachen) [20 PUNKTE]

(a) [4 Punkte] Geben Sie einen regulären Ausdruck für die Sprache über dem Alphabet $\{a, b\}$ an, die genau alle Wörter enthält, die eine gerade Anzahl a's haben.

(b) [16 Punkte] Sei A der folgende DEA über dem Alphabet $\{a, b\}$:



Führen Sie den Minimierungsalgorithmus für A durch und geben Sie den minimalen äquivalenten DEA für $L(A)$ als Zeichnung an.

Aufgabe 2 (Kontextfreie Sprachen) [40 PUNKTE]

(a) [24 Punkte] Betrachten Sie die folgenden Sprachen:

$$L_1 = \{a^n b^{2n} c^{2m} d^m \mid n, m \in \mathbb{N}\}$$
$$L_2 = \{a^n b^{2n} c^{2n} d^n \mid n \in \mathbb{N}\}$$

Zeigen Sie für L_1 und L_2, ob sie kontextfrei sind oder nicht. Für den Beweis von Kontext-Freiheit in dieser Frage reicht die Angabe eines Automaten oder einer Grammatik. (Beschreiben Sie dann die Konstruktionsidee des Automaten oder der Grammatik.) Für den Beweis von Nicht-Kontext-Freiheit verwenden Sie eine der üblichen Methoden.

(b) [16 Punkte] Eine kontextfreie Grammatik ist in Chomsky-Normalform, falls die folgenden Bedingungen erfüllt sind:

- alle Regeln sind von der Form $X \to YZ$ oder $X \to \sigma$ mit Nichtterminalzeichen X, Y, Z und Terminalzeichen σ,
- alle Nichtterminalzeichen sind erreichbar vom Startsymbol und
- alle Nichtterminalzeichen sind erzeugend, d. h. für jedes Nichtterminalzeichen X gibt es ein Wort w über dem Terminalalphabet, so dass $X \Rightarrow^* w$.

Fortsetzung nächste Seite!

Bringen Sie die folgende Grammatik in Chomsky-Normalform.

$S \rightarrow AAB \mid CD \mid abc$

$A \rightarrow AAAA \mid a$

$B \rightarrow BB \mid S$

$C \rightarrow CCC \mid CC$

$D \rightarrow d$

Das Startsymbol der Grammatik ist S, das Terminalalphabet ist $\{a, b, c, d\}$ und die Menge der Nichtterminalzeichen ist $\{S, A, B, C, D\}$.

Aufgabe 3 (Entscheidbarkeit und Komplexität) [30 PUNKTE]

Betrachten Sie die folgenden Entscheidungsprobleme:

SAT	
Gegeben:	Aussagenlogische Formel F in KNF.
Frage:	Gibt es eine erfüllende Belegung für F?

SAT'	
Gegeben:	Aussagenlogische Formel F in KNF.
Frage:	Gibt es eine Belegung, die in jeder Klausel mindestens ein Literal falsch macht?

Beachten Sie, dass die Belegung bei SAT' nicht erfüllend sein muss.

(a) [20 Punkte] Zeigen Sie, dass SAT in polynomieller Zeit reduzierbar auf SAT' ist.

(b) [10 Punkte] Zeigen Sie, dass SAT' NP-vollständig ist. Sie dürfen annehmen, dass SAT NP-vollständig ist.

Fortsetzung nächste Seite!

Aufgabe 4 (Sortierverfahren) [40 PUNKTE]

In der folgenden Aufgabe soll ein Feld A von ganzen Zahlen aufsteigend sortiert werden. Das Feld habe n Elemente A[1] bis A[n]. Der folgende Algorithmus sei gegeben:

```
1  var A : array[1..n] of integer;
2
3  procedure selection_sort
4  var i, j, smallest, tmp : integer;
5  begin
6    for j := 1 to n-1 do begin
7      smallest := j;
8      for i := j + 1 to n do begin
9        if A[i] < A[smallest] then
10         smallest := i;
11     end
12     tmp = A[j];
13     A[j] = A[smallest];
14     A[smallest] = tmp;
15   end
16 end
```

(a) [18 Punkte] Sortieren Sie das folgende Feld mittels des Algorithmus. Notieren Sie alle Werte, die die Variable smallest jeweils beim Durchlauf der inneren Schleife annimmt. Geben Sie die Belegung des Feldes nach jedem Durchlauf der äußeren Schleife in einer neuen Zeile an.

Index	1	2	3	4	5	6	7	8	9	10
Wert	27	32	3	6	17	44	42	29	8	14

(b) [4 Punkte] Der Wert der Variablen smallest wird bei jedem Durchlauf der äußeren Schleife mindestens ein Mal neu gesetzt. Wie muss das Feld A beschaffen sein, damit der Variablen smallest ansonsten niemals ein Wert zugewiesen wird? Begründen Sie Ihre Antwort.

(c) [4 Punkte] Welche Auswirkung auf die Sortierung oder auf die Zuweisungen an die Variable smallest hat es, wenn der Vergleich in Zeile 9 des Algorithmus statt A[i] $<$ A[smallest] lautet A[i] $\leq$ A[smallest]? Begründen Sie Ihre Antwort.

(d) [4 Punkte] Betrachten Sie den Algorithmus unter der Maßgabe, dass Zeile 9 wie folgt geändert wurde:

if A[i] $>$ A[smallest] **then**

Welches Ergebnis berechnet der Algorithmus nun?

(e) [10 Punkte] Betrachten Sie die folgende *rekursive* Variante des Algorithmus. Der erste Parameter ist wieder das zu sortierende Feld, der Parameter n ist die Größe des Feldes und der Parameter index ist eine ganze Zahl. Die Funktion min_index(A, x, y) berechnet für $1 \leq x \leq y \leq n$ den Index des kleinsten Elements aus der Menge $\{A[x], A[x+1], \ldots, A[y]\}$.

Fortsetzung nächste Seite!

```
1  procedure rek_selection_sort(A, n, index : integer)
2  var k, tmp : integer;
3  begin
4    if ⟨Abbruchbedingung⟩ then return;
5    k = min_index(A, index, n);
6    if k ≠ index then begin
7      tmp := A[k];
8      A[k] := A[index];
9      A[index] := tmp;
10   end
11   ⟨rekursiver Aufruf⟩
12 end
```

Der initiale Aufruf des Algorithmus lautet:

rek_selection_sort(A, n, 1)

Vervollständigen Sie die fehlenden Angaben in der Beschreibung des Algorithmus für

- die Abbruchbedingung in Zeile 4 und
- den rekursiven Aufruf in Zeile 11.

Begründen Sie Ihre Antworten.

Aufgabe 5 (O-Notation) [15 PUNKTE]

(a) [9 Punkte] Sortieren Sie die unten angegebenen Funktionen der O-Klassen $O(a(n))$, $O(b(n))$, $O(c(n))$, $O(d(n))$ und $O(e(n))$ bezüglich ihrer Teilmengenbeziehungen. Nutzen Sie ausschließlich die echte Teilmenge $\subset$ sowie die Gleichheit $=$ für die Beziehung zwischen den Mengen. Folgendes Beispiel illustriert diese Schreibweise für einige Funktionen f_1 bis f_5 (diese haben nichts mit den unten angegebenen Funktionen zu tun):

$$O(f_4(n)) \subset O(f_3(n)) = O(f_5(n)) \subset O(f_1(n)) = O(f_2(n))$$

Die angegebenen Beziehungen müssen weder bewiesen noch begründet werden.

- $a(n) = n^2 \cdot \log_2(n)$
- $b(n) = 2^n + n^4 + 42$
- $c(n) = 2^{n+4}$
- $d(n) = 3 \cdot \sqrt{n^5}$

(b) [6 Punkte] Bestimmen Sie eine asymptotische Lösung (in Θ-Schreibweise) für die folgende Rekursionsgleichung:

$T(n) = 4 \cdot T(\frac{n}{2}) + n^2$

Fortsetzung nächste Seite!

Aufgabe 6 (Stacks) [25 PUNKTE]

Gegeben sei die Implementierung eines Stacks ganzer Zahlen mit folgender Schnittstelle:

```
1 public class IntStack
2 {
3   // legt Element i auf den Stack
4   public void push(int i);
5   // gibt oberstes Element vom Stack
6   public int pop();
7   // fragt ab ob Stack leer ist
8   public boolean isEmpty();
9 }
```

Betrachten Sie nun die Realisierung der folgenden Datenstruktur Mystery, die zwei Stacks benutzt.

```
1  public class Mystery
2  {
3      private IntStack a = new IntStack();
4      private IntStack b = new IntStack();
5
6      public void foo(int item) {
7          a.push(item);
8      }
9
10     public int bar() {
11         if (b.isEmpty()) {
12             while (!a.isEmpty()) {
13                 b.push(a.pop());
14             }
15         }
16         return b.pop();
17     }
18 }
```

(a) [9 Punkte] Skizzieren Sie nach jedem Methodenaufruf der im folgenden angegebenen Befehlssequenz den Zustand der beiden Stacks eines Objekts m der Klasse Mystery. Geben Sie zudem bei jedem Aufruf der Methode bar an, welchen Wert diese zurückliefert.

```
1  Mystery m = new Mystery();
2  m.foo(3);
3  m.foo(5);
4  m.foo(4);
5  m.bar();
6  m.foo(7);
7  m.bar();
8  m.foo(2);
9  m.bar();
10 m.bar();
```

Fortsetzung nächste Seite!

(b) Sei n die Anzahl der in einem Objekt der Klasse Mystery gespeicherten Werte. Im folgenden wird gefragt, wieviele Aufrufe von Operationen der Klasse IntStack einzelne Aufrufe von Methoden der Klasse Mystery verursachen. Begründen Sie jeweils Ihre Antwort.

(i) [2 Punkte] Wie viele Aufrufe von Operationen der Klasse IntStack verursacht die Methode foo(x) im besten Fall?

(ii) [2 Punkte] Wie viele Aufrufe von Operationen der Klasse IntStack verursacht die Methode foo(x) im schlechtesten Fall?

(iii) [3 Punkte] Wie viele Aufrufe von Operationen der Klasse IntStack verursacht die Methode bar() im besten Fall?

(iv) [3 Punkte] Wie viele Aufrufe von Operationen der Klasse IntStack verursacht die Methode bar() im schlechtesten Fall?

(c) [6 Punkte] Welche allgemeinen Eigenschaften werden durch die Methoden foo und bar realisiert? Unter welchem Namen ist diese Datenstruktur allgemein bekannt?

Aufgabe 7 (Algorithmenentwurf) [10 PUNKTE]

Gegeben sei ein Array A von n ganzen Zahlen. In A sollen Duplikate eliminiert werden. Ausgabe soll ein neues Array A' sein, das jedes Element von A nur ein Mal enthält.

Entwerfen Sie einen Algorithmus mit Laufzeit $O(n \cdot \log_2 n)$, der diese Aufgabe löst.

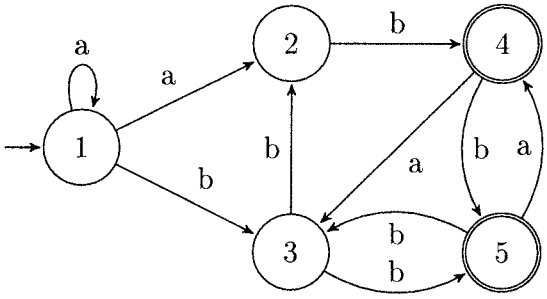
Sie dürfen die Funktionalität allgemein bekannter Algorithmen und Datenstrukturen (Listen, Graphen, Stacks, Heaps, Queues, Hash-Tabellen, Algorithmen zum Sortieren) unter Annahme der jeweils bekannten effizienten Implementierungen verwenden.

Thema Nr. 2
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1 (Komplemetieren eines NEA) [18 PUNKTE]

Es sei der nichtdeterministische endliche Automat $A = (\{a, b\}, \{1, 2, 3, 4, 5\}, \delta, 1, \{4, 5\})$ gegeben, wobei δ durch folgende Zeichnung beschrieben ist.



Konstruieren Sie nachvollziehbar einen deterministischen endlichen Automaten A', der das Komplement von $L(A)$ akzeptiert!

Aufgabe 2 (Rechtslineare Grammatik) [26 PUNKTE]

Gegeben ist die rechtslineare Grammatik $G = (\{a, b\}, \{S, A, B, C, D\}, S, R)$ mit $R = \{S \to aA, A \to bB, A \to aD, B \to aC, B \to bB, C \to bD, C \to b, D \to aA\}$. Sei L die von G erzeugte Sprache.

(a) [8 Punkte] Zeichnen Sie einen nichtdeterministischen endlichen Automaten, der L akzeptiert!

(b) [18 Punkte] Konstruieren Sie auf nachvollziehbare Weise einen regulären Ausdruck α mit $L(\alpha) = L$!

Aufgabe 3 (Regularität) [16 PUNKTE]

Bestimmen Sie, ob die Sprache $A = \{w \in \{a, b\}^* \mid |w|_a \geq 2|w|_b\}$ regulär ist und beweisen Sie Ihre Behauptung! Dabei bezeichnet $|w|_a$ die Anzahl der a's in w und $|w|_b$ die Anzahl der b's in w.

Aufgabe 4 (Entscheidbarkeit, Reduzierbarkeit, Aufzählbarkeit) [30 PUNKTE]

Sei $M_0, M_1, \ldots$ eine Gödelisierung aller Registermaschinen (RAMs).

(a) [12 Punkte] Zeigen Sie die Unentscheidbarkeit der Menge

$A = \{i \in \mathbb{N} \mid$ die von M_i berechnete Funktion $f : \mathbb{N} \to \mathbb{N}$ hat die Eigenschaft $W_f \cap \mathbb{P} = \emptyset\}$,

wobei W_f der Wertebereich von f und $\mathbb{P} = \{2, 3, 5, 7, \ldots\}$ die Menge der Primzahlen ist!

Fortsetzung nächste Seite!

(b) [18 Punkte] Im Folgenden bezeichne $M_i(i)$ die Berechnung der Maschine M_i bei Eingabe i. Sei B die Menge aller Paare (j,k), sodass $M_j(j)$ mindestens so lange wie $M_k(k)$ läuft, d. h.

$$B = \{(j,k) \in \mathbb{N}\times\mathbb{N} \mid \text{für alle natürlichen Zahlen } t \text{ gilt:} \\ M_j(j) \text{ hält innerhalb von } t \text{ Takten} \Rightarrow M_k(k) \text{ hält innerhalb von } t \text{ Takten}\}.$$

Weiter sei $\overline{K_0}$ das Komplement des speziellen Halteproblems $K_0 = \{i \in \mathbb{N} \mid M_i(i) \text{ hält}\}$. Es ist bekannt, dass $\overline{K_0}$ nicht rekursiv aufzählbar ist. Beweisen Sie, dass $\overline{K_0}$ Polynomialzeit-reduzierbar auf B ist! Folgern Sie daraus, dass B nicht rekursiv aufzählbar ist.

Aufgabe 5 (Quicksort) [21 PUNKTE]

Gegeben ist das Array $a = [13, 32, 9, 14, 42, 3, 10, 21]$.

(a) [14 Punkte] Sortieren Sie a mit Quicksort aufsteigend und in-situ von links nach rechts. Geben Sie die (Teil-)Arrays am Anfang jedes rekursiven Aufrufs und nach jedem Tauschen von Elementen (auch mit sich selber) an. Verwenden Sie als Pivotelement jeweils das rechteste Element im Teilarray. Kennzeichnen Sie immer das aktuelle Pivotelement. Geben Sie zum Schluss das Ergebnis an.

(b) [7 Punkte] Welche asymptotische best-case Laufzeit hat Quicksort? Geben Sie ein Array der Länge mindestens 7 an, mit welchem die Quicksort-Variante aus (a) nur die minimale Anzahl an Vergleichen benötigt (ohne Begründung).

Aufgabe 6 (Mastertheorem) [10 PUNKTE]

Sei $T : \mathbb{N} \to \mathbb{R}_0^+$ eine nicht-negative rekursive Funktion mit folgender Definition:

$$T(n) = \begin{cases} 1, & \text{falls } n \leq n_0 \\ a \cdot T\left(\frac{n}{b}\right) + f(n), & \text{sonst} \end{cases} \quad \text{mit } a \geq 1, b > 1, f : \mathbb{N} \to \mathbb{R}_0^+, n_0 \in \mathbb{N}$$

Laut Mastertheorem gilt:

Fall 1: $\exists \varepsilon > 0 : f(n) \in \mathcal{O}(n^{\log_b a - \varepsilon}) \Rightarrow T \in \Theta(n^{\log_b a})$

Fall 2: $f(n) \in \Theta(n^{\log_b a}) \Rightarrow T \in \Theta(n^{\log_b a} \cdot \log_b n)$

Fall 3: $\left(\exists \varepsilon > 0 : f(n) \in \Omega(n^{\log_b a + \varepsilon})\right) \wedge \left(\exists c : 0 < c < 1 \wedge a \cdot f\left(\frac{n}{b}\right) \leq c \cdot f(n)\right) \Rightarrow T \in \Theta(f(n))$.

Fortsetzung nächste Seite!

Gegeben sei die folgende Variante von MergeSort:

```
3MergeSort(int[] A, int l = 1, int r = A.length)
  if l < r then
    m1 = l + ⌊(1/3) · (r − l)⌋
    m2 = l + ⌊(2/3) · (r − l)⌋
    3MergeSort (A, l, m1)
    3MergeSort (A, m1 + 1, m2)
    3MergeSort (A, m2 + 1, r)
    Merge (A, l, m1, m2)
    Merge (A, l, m2, r)
  end
```

(a) [6 Punkte] Geben Sie zu obiger Funktion T die Werte der Konstanten **a** und **b**, sowie eine passende Definition der Funktion **f** an, sodass $T(n)$ der Anzahl von Rechenschritten des gegebenen Algorithmus entspricht.

(b) [4 Punkte] Bestimmen Sie die asymptotische Laufzeit des Algorithmus mithilfe des Mastertheorems. Geben Sie dabei den verwendeten Fall an.

Aufgabe 7 (Heapify) [18 PUNKTE]

Schreiben Sie in Pseudocode eine Methode heapify(int[] a), welche im übergebenen Array der Länge n die Heapeigenschaft in $\mathcal{O}(n)$ Schritten herstellt. D. h. als Ergebnis soll in a gelten, dass $a[i] \leq a[2i+1]$ und $a[i] \leq a[2i+2]$.

Fortsetzung nächste Seite!

Aufgabe 8 (Maximaler Spannbaum mit Jarník/Prim) [25 PUNKTE]

(a) [16 Punkte] Durch folgende Adjazenzmatrix sei ein ungerichteter Graph G mit Kantenlängen gegeben.

	a	b	c	d	e	f	g	h	i
a	–	–	7	–	9	2	–	4	–
b	–	–	–	–	–	3	–	5	2
c	7	–	–	4	4	–	–	–	–
d	–	–	4	–	–	–	2	-3	–
e	9	–	4	–	–	–	0	–	–
f	2	3	–	–	–	–	8	10	2
g	–	–	–	2	0	8	–	–	–
h	4	5	–	-3	–	10	–	–	–
i	–	2	–	–	–	2	–	–	20

Wenden Sie den Algorithmus von Jarník/Prim auf G ausgehend von Knoten d an, um einen Spannbaum T mit **maximalem** Gewicht zu berechnen. Erstellen Sie dazu eine Tabelle mit zwei Spalten und stellen Sie jeden einzelnen Schritt des Verfahrens in einer eigenen Zeile dar. Geben Sie in der ersten Spalte denjenigen Knoten v an, der vom Algorithmus als nächstes in T aufgenommen wird (dieser sog. "schwarze" Knoten ist damit fertiggestellt). Führen Sie in der zweiten Spalte alle anderen vom aktuellen Baum T direkt erreichbaren Knoten v (sog. "graue Randknoten") auf.

Geben Sie in der Tabelle Knoten stets als Tripel $(v, \delta, v.\pi)$ an, mit v als Knotenname, $v.\pi$ als aktueller Vorgängerknoten (anderer Knoten der Kante) und δ als Länge der Kante $\{v, v.\pi\}$.

(b) [9 Punkte] Sei $G = (V, E, w)$ ein Graph mit Kantenlängen $w : E \to \mathbb{N}$ und T ein Spannbaum von G mit maximalem Gewicht. Beweisen oder widerlegen Sie die folgende Aussage:

Längste (einfache) Wege zwischen zwei Knoten $u, v \in V$ enthalten nur Kanten aus T.

Fortsetzung nächste Seite!

Aufgabe 9 (Hashing) [16 PUNKTE]

(a) [12 Punkte] Verwenden Sie die Hash-Funktion $h(k,i) = (\,h'(k)+2i+i^2\,) \bmod 11$ mit $h'(k) = k \bmod 13$, um die Schlüssel 20, 12 und 30 in die folgende Hashtabelle einzufügen.

Dokumentieren Sie dabei jeweils, auf welche Zellen zugegriffen wird.

0	1	2	3	4	5	6	7	8	9	10
	1			17				8		

(a) [4 Punkte] Beschreiben Sie, welchen Vorteil quadratisches Sondieren bei der Kollisionsauflösung gegenüber linearem Sondieren bietet.