Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Herbst 2018

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: 12

Bitte wenden!

Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1 (Aussagen) [24 PUNKTE]

Zeigen oder widerlegen Sie die folgenden Aussagen (die jeweiligen Beweise sind sehr kurz).

Schreiben Sie zunächst zur Aussage „Richtig" oder „Falsch" und dann Ihre Begründung.

(a) [6 PUNKTE] Seien $L, L' \subseteq \Sigma^*$ formale Sprachen. Falls L regulär und L' semi-entscheidbar (= rekursiv aufzählbar) ist, dann ist der Durchschnitt $L \cap L'$ regulär.

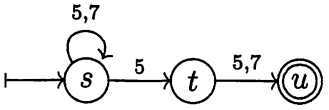
(b) [6 PUNKTE] Die Menge aller semi-entscheidbaren (= rekursiv-aufzählbaren) Sprachen ist überabzählbar unendlich.

(c) [6 PUNKTE] Angenommen es gilt $\mathcal{P} \neq \mathcal{NP}$. Dann ist jede formale Sprache $L \in \mathcal{NP} \setminus \mathcal{P}$ sicher nicht regulär.

(d) [6 PUNKTE] Das Halteproblem liegt in der Klasse $\mathcal{NP}$.

Aufgabe 2 (Reguläre Sprachen) [35 PUNKTE]

(a) [10 PUNKTE] Es sei $L \subseteq \{5,7\}^*$ die von dem folgenden nichtdeterministischen Automaten akzeptierte Sprache. Geben Sie einen Automaten für das Komplement $\{5,7\}^* \setminus L$ der Sprache L an.



(b) [15 PUNKTE] Minimieren Sie den deterministischen Automaten

$$A = (\{q, r, s, t, u, v\}, \{0, 1\}, \delta, q, \{q, s, u\})$$

mit der folgenden tabellarisch gegebenen Zustandsüberführungsfunktion:

δ	0	1
q	s	v
r	q	s
s	q	v
t	r	t
u	r	q
v	r	t

Machen Sie dabei Ihren Rechenweg deutlich.

Fortsetzung nächste Seite!

(c) [10 PUNKTE] Beweisen Sie, dass die folgende formale Sprache über $\Sigma = \{a, b\}$ nicht regulär ist:

$$L = \{b^n a^m \mid 2n \neq m\}$$

Aufgabe 3 (Kontextfreie Sprachen) [25 PUNKTE]

(a) [10 PUNKTE] Entwerfen Sie eine kontextfreie Grammatik für die folgende kontextfreie Sprache über dem Alphabet $\Sigma = \{a, b, c\}$:

$$L = \{wb^{3k}c^{2k+1}v \mid k \in \mathbb{N}, |w|_c = |v|_a\}.$$

(Hierbei bezeichnet $|u|_x$ die Anzahl des Zeichens x in dem Wort u, und es gilt $0 \in \mathbb{N}$.) Erklären Sie den Zweck der einzelnen Nichtterminale (Variablen) und der Grammatikregeln Ihrer Grammatik.

(b) [10 PUNKTE] Betrachten Sie die folgende kontextfreie Grammatik

$$G = (\{S, X, Y, Z\}, \{x, y\}, P, S)$$

mit den Produktionen

$$\begin{aligned} P: \quad S &\to ZX \mid y \\ X &\to ZS \mid SS \mid x \\ Y &\to SX \mid YZ \\ Z &\to XX \mid XS \end{aligned}$$

Benutzen Sie den Algorithmus von Cocke-Younger-Kasami (CYK) um zu zeigen, dass das Wort $xxxyx$ zu der von G erzeugten Sprache $L(G)$ gehört.

(c) [5 PUNKTE] Geben Sie eine Ableitung des Wortes $xxxyx$ mit G an.

Aufgabe 4 (Entscheidbarkeit) [36 PUNKTE]

(a) [8 PUNKTE] Benutzen Sie den Satz von Rice um zu beweisen, dass das folgende Problem nicht entscheidbar ist:

Eingabe: eine (geeignet codierte) Turingmaschine M, die eine (möglicherweise partielle) Funktion $f_M : \mathbb{N} \to \mathbb{N}$ berechnet;

Aufgabe: entscheiden, ob es ein $n \in \mathbb{N}$ mit $f_M(n) = 42$ gibt.

(b) [8 PUNKTE] Zeigen Sie, dass das Problem aus (a) semi-entscheidbar (= rekursiv-aufzählbar) ist.

(c) [20 PUNKTE] Zeigen Sie mit Hilfe einer Reduktion, dass das folgende Problem nicht entscheidbar ist:

Fortsetzung nächste Seite!

Eingabe: zwei (geeignet codierte) Turingmaschinen M_1 und M_2;

Aufgabe: entscheiden, ob für jedes Eingabewort w M_1 gestartet auf w genau dann hält, wenn M_2 gestartet auf w hält.

(Für die Reduktion kann benutzt werden, dass das Problem zu entscheiden, ob eine gegebene Turingmaschine auf jedes Eingabewort hält, unentscheidbar ist.)

Aufgabe 5 (AVL Bäume) [54 PUNKTE]

Hinweis: Wir betrachten in dieser Aufgabe binäre Suchbäume, bei denen jeder innere Knoten genau zwei Kinder hat. Schlüssel werden nur in den inneren Knoten gespeichert - die Blätter speichern keinerlei Informationen.

5.1 [2 PUNKTE] Welche Eigenschaften muss ein binärer Suchbaum haben, damit er ein AVL-Baum ist?

5.2 [8 PUNKTE] Mit $n(h)$ bezeichnen wir die *minimale* Anzahl *innerer* Knoten eines AVL-Baums der Höhe h.

(a) [2 PUNKTE] Begründen Sie, dass $n(1) = 1$ und $n(2) = 2$.

(b) [3 PUNKTE] Begründen Sie, dass $n(h) = 1 + n(h-1) + n(h-2)$.

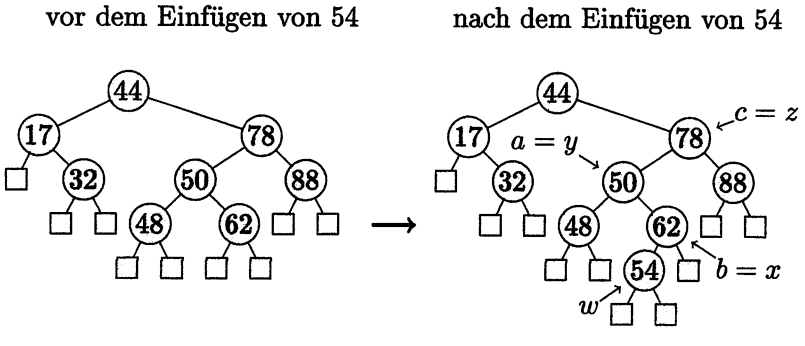
(c) [3 PUNKTE] Folgern Sie, dass $n(h) > 2^{\frac{h}{2}-1}$.

5.3 [2 PUNKTE] Warum ist die Höhe *jedes* AVL-Baums mit n inneren Knoten $O(\log n)$?

5.4 [16 PUNKTE] Fügen Sie die Elemente $(45, 16, 79, 31, 51, 87, 49, 61)$ in der angegebenen Reihenfolge in einen anfangs leeren binären Suchbaum ein (*ohne Rebalancierungen*). Zeichnen Sie den resultierenden Suchbaum nach jeder Einfügeoperation.

5.5 [2 PUNKTE] Ist der resultierende Suchbaum aus Teilaufgabe 5.4 ein AVL-Baum? Begründen Sie Ihre Antwort.

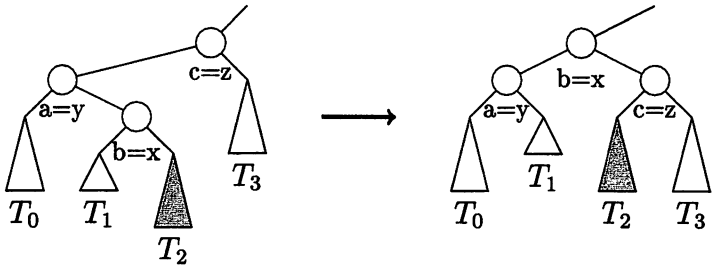
5.6 [12 PUNKTE] Das Einfügen in einen AVL-Baum funktioniert (zunächst) wie beim binären Suchbaum durch Erweitern eines äußeren Knotens w:



Fortsetzung nächste Seite!

Anschließend wird die AVL-Baum Eingenschaft (falls notwending) durch eine (Doppel-)Rotation wiederhergestellt: Wenn z der erste Knoten auf dem Pfad P von w zur Wurzel ist, der nicht balanciert ist, y das Kind von z auf P und x das Kind von y auf P, und wenn (a, b, c) die Inorder-Reihenfolge von x, y, z ist, dann führen wir die Rotation aus, die benötigt wird, um b zum obersten Knoten der drei zu machen.

Die folgende Illustration zeigt den Fall, dass $key(y) < key(x) < key(z)$, d.h. $(a, b, c) = (y, x, z)$, wobei w ein Knoten in T_2 ist.



Sei h die Höhe des Teilbaums T_3. Für $i = 0, 1, 2$ sei h_i die Höhe des Teilbaums T_i und für $v = x, y, z$ sei h_v die Höhe des Teilbaums mit der Wurzel v *vor* der Restrukturierung. Begründen Sie, dass

(a) [2 PUNKTE] $h_0 = h$

(b) [2 PUNKTE] $h_1 = h - 1$

(c) [2 PUNKTE] $h_2 = h$

(d) [2 PUNKTE] $h_x = h + 1$

(e) [2 PUNKTE] $h_y = h + 2$

(f) [2 PUNKTE] $h_z = h + 3$

5.7 [3 PUNKTE] Welche Höhe haben die Teilbäume mit den Wurzeln x, y, z *nach* der Restrukturierung? Begründen Sie Ihre Antworten.

5.8 [4 PUNKTE] Begründen Sie, dass die oben gezeigte Doppelrotation die AVL-Baum-Eigenschaft wiederherstellt.

5.9 [3 PUNKTE] Beschreiben Sie, wie ein binärer Baum der Höhe h in einem Array repräsentiert werden kann. Wie viel Speicherplatz ist für so eine Darstellung erforderlich?

5.10 [2 PUNKTE] Warum verwendet man bei der Implementierung von AVL-Bäumen eine verzeigerte Struktur und nicht eine Array-basierte Repräsentation?

Fortsetzung nächste Seite!

Aufgabe 6 (Ein Spiel mit Zahlen) [51 PUNKTE]

Wir spielen folgendes Spiel: Gegeben sei eine Folge $S = (x_1, \ldots, x_n)$ von n Zahlen. In einer *Runde* dürfen wir in S zwei beliebige *benachbarte* Zahlen x_k, x_{k+1} entfernen und durch die Zahl $2 \cdot (x_k + x_{k+1})$ ersetzen, ohne die Reihenfolge der anderen Zahlen in S zu verändern. Wir erhalten so eine neue Folge $S' = (x'_1, \ldots, x'_{n-1})$ von $n-1$ Zahlen, wobei $x'_i = x_i$ für $1 \leq i < k$, $x'_k = 2 \cdot (x_k + x_{k+1})$, und $x'_i = x_{i+1}$ für $k < i < n$. In der nächsten Runde spielen wir mit S' weiter. Nach $n-1$ Runden erhalten wir eine einelementige Folge $S^* = (x^*_1)$. Die Zahl x^*_1 ist unser *Gewinn.*

Ein mögliches Spiel auf der Folge $(2,1,3,4,1,2,3)$ sieht so aus:

$$\underline{(2,1},3,4,1,2,3) \to (6,3,4,\underline{1,2},3) \to (6,\underline{3,4},6,3) \to$$
$$(6,14,\underline{6,3}) \to (\underline{6,14},18) \to (\underline{40,18}) \to (116).$$

Ein Spiel können wir auch als eine *Klammerung* der Elemente von S interpretieren: Anstatt in einer Runde x_k, x_{k+1} zu entfernen und durch $2 \cdot (x_k + x_{k+1})$ zu ersetzen, ersetzen wir die beiden Zahlen durch den geklammerten Ausdruck $(x_k * x_{k+1})$. In unserem Beispiel sieht das so aus:

$$\underline{(2,1},3,4,1,2,3) \to ((2*1),3,4,\underline{1,2},3) \to ((2*1),\underline{3,4},(1*2),3) \to$$
$$((2*1),(3*4),\underline{(1*2)*3}) \to \underline{((2*1),(3*4)},((1*2)*3)) \to$$
$$\underline{(((2*1)*(3*4)},((1*2)*3)) \to$$
$$(((\underline{(2*1)*(3*4)})*((1*2)*3))).$$

Nach $n-1$ Runden entsteht ein Klammerausdruck K auf den Elementen von S. Wenn wir den Operator $*$ definieren als $x * y := 2 \cdot (x+y)$, dann ist der Wert $w(K)$ dieses letzten Klammerausdrucks gerade der Gewinn des Spiels: $w((((2*1)*(3*4))*((1*2)*3))) = 116$.

Einen Klammerausdruck der den *größten* Gewinn erzielt nennen wir *optimal.* Für $1 \leq i \leq j \leq n$ betrachten wir die Teilfolge $S[i,j] := (x_i, \ldots, x_j)$ von S. Mit $W(i,j)$ bezeichnen wir den Wert eines optimalen Klammerausdrucks für $S[i,j]$.

6.1 [5 PUNKTE] Angenommen K ist ein *optimaler* Klammerausdruck für $S[i,j]$, dessen *letzte* $*$-Operation zwischen x_l und x_{l+1} liegt, d.h. $K = L * R$, wobei L ein Klammerausdruck für die Teilfolge $S[i,l]$ und R ein Klammerausdruck für die Teilfolge $S[l+1,j]$ ist.

(a) [1 PUNKT] Warum ist $w(K) = 2 \cdot (w(L) + w(R))$?

(b) [4 PUNKTE] Begründen Sie, dass L und R *optimale* Klammerausdrücke für $S[i,l]$ bzw. $S[l+1,j]$ sind.

6.2 [7 PUNKTE] Begründen Sie, dass

(a) [1 PUNKT] $W(i,i) = x_i$ für $1 \leq i \leq n$

(b) [6 Punkte] und dass für $1 \leq i < j \leq n$ folgende Rekursionsgleichung gilt:

$$W(i,j) = \max_{i \leq l < j} \{2 \cdot (W(i,l) + W(l+1,j))\}$$

Fortsetzung nächste Seite!

6.3 [6 PUNKTE] Formulieren Sie auf Basis obiger Rekursionsgleichung einen *rekursiven* Algorithmus WINGAMEREC(i,j) zur Berechnung von $W(i,j)$ in Pseudocode. Die Laufzeit $T(n)$ Ihres Algorithmus beim Aufruf WINGAMEREC$(1,n)$ sollte folgender asymptotischer Rekursionsungleichung genügen:

$$T(n) = \sum_{1 \le l < n} (T(l) + T(n-l) + \Theta(1)), \text{ für } n > 1 \quad (\text{mit } T(1) = \Theta(1))$$

Begründen Sie, dass dem so ist.

6.4 [6 PUNKTE] Finden Sie eine (möglichst kurze) Eingabefolge S an der deutlich wird, dass sich die Teilprobleme der Rekursion zur Berechnung von $W(1,n)$ überlappen. Illustrieren Sie das Phänomen, indem Sie den Rekursionsbaum zeichnen und die mehrfach berechneten Teilprobleme markieren.

6.5 [3 PUNKTE] Begründen Sie, dass $T(n) = 2T(n-1) + \Omega(1)$ für $n > 1$.

6.6 [4 PUNKTE] Begründen Sie, dass $T(n) = \Omega(2^n)$.

6.7 [4 PUNKTE] Wie sieht eine Reihenfolge aus, in der die $W(i,j)$ bottom-up berechnet werden können, die mit obiger Rekursionsgleichung kompatibel ist? Begründen Sie Ihre Antwort.

6.8 [6 PUNKTE] Geben Sie ein *dynamisches Programm* WINGAMEDP(S) in Pseudocode an, das den Wert $W(1,n)$ in *polynomieller Zeit* berechnet.

Hinweis: Wenn Sie die vorige Aufgabe nicht lösen konnten, verwenden Sie die lexikographische Reihenfolge zur bottom-up Berechnung der $W(i,j)$.

6.9 [6 PUNKTE] Begründen Sie, warum Ihr Algorithmus in Teilaufgabe 6.8 korrekt ist.

6.10 [4 PUNKTE] Bestimmen Sie die asymptotische Laufzeit und den Speicherplatzbedarf Ihres Algorithmus aus Teilaufgabe 6.8 möglichst genau.

Thema Nr. 2
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Aufgabe 1 (Reguläre Sprachen) [22 PUNKTE]

(a) [6 PUNKTE] Geben Sie alle Nerode-Äquivalenzklassen von $L((aa)^*)$ über dem Alphabet $\{a, b\}$ an. Geben Sie für jede Klasse 3 Wörter an, die zu dieser gehören.

(b) [6 PUNKTE] Geben Sie einen DEA mit maximal 4 Zuständen an, zu dem es keinen äquivalenten DEA mit höchstens einem akzeptierenden Zustand gibt. Verwenden Sie ein Alphabet mit höchstens zwei Symbolen. Warum erfüllt Ihr DEA diese Eigenschaft?

(c) [10 PUNKTE] Ist $L_1 = \{a^i b^j c^\ell \mid i + j = \ell\} \cup L((a + c)^*) \cup L((b + c)^*)$ eine reguläre Sprache? Geben Sie einen Beweis für Ihre Antwort. Um Regularität zu zeigen reicht die Angabe eines Automaten oder regulären Ausdrucks, der die Sprache akzeptiert.

Aufgabe 2 (Kontextfreie Sprachen) [22 PUNKTE]

(a) [6 PUNKTE] Entscheiden Sie mit Hilfe des CYK-Algorithmus, ob das Wort *acbaabaa* in der Sprache der nachfolgenden Grammatik enthalten ist.

$$S \to c \mid BA$$

$$A \to a \mid b \mid CB$$

$$B \to AA \mid AS$$

$$C \to SA$$

(b) [4 PUNKTE] Seien L_1 und L_2 kontextfreie Sprachen. Ist $L_1 \cup L_2$ eine entscheidbare Sprache? Ist $L_1 \cap L_2$ eine entscheidbare Sprache? Begründen Sie Ihre Antworten.

(c) [6 PUNKTE] Sei L_3 die Komplementsprache von $L = \{a^n b^n c^n \mid n \in \mathbb{N}\}$ über Alphabet $\{a, b, c\}$, d.h., $L_3 = \{w \in \{a, b, c\}^* \mid w \notin L\}$. Ist L_3 kontextfrei? Geben Sie einen Beweis für Ihre Antwort. Um Kontextfreiheit zu zeigen reicht die Angabe eines Automaten oder einer Grammatik, die die Sprache akzeptiert. Bitte erläutern Sie dann Ihre Konstruktionsidee. Sie müssen nicht formal beweisen, dass Ihr Automat oder Ihre Grammatik korrekt ist.

(d) [6 PUNKTE] Sei L_4 die Sprache $\{a^{n^2} \mid n \in \mathbb{N}\}$ über Alphabet $\{a\}$. Ist L_4 kontextfrei? Geben Sie einen Beweis für Ihre Antwort. Um Kontextfreiheit zu zeigen reicht die Angabe eines Automaten oder einer Grammatik, die die Sprache akzeptiert. Bitte erläutern Sie dann Ihre Konstruktionsidee. Sie müssen nicht formal beweisen, dass Ihr Automat oder Ihre Grammatik korrekt ist.

Fortsetzung nächste Seite!

Aufgabe 3 (Entscheidbarkeit) [16 PUNKTE]

Wir bezeichnen mit M_w die Turingmaschine, die von einem Wort w kodiert wird.

(a) [8 PUNKTE] Ist die Sprache $\{w \mid M_w$ hält auf jeder Eingabe nach maximal 42 Schritten an$\}$ entscheidbar?

(b) [8 PUNKTE] Ist die Sprache $\{w \mid M_w$ hält auf Eingabe ε nach einer geraden Anzahl von Schritten an$\}$ entscheidbar?

Beweisen Sie Ihre Antwort.

Aufgabe 4 (Komplexität) [10 PUNKTE]

Wir betrachten ungerichtete Graphen $G = (V, E)$, wo E eine Teilmenge E_e hat, die wir *exklusive Kanten* nennen. Eine *beschränkte Überdeckung* von G ist eine Teilmenge U von V, so dass

- jeder Knoten einen Nachbarknoten in U hat oder selbst in U liegt (für jeden Knoten $u \in V \setminus U$ gibt es einen Knoten $v \in U$ mit $(u, v) \in E$) und
- für jede exklusive Kante $(u, v) \in E_e$ genau einer der Knoten u, v in U liegt.

Betrachten Sie nun die folgenden Entscheidungsprobleme:

3SAT	
Gegeben:	Aussagenlogische Formel φ in 3KNF
Gefragt:	Hat φ eine erfüllende Belegung?

BÜ	
Gegeben:	Graph $G = (V, E)$, exklusive Kantenmenge $E_e \subseteq E$ und $k \in \mathbb{N}$
Gefragt:	Hat G eine *beschränkte Überdeckung* U mit $\lvert U \rvert \leq k$?

Beweisen Sie, dass BÜ NP-vollständig ist. Sie dürfen dabei annehmen, dass 3SAT NP-vollständig ist.

Aufgabe 5 (Studienzeitoptimierung) [45 PUNKTE]

Für den Bachelorstudiengang Informatik an Ihrer Hochschule werden die Module als Knoten eines gerichteten Graphen $G = (V, E)$ modelliert. Es führt eine Kante (u, w) von Modul u zu Modul w genau dann, wenn der (erfolgreiche) Abschluss von Modul u notwendige Voraussetzung für die Teilnahme an Modul w ist.

Der Einfachheit halber machen wir zwei Annahmen:

Fortsetzung nächste Seite!

1. Jedes Modul dauert genau ein Semester.
2. In jedem Semester werden alle Module des Studiengangs angeboten.

Wenn k die Länge des *längsten Pfades* in G ist, dann ist die Mindeststudiendauer $k+1$ Semester.

Ihre Aufgabe ist es nun, den längsten Pfad in G zu berechnen und dafür einen möglichst effizienten Algorithmus zu entwickeln.

Zunächst folgen einige Tipps, dann drei Fragen als Vorüberlegung. Anschließend geht es zur Implementierung.

Tipps:

1. Der Graph ist zyklusfrei.
2. Der Graph ist kein gerichteter Wald, d.h. es kann Knoten geben, die mehrere Vorgänger haben.

(a) [5 PUNKTE] Von welchen Knoten aus sollte die Suche nach dem längsten Pfad beginnen? Begründen Sie Ihre Antwort.

(b) [5 PUNKTE] Beschreiben Sie, wie man *induktiv* für einen beliebigen Knoten v die Länge des längsten Teilpfades bis zu diesem Knoten aus der maximalen Pfadlänge zu allen Vorgängerknoten berechnen kann. Beschreiben Sie die Umstände der *Induktionsverankerung.*

(c) [5 PUNKTE] Welches ist eine sinnvolle Reihenfolge, in der die Knoten von G bei der Durchführung des Algorithmus abgearbeitet werden? Begründen Sie Ihre Antwort.

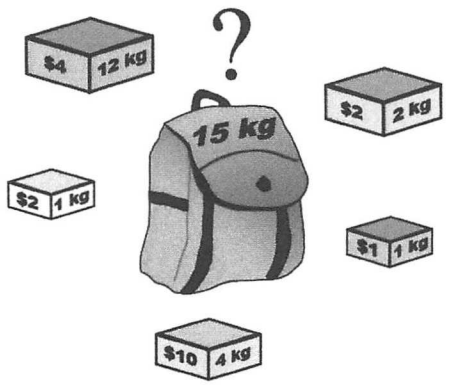
(d) [25 PUNKTE] Formulieren Sie einen Algorithmus, der Ihnen die minimale Studiendauer berechnet. Geben Sie zudem die asymptotische Laufzeit Ihres Algorithmus mittels O-Notation an.

(e) [5 PUNKTE] Wie ändert sich die Problemstellung, wenn es Module gibt, die nur im Sommer- bzw. nur im Wintersemester angeboten werden? Müssen Sie den Algorithmus dafür abändern? Begründen Sie Ihre Antwort. (Annahme: Das Studium kann sowohl zum Sommersemester als auch zum Wintersemester aufgenommen werden.)

Aufgabe 6 (Backtracking) [30 PUNKTE]

Ein sehr bekanntes Optimierungsproblem ist das sogenannte Rucksackproblem: Gegeben ist ein Rucksack mit der Tragfähigkeit B. Weiterhin ist eine endliche Menge von Gegenständen mit Werten und Gewichten gegeben. Nun soll eine Teilmenge der Gegenstände so ausgewählt werden, dass ihr Gesamtwert maximal ist, aber ihr Gesamtgewicht die Tragfähigkeit des Rucksacks nicht überschreitet.

Fortsetzung nächste Seite!



Mathematisch exakt kann das Rucksackproblem wie folgt formuliert werden:

Gegeben ist eine endliche Menge von Objekten U. Durch eine Gewichtsfunktion $w : U \to \mathbb{R}^+$ wird den Objekten ein Gewicht und durch eine Nutzenfunktion $v : U \to \mathbb{R}^+$ ein festgelegter Nutzwert zugeordnet.

Des Weiteren gibt es eine vorgegebene Gewichtsschranke $B \in \mathbb{R}^+$. Gesucht ist eine Teilmenge $K \subseteq U$, die die Bedingung $\sum_{u \in K} w(u) \leq B$ einhält und die Zielfunktion $\sum_{u \in K} v(u)$ maximiert.

Das Rucksackproblem ist NP-vollständig (Problemgröße ist die Anzahl der Objekte), sodass es an dieser Stelle wenig Sinn macht, über eine effiziente Lösung nachzudenken. Lösen Sie das Rucksackproblem daher mittels *Backtracking* und formulieren Sie einen entsprechenden Algorithmus. Gehen Sie davon aus, dass die Gewichtsschranke B sowie die Anzahl an Objekten N beliebig, aber fest vorgegeben sind.

Das Programm soll folgende Ausgaben liefern:

1. Maximaler Nutzwert, der durch eine Objektauswahl unter Einhaltung der Gewichtsschranke B erreicht werden kann.
2. Das durch die maximierende Objektmenge erreichte Gesamtgewicht.
3. Diejenigen Objekte (Objektnummern) aus U, die zur Maximierung des Nutzwerts beigetragen haben.

Aufgabe 7 (O-Nation) [20 PUNKTE]

(a) Beweisen Sie die folgenden Aussagen formal nach den Definitionen der O-Notation oder widerlegen Sie sie.

i. [5 PUNKTE] $O(n \cdot \log_2 n) \subseteq O(n \cdot (\log_2 n)^2)$

ii. [5 PUNKTE] $2^{n+1} \in O(2^n)$

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

[10 PUNKTE] $T(n) = 4 \cdot T(\frac{n}{2}) + n^2$

Fortsetzung nächste Seite!

Aufgabe 8 (Sortierverfahren) [25 PUNKTE]

Gegeben sei das folgende Feld A mit 7 Schlüsseln:

$$[15, 4, 10, 7, 1, 8, 10]$$

(a) [10 PUNKTE] Sortieren Sie das Feld mittels des Sortierverfahrens *Bubblesort*. Markieren Sie jeweils, welche zwei Feldwerte verglichen werden und geben Sie den Zustand des gesamten Feldes jeweils neu an, wenn Sie eine Vertauschung durchgeführt haben.

(b) [10 PUNKTE] Sortieren Sie das Feld mittels des Sortierverfahrens *Selectionsort*. Markieren Sie jeweils, welche zwei Feldwerte verglichen werden und geben Sie den Zustand des gesamten Feldes jeweils neu an, wenn Sie eine Vertauschung durchgeführt haben.

(c) [5 PUNKTE] Vergleichen Sie beide Sortierverfahren hinsichtlich ihres Laufzeitverhaltens im *best case*. Welches Verfahren ist in dieser Hinsicht besser, wenn das zu sortierende Feld anfangs bereits sortiert ist? Begründen Sie Ihre Antwort.