Korrektur fehlerhafte Angabe in der Prüfung

Einzelprüfungsnummer:	66115
Zeitverlängerung:	10 Minuten
Ort des Fehlers:	Seite 12 / Thema Nr. 2 / Teilaufgabe II / Aufgabe 2 c)

Die Ergänzung ist farblich gekennzeichnet.

$$L = \{\omega 1^n v \mid n \in \mathbb{N},\ \omega, v \in \{0,2\}^*, n = |\omega|_0 \cdot |v|_2\}$$

Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer
Kennzahl: _____	**Frühjahr 2023**	**66115**
Kennwort: _____		
Arbeitsplatz-Nr.: _____		

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach:	**Informatik (vertieft studiert)**
Einzelprüfung:	**Theoretische Informatik, Algorithmen**
Anzahl der gestellten Themen (Aufgaben):	2
Anzahl der Druckseiten dieser Vorlage:	13

Bitte wenden!

Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Teilaufgabe I: Algorithmen

Aufgabe 1 ($\mathcal{O}$-Notation und Funktionenwachstum) [30 PUNKTE]

a) Sind die folgenden Aussagen korrekt? Begründen Sie Ihre Antwort.

i) $\mathcal{O}(n^2) = \mathcal{O}(n \log n)$.

ii) $n^2 + o(n) = \omega(n \log n)$.

iii) $\Omega(n^3) = \Omega(n^2)$.

b) Ordnen Sie die unten angegebenen Funktionen $f_1, \ldots, f_8$ bzgl. der $\mathcal{O}$-Notation. Schreiben Sie dazu die Mengen $\mathcal{O}(f_i(n))$ als Kette von Relationen „$\subset$“ bei echten Teilmengenbeziehungen oder „$=$“ bei Gleichheit der Mengen. Beispiel für die Formatierung Ihrer Antwort für Beispielfunktionen $a(n), b(n), c(n), d(n)$:

$$\text{„}\mathcal{O}(b(n)) \subset \mathcal{O}(a(n)) = \mathcal{O}(d(n)) \subset \mathcal{O}(c(n))\text{“}$$

Geben Sie für jede Relation „$\subset$“ und „$=$“ eine kurze Begründung an.

- $f_1(n) = n(n+1)(n+3)$
- $f_2(n) = \log_2 n$
- $f_3(n) = 2^{n/2}$
- $f_4(n) = 2^n$
- $f_5(n) = \ln n$
- $f_6(n) = (\sqrt{n})^6$
- $f_7(n) = 4^{n/2}$
- $f_8(n) = 42$

c) Berechnen Sie eine geschlossene Formel in Θ-Notation für die folgende Rekursionsgleichung. Wir nehmen vereinfachend an, dass $n \geq 1$ eine Potenz von 4 ist.

$$T(n) = \begin{cases} 3 \cdot T(n/4) + 2n & \text{falls } n > 1 \\ 8 & \text{falls } n = 1 \end{cases}$$

Aufgabe 2 (HeapSort) [28 PUNKTE]

a) Im Kontext des Algorithmus HEAPSORT interpretieren wir ein 1-indiziertes (d. h. Index mit 1 beginnend) Feld/Array als Binärbaum, in dem die Wurzel an Position 1 steht und jeder Knoten an Position i mögliche Kinder an Position $2i$ (linkes Kind) und Position $2i+1$ (rechtes Kind) hat. Zeichnen Sie den Binärbaum, der aus dem folgenden Array resultiert.

$$A = [3, 9, 8, 5, 4, 6, 7, 1, 2]$$

b) Ein durch ein Feld/Array A gegebener Binärbaum ist ein Max-Heap, wenn für alle Knoten i außer der Wurzel gilt: $A[\text{Elter}(i)] \geq A[i]$.
Wenden Sie die Prozedur MAXHEAPIFY$(A, 1)$ (siehe Pseudocode) auf die Wurzel des Binärbaums aus Aufgabenteil a) an und zeichnen Sie den resultierenden Binärbaum. Ist der entstandene Baum ein Max-Heap? Begründen Sie Ihre Antwort.

Algorithmus 1: MAXHEAPIFY(A, i)

```
1 if Knoten i hat mindestens ein Kind then
2 |   Sei j der Index des Kindes von i,
3 |   das den größten Wert in A hat.
4 |   if A[j] > A[i] then
5 |   |   Vertausche die Werte von A[i] und A[j].
6 |   |   MaxHeapify(A, j)
7 |   end
8 end
```

c) Betrachten Sie nun einen beliebigen Binärbaum T. Sei i ein Knoten von T und $T(i)$ der Teilbaum von T mit Wurzel i. Beweisen Sie folgende Aussage per Induktion über die Höhe von $T(i)$.

Für jeden Knoten i gilt: falls alle Kinder von i Wurzeln von Max-Heaps sind, ist nach Ausführung von MAXHEAPIFY(A, i) der Teilbaum $T(i)$ ein Max-Heap.

d) Die Prozedur BUILDMAXHEAP(A) soll aus einem beliebigen Array A einen Max-Heap erzeugen. Der folgende Pseudocode enthält allerdings einen gravierenden Fehler.

Algorithmus 2: BUILDMAXHEAP(A)

```
1 for i = 1 to ⌊A.length/2⌋ do
2 |   MaxHeapify(A, i)
3 end
```

(i) Erklären Sie kurz, wo der Fehler liegt.

(ii) Geben Sie ein Array A an, auf dem BUILDMAXHEAP nicht korrekt arbeitet. Begründen Sie Ihre Antwort.

(iii) Erklären Sie, wie der obige Pseudocode mit möglichst kleinen Änderungen repariert werden kann.

Aufgabe 3 (Dynamische Programmierung) [30 PUNKTE]

Sie haben w Würfel, die jeweils eine Zahl aus $\{1, 2, 3, 4, 5, 6\}$ anzeigen. Gegeben sei eine natürliche Zahl $S \geq 1$. Gesucht ist die Anzahl an Möglichkeiten, wie die Würfel fallen können, so dass die Summe der Augen genau S ergibt. Die Würfel sind unterscheidbar.

Beispiel 1: $w = 1$, $S = 3$. Dann ist die Antwort „1“, da es nur eine Möglichkeit gibt, mit einem Würfel eine Summe von 3 zu erreichen.

Beispiel 2: $w = 2$, $S = 7$. Dann ist die Antwort „6“, da es 6 Möglichkeiten gibt, mit zwei unterscheidbaren Würfeln eine Summe von 7 zu erreichen: 1+6, 2+5, 3+4, 4+3, 5+2, 6+1

In dieser Aufgabe sollen Sie einen Algorithmus `int` ANZAHL(`int` w, `int` S) entwerfen, der die gesuchte Ausgabe für Eingaben $w \in \mathbb{N}$ und $S \in \mathbb{N}$ mithilfe von Dynamischer Programmierung berechnet. Ihr Algorithmus soll Lösungen für Teilprobleme in einer Tabelle T speichern.

a) Beschreiben Sie die Größe der Tabelle T für gegebene Werte von w und S und welche Bedeutung ein Eintrag $T[i, j]$ in der Tabelle hat.

b) Geben Sie eine Rekursionsgleichung an, mit der ANZAHL(w, S) für $w \geq 1$ und $S \geq 1$ rekursiv berechnet werden kann. Begründen Sie Ihre Antwort.

c) Geben Sie einen Algorithmus in Pseudocode an, der die Tabelle T mithilfe von Dynamischer Programmierung so befüllt, dass der Wert ANZAHL(w, S) berechnet wird.

Geben Sie zudem an, welcher Tabelleneintrag dem gesuchten Wert ANZAHL(w, S) entspricht.

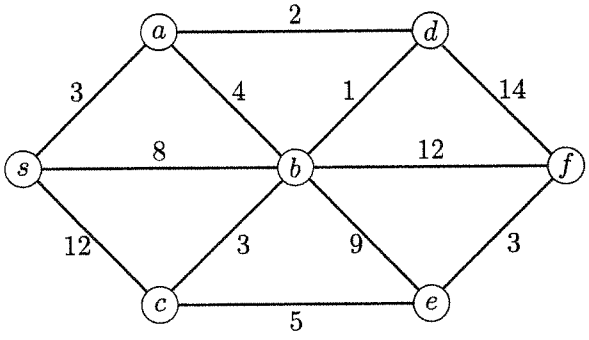
d) Geben Sie die Laufzeit Ihres Algorithmus in $\mathcal{O}$-Notation an. Begründen Sie Ihre Antwort kurz.

Aufgabe 4 (Algorithmus von Dijkstra) [32 PUNKTE]

Der Algorithmus von Dijkstra berechnet kürzeste Wege von einem Knoten s zu allen anderen Knoten eines Graphen, indem die Knoten des Graphen nach aufsteigender Entfernung von s zu einer Menge S hinzugefügt werden.

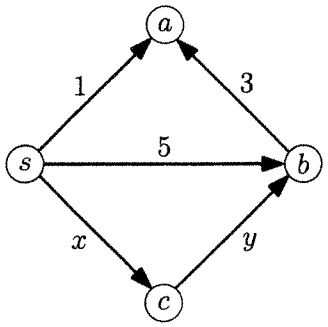
a) Führen Sie den Algorithmus von Dijkstra auf dem folgenden ungerichteten Graphen $G = (V, E)$ aus, startend am Knoten s. Geben Sie tabellarisch für alle Knoten $v \in V \setminus \{s\}$ an, welche Distanz von s nach v der Algorithmus von Dijkstra in jedem Schritt speichert. Geben Sie außerdem in jedem Schritt die Knotenmenge S an.

Geben Sie anschließend den kürzesten Weg von s nach f an.



b) Wie oft wird bei der Ausführung des Algorithmus von Dijkstra auf dem Graphen G die Distanz von s zu f verringert? Kann sich diese Anzahl erhöhen, wenn für die zu f inzidenten Kanten andere positive Kantengewichte gewählt werden? Begründen Sie Ihre Antwort.

c) Der Algorithmus von Dijkstra arbeitet korrekt, wenn alle Kantengewichte nichtnegativ sind. Betrachten Sie folgenden gerichteten Graphen, in dem zwei Kantengewichte x und y noch nicht festgelegt sind.



Wählen Sie x und y aus der Menge $\{-9, -8, \ldots, +8, +9\}$, so dass $x \notin \{1, 5\}$ (damit alle Kantengewichte an s verschieden sind) und der Algorithmus von Dijkstra ein falsches Ergebnis für den kürzesten Weg von s nach a liefert.

Erläutern Sie, welchen kürzesten Weg von s nach a der Algorithmus mit Ihrer Wahl von x und y berechnet und wie der tatsächliche kürzeste Weg aussieht. Erklären Sie kurz, *warum* der Algorithmus von Dijkstra hier fehlschlägt.

Teilaufgabe II: Theoretische Informatik

Aufgabe 1 (Reguläre Sprachen) [26 PUNKTE]

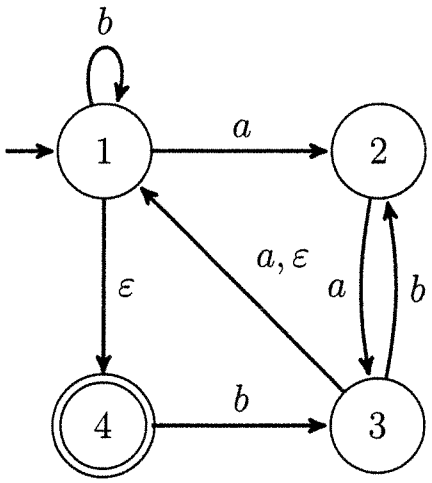
a) Geben Sie einen DEA für die Sprache

$$L_1 = \{w \in \{a, b, c\}^* \mid \text{ nach jedem } aa \text{ kommt direkt ein } b\}$$

an. So sind die Wörter a und $babab$ in der Sprache enthalten (da aa nicht vorkommt), sowie $baababaab$ (da direkt nach jedem Vorkommen von aa ein b kommt), aber nicht aa oder $baaab$.

b) Wieviele Äquivalenzklassen hat die Nerode-Relation bezüglich $\sim_{L_1}$? Geben Sie für jede Äquivalenzklasse drei Beispielwörter an.

c) Betrachten Sie den folgenden ε-NEA A:



Konstruieren Sie mit Hilfe der Potenzmengenkonstruktion den DEA A', der dieselbe Sprache wie A erkennt. Erklären Sie insbesondere, wie Sie mit ε-Kanten umgehen und benennen Sie Ihre Zustände sinnvoll.

Aufgabe 2 (Regularität und Kontextfreiheit) [36 PUNKTE]

a) Zeigen Sie, dass die Sprache

$$L_1 = \{a^m b^n c^n \mid n, m \geq 1; n, m \in \mathbb{N}\} \cup \{b^m c^n \mid n, m \geq 0; n, m \in \mathbb{N}\}$$

nicht regulär ist.

Hinweis: Sie können $L_1 \cap aa^*b^*c^*$ betrachten.

b) Ist die Sprache

$$L_2 = \{(abc)^n ab(cab)^n c(abc)^n \mid n \geq 0; n \in \mathbb{N}\}$$

regulär? Beweisen Sie Ihre Antwort.

c) Zeigen Sie, dass die Sprache

$$L_3 = \{a^n b^m c d^m e^n \mid m, n \in \mathbb{N}, n+m \text{ ist gerade}\}$$

kontextfrei ist.

d) Zeigen Sie, dass die Sprache

$$L_4 = \{w_1 c^n w_2 \mid n \geq 0, n \in \mathbb{N}, w_1, w_2 \in \{a,b\}^*, \#_a(w_1) > n \text{ und } \#_b(w_2) < n\}$$

nicht kontextfrei ist. Hierbei entspricht $\#_a(w)$ der Anzahl der Buchstaben a im Wort w und $\#_b(w)$ der Anzahl der Buchstaben b im Wort w. Z. B. ist $\#_a(abbbaab) = 3$ und $\#_b(abbbaab) = 4$.

Aufgabe 3 (Entscheidbarkeit) [26 PUNKTE]

Seien L, L_1 und L_2 Sprachen über dem Alphabet Σ. Sei $\overline{L}$ das Komplement von L, das heißt $\overline{L} = \{w \in \Sigma^* \mid w \notin L\}$. Sei P die Klasse der Entscheidungsprobleme, die deterministisch in Polynomialzeit lösbar sind.

a) Entscheiden Sie begründet, ob folgende Aussage korrekt ist: Falls L_1 regulär und L_2 kontextfrei ist, dann ist $L_1 - L_2$ in P.

b) Zeigen Sie: Ist die Sprache L semi-entscheidbar, dann sind die Sprachen $L \cap \overline{L}$ und $L \cup \overline{L}$ immer entscheidbar.

c) Zeigen Sie: Sind die Sprachen L und $\overline{L}$ semi-entscheidbar, dann ist auch L entscheidbar.

d) Entscheiden Sie begründet, ob folgende Aussage korrekt ist: Falls L in NP liegt, dann ist L entscheidbar.

Aufgabe 4 (Komplexität) [32 PUNKTE]

Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph mit Knotenmenge V und Kantenmenge E. Ein *Pfad* in G ist eine Folge $p = v_1 \cdots v_k$ von Knoten, so dass $(v_i, v_{i+1}) \in E$ ist (falls G gerichtet ist) oder $\{v_i, v_{i+1}\} \in E$ (falls G ungerichtet ist) für alle $i \in \{1, \ldots, k-1\}$. Wir sagen, dass p einen *Kreis* bildet, falls $v_1 = v_k$. Wir sagen, dass p *einfach* ist, falls $v_i \neq v_j$ für alle $1 \leq i < j \leq n$ mit $(i,j) \neq (1,k)$.

a) Betrachten Sie die folgenden Probleme:

2VERTEXDISJOINTPATHS (2VDP)	
Gegeben:	Ein gerichteter Graph G und Knoten s_1, s_2, t_1, t_2 von G.
Frage:	Gibt es Pfade p_1 von s_1 nach t_1 und p_2 von s_2 nach t_2 in G, die Knoten-disjunkt sind, d.h., wenn p_1 einen Knoten benutzt, darf p_2 diesen nicht benutzen und umgekehrt?

SIMPLECYCLE (SC)	
Gegeben:	Ein gerichteter Graph G und zwei Kanten e_1, e_2 von G.
Frage:	Gibt es einen gerichteten einfachen Kreis in G, der die Kanten e_1 und e_2 verwendet?

Zeigen Sie die NP-Vollständigkeit von SC. Nutzen Sie dafür, dass das Problem 2VDP NP-vollständig ist. Dies muss nicht gezeigt werden.

b) Betrachten Sie die folgenden Probleme:

UNDIRECTED2VERTEXDISJOINTPATHS (U2VDP)	
Gegeben:	Ein **ungerichteter** Graph G und Knoten s_1, s_2, t_1, t_2 von G.
Frage:	Gibt es Pfade p_1 zwischen s_1 und t_1 und p_2 zwischen s_2 und t_2 in G, die Knoten-disjunkt sind, d.h., wenn p_1 einen Knoten benutzt, darf p_2 diesen nicht benutzen und umgekehrt?

UNDIRECTEDSIMPLECYCLE (USC)	
Gegeben:	Ein **ungerichteter** Graph G und zwei Kanten e_1, e_2 von G.
Frage:	Gibt es einen ungerichteten einfachen Kreis in G, der die Kanten e_1 und e_2 verwendet?

Es ist bekannt, dass das Problem U2VDP in P ist. Dies muss nicht gezeigt werden.

Sei A ein Algorithmus mit polynomieller Laufzeit in der Größe der Eingabe für U2VDP. Zeigen Sie, dass USC in P ist, indem Sie einen (korrekten) Algorithmus für USC konstruieren und begründen, warum dieser eine polynomielle Laufzeit in der Größe der Eingabe hat. Sie müssen nicht formal beweisen, dass Ihr Algorithmus korrekt ist.

Thema Nr. 2
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Teilaufgabe I: Algorithmen

Aufgabe 1 (Asymptotische Notation) [30 PUNKTE]

a) Sortieren Sie die folgenden Funktionen nach ihrem asymptotischen Wachstum, so dass $f_i(n) \in O(f_{i+1}(n))$ gilt. Markieren Sie auch, ob $f_i(n) \in \Theta(f_{i+1}(n))$ gilt.

$$5^n, \quad \sqrt{n}, \quad \log_2(7^{\sqrt{n}}), \quad n^{\frac{1}{\log_2 n}}, \quad 2^{2n}, \quad n!, \quad (\log_2 n)^7$$

b) Zeigen Sie formal, dass $\log_2(n!) \in \Theta(n \log_2 n)$.

Aufgabe 2 (Laufzeit) [18 PUNKTE]

a) Gegeben ist der folgende Algorithmus:

```
1 float g(int A[], int f, int n) {
2     if (n <= 0) return 0;
3     if (n == 1) return A[f];
4     int m0 = n / 2;
5     int m1 = n − m0;
6     float x0 = g(A, f, m0);
7     float x1 = g(A, f + m0, m1);
8     return (x0*m0 + x1*m1) / n;
9 }
```

Beschreiben Sie konzeptionell, was dieser Algorithmus bei Eingabe f = 0 und n gleich der Länge des Feldes macht.

Welche **bestmögliche** asymptotische **obere** Schranke besitzt die Worst-Case-Laufzeit dieses Algorithmus? Begründen Sie Ihre Antwort!

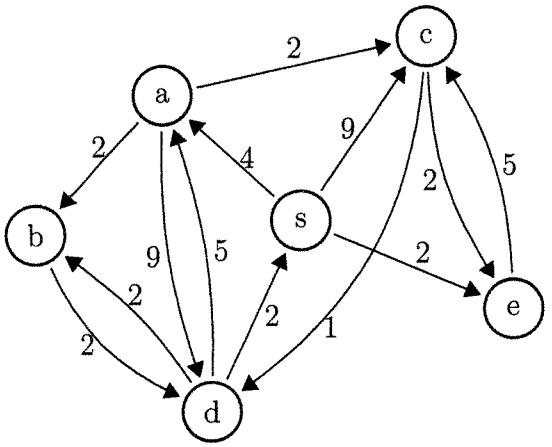
(b) Gegeben ist der folgende Algorithmus:

```
1  int h(int A[], int f, int n) {
2      if (n <= 0) return 0;
3      int m = n/2;
4      int r;
5      if (A[m+f] > 0) {
6          r = h(A, f, n/4);
7      } else {
8          r = h(A, f+m+1, n/4)
9      }
10     return r + A[m+f];
11 }
```

Welche **bestmögliche** asymptotische **obere** Schranke besitzt die Worst-Case-Laufzeit dieses Algorithmus? Begründen Sie Ihre Antwort!

Aufgabe 3 (Dijkstra) [25 PUNKTE]

Gegeben ist der folgende Graph:



a) Geben Sie die **starken Zusammenhangskomponenten** des obigen Graphen an.

b) Finden Sie mit Hilfe von **Dijkstras** Algorithmus die kürzesten Wege vom Knoten s zu allen anderen Knoten.

Notieren Sie dabei in jedem Schritt den aktuell bearbeiteten Knoten, sowie das Parent-Array[1] und den Inhalt der Priority Queue, nachdem der Knoten abgearbeitet wurde. Übertragen Sie dazu die folgende Tabelle auf Ihren Bearbeitungsbogen.

[1]Ein Parent-Knoten ist der Vorgängerknoten im aktuellen Kürzeste-Wege-Baum.

Schritt	aktueller Knoten mit Distanz	Parent-Array a	b	c	d	e	s	Inhalt der Priority-Queue
0.	-						-1	$(s,0),(a,\infty),(b,\infty),(c,\infty),(d,\infty),(e,\infty)$
							-1	
							-1	
							-1	

.....

Aufgabe 4 (Algorithmenentwurf) [47 PUNKTE]

Das Ziel der folgenden Aufgabe ist es, möglichst effiziente Algorithmen zu entwickeln, d.h. korrekte Algorithmen mit einer suboptimalen Laufzeit erhalten nicht die volle Punktzahl (Sie müssen aber nicht die Optimalität Ihres Algorithmus zeigen.)

a) Es sind zwei **sortierte** Ganzzahlarrays A und B der Länge n und ein Integer k mit $1 \leq k \leq n$ gegeben. Die Zahlen in den Arrays sind **paarweise verschieden**.

Sei $M = \{A[1], \ldots, A[n]\} \cup \{B[1], \ldots, B[n]\}$ die Menge der Elemente in beiden Arrays. Wir möchten die k-kleinste Zahl in M bestimmen.

i) Sei $0 < k_a < k$. Betrachten Sie die Elemente $A[k_a]$, $A[k_a + 1]$, $B[k - k_a]$, $B[k - k_a + 1]$. Welche Beziehungen zwischen diesen vier Elementen müssen gelten, damit $A[k_a]$ das k-kleinste Element ist? Begründen Sie Ihre Antwort!

ii) Entwickeln Sie einen Algorithmus, der die k-kleinste Zahl in Zeit $O(\log k)$ bestimmt. Benutzen Sie Ihre Erkenntnisse aus der vorherigen Teilaufgabe.

Implementieren Sie die Funktion `kth_smallest` in Pseudocode oder einer objektorientierten Programmiersprache Ihrer Wahl. Die Funktion bekommt die beiden Arrays A und B und k als Parameter.

Die Laufzeit der Methode `kth_smallest` soll in $O(\log k)$ sein.

b) Gegeben sei ein gerichteter Graph $G = (V, E)$. Entwickeln Sie einen Algorithmus, der in linearer Zeit bestimmt, ob es einen Knoten $s \in V$ gibt, der alle anderen Knoten erreichen kann.

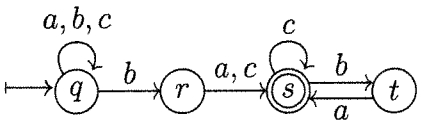
Beschreiben Sie die Idee Ihres Algorithmus. Pseudocode ist nicht verlangt. Begründen Sie die Korrektheit Ihres Algorithmus und analysieren Sie die Laufzeit!

Hinweis: Benutzen Sie DFS!

Teilaufgabe II: Theoretische Informatik

Aufgabe 1 (Reguläre Sprachen) [32 PUNKTE]

a) Es sei $L \subseteq \{a, b, c\}^*$ die von dem folgenden nichtdeterministischen Automaten akzeptierte Sprache:



Geben Sie einen regulären Ausdruck für die Sprache L an.

b) Geben Sie einen (vollständigen) deterministischen endlichen Automaten für das Komplement $\{a, b, c\}^* \setminus L$ an. Machen Sie hierbei Ihren Lösungsweg deutlich (z.B. den Rechenweg eines allgemeinen Verfahrens).

c) Minimieren Sie den deterministischen Automaten

$$A = (\{q, r, s, t, x, y, z\}, \{0, 1\}, \delta, s, \{q, z\})$$

mit der tabellarisch gegebenen Zustandsüberführungsfunktion

δ	q	r	s	t	x	y	z
0	x	q	y	q	r	z	x
1	r	x	q	y	z	x	y

Verwenden Sie ein allgemeines Verfahren und machen Sie Ihren Rechenweg deutlich!

Aufgabe 2 (Kontextfreie Sprachen) [24 PUNKTE]

a) Entwerfen Sie eine kontextfreie Grammatik für die folgende kontextfreie Sprache über dem Alphabet $\Sigma = \{0, 1, 2\}$:

$$L = \{w1^n v \mid n \in \mathbb{N}, w, v \in \{0, 2\}^*, n = |w|_0 + |v|_2\}.$$

Erklären Sie den Zweck der einzelnen Nichtterminale (Variablen) und der Regeln (Produktionen) Ihrer Grammatik.

b) Geben Sie eine Ableitung des Wortes 20221112020 mit Ihrer Grammatik an.

c) Beweisen Sie, dass die folgende formale Sprache über $\Sigma = \{0, 1, 2\}$ nicht kontextfrei ist:

$$L = \{w1^n v \mid n \in \mathbb{N}, v \in \{0, 2\}^*, n = |w|_0 \cdot |v|_2\}.$$

Aufgabe 3 (Entscheidbarkeit) [20 PUNKTE]

Gegeben ist das folgende Entscheidungsproblem:

Eingabe: eine Turingmaschine M mit Eingabealphabet $\Sigma = \{0, 1\}$, die eine formale Sprache $L \subseteq \Sigma^*$ akzeptiert.

Aufgabe: entscheiden, ob es für jedes $n \leq 42$ ein Wort $w \in \Sigma^*$ mit $|w| = n$ gibt, das M akzeptiert.

a) Beweisen Sie, dass das gegebene Problem semi-entscheidbar ist.

b) Beweisen oder widerlegen Sie, dass das gegebene Problem entscheidbar ist.

Aufgabe 4 (NP-Vollständigkeit) [20 PUNKTE]

Sei $G = (V, E)$ ein Graph. Eine *fast unabhängige Menge* ist eine Menge $V' \subseteq V$ von Knoten, sodass höchstens eine Kante zwischen Knoten aus V' in G existiert.

Eingabe: ein ungerichteter Graph $G = (V, E)$ und eine natürliche Zahl k.

Aufgabe: entscheiden, ob G eine fast unabhängige Menge mit mindestens k Knoten hat.

a) Zeigen Sie, dass das Problem in NP liegt.

b) Zeigen Sie, dass das Problem NP-vollständig ist.

Hinweise:

- Sie dürfen verwenden, dass das Problem *unabhängige Menge* NP-schwer ist.
- Sei $G = (V, E)$ ein Graph. Eine *unabhängige Menge* ist eine Menge $V' \subseteq V$ von Knoten, sodass keine Kante zwischen Knoten aus V' in G existiert.

Aufgabe 5 (Aussagen) [24 PUNKTE]

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

Notieren Sie zunächst zu jeder Aussage, ob sie richtig oder falsch ist. Begründen Sie Ihre jeweilige Entscheidung!

a) Es gibt ein Alphabet Σ über dem alle formalen Sprachen unentscheidbar sind.

b) Für formale Sprachen L über dem Alphabet $\Sigma = \{a\}$ gilt: falls L^* regulär ist, dann ist auch L regulär.

c) Wenn L unentscheidbar ist, liegt sein Komplement nicht in der Klasse NP.

d) Es sei L eine formale Sprache, die von einer Turingmaschine M in konstanter Zeit entschieden wird (d.h. es existiert eine natürliche Zahl k, so dass M bei jedem Eingabewort in höchstens k Schritten hält und das Eingabewort akzeptiert oder ablehnt). Dann ist L regulär.