Berlin, 28.07.2026

Name: ..... **Matr.-Nr.: .....**

Klausur Algorithmentheorie

(Froese/Breitkopf, Sommersemester 2026)

Aufgabe Nr.:	1	2	3	4	5	6	7	8	**Summe**
Punktzahl:	15	15	15	16	15	15	15	14	120
Davon erreicht:									

Bearbeitungszeit:	120 Minuten
Max. Punktezahl:	120 Punkte
Bestanden:	ab 50 Punkten
Bestnote:	ab 95 Punkten

Allgemeine Hinweise:

- Als Hilfsmittel ist nur ein handschriftlich beschriebenes Blatt (A4) erlaubt.
- Benutzen Sie keinen Bleistift, sondern einen Kugelschreiber in der Farbe Schwarz oder Blau. Insbesondere sind Rot und Grün nicht erlaubt.
- Beschriften Sie jedes Blatt mit Matrikelnummer, das Titelblatt zusätzlich mit Vor- und Nachnamen.
- Falls in der Aufgabenstellung nicht explizit ausgeschlossen, so sind alle Antworten zu begründen! Insbesondere muss bei Algorithmen die Korrektheit und die Laufzeit argumentiert werden.
- Auf der letzten Seite finden Sie einige Problemdefinition und Sätze, die Sie verwenden können.

Viel Erfolg!

Aufgabe 1: **Greedy Algorithmen** (15 Punkte)

Sie organisieren einen Triathlon für n Personen. Für jede Person $i \in \{1, \ldots, n\}$ kennen Sie die Zeiten, die diese Person zum Schwimmen ($s_i > 0$), Fahrradfahren ($f_i > 0$) bzw. Laufen ($\ell_i > 0$) benötigt. Die Disziplinen müssen in dieser Reihenfolge absolviert werden. Für die Durchführung steht nur ein Schwimmbecken zur Verfügung, in dem sich nie zwei Personen gleichzeitig aufhalten dürfen. Das heißt, die erste Person beginnt zu schwimmen und sobald diese fertig ist, kann die zweite Person beginnen, zu schwimmen. Die Fahrradstrecke und die Laufbahn können von beliebig vielen Personen gleichzeitig genutzt werden. Die Zeit zwischen den Disziplinen zum Umziehen etc. wird ignoriert.

Der Wettkampf beginnt zum Zeitpunkt 0 und endet an dem Zeitpunkt Z, an dem die letzte Person ins Ziel kommt.

Geben Sie einen Algorithmus an, der in $\mathcal{O}(n \log n)$ Rechenschritten die Reihenfolge der Teilnehmenden so festlegt, dass die Wettkampfdauer Z minimiert wird.

Matr.-Nr.: .....

Aufgabe 2: **Divide and Conquer (Fast-Fourier-Transformation)** (15 Punkte)

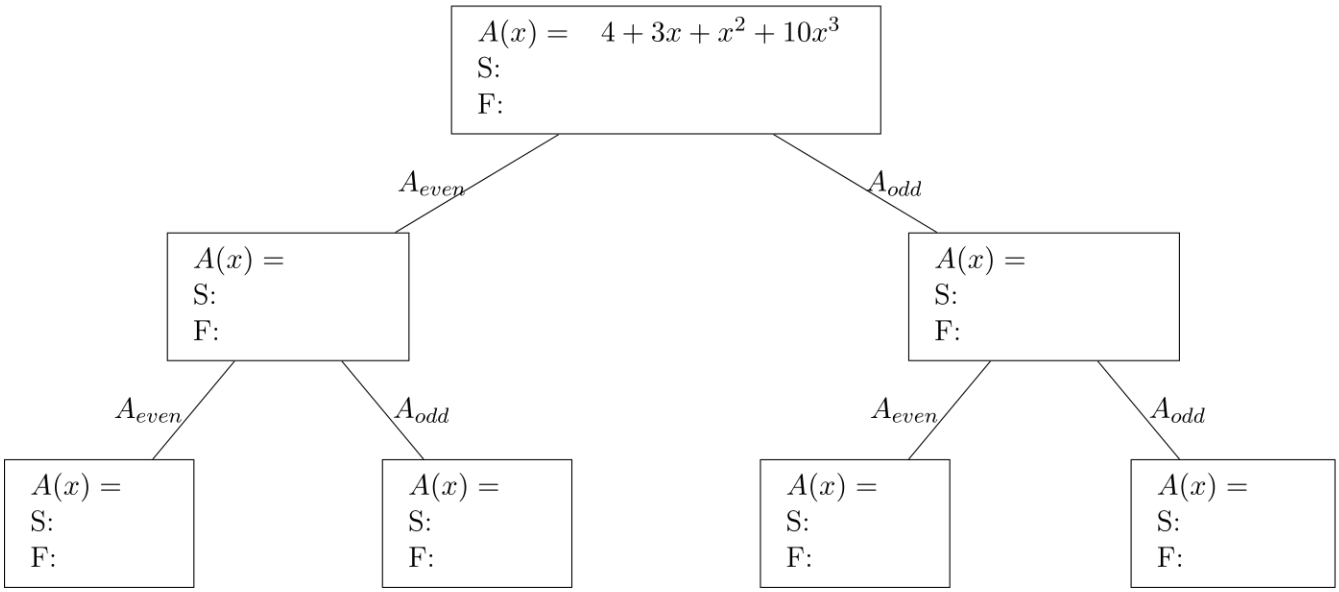
Gesucht ist die Stützstellendarstellung des Polynoms:

$$A(x) = 4 + 3x + x^2 + 10x^3$$

(a) Wieviele Stützstellen sind nötig, um A eindeutig zu beschreiben? (ohne Begründung) (3 P)

(b) Berechnen Sie eine Stützstellendarstellung mit Hilfe der Fast-Fourier-Transformation, indem Sie folgende Abbildung vervollständigen. (12 P)

Tragen Sie dazu in der ersten Zeile jeweils das Polynom ein, das der Algorithmus im entsprechenden (rekursiven) Aufruf betrachtet. Tragen Sie in der zweiten Zeile (S) die Stützstellen ein, an denen das Polynom ausgewertet wird und tragen Sie in die dritte Zeile (F) die Funktionswerte des Polynoms an diesen Stützstellen ein.



Im Folgenden sei $\omega_n := e^{2\pi i/n} = \cos(2\pi/n) + i\sin(2\pi/n)$ für $n \geq 1$. (Z.B. gilt $\omega_1 = 1$, $\omega_2 = -1$, und $\omega_4 = i$.)

Fast-Fourier-Transformation:

Gegeben ein Polynom $A(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}$ vom Grad $n-1$ (für $n = 2^N$ mit $N \geq 0$), gehe folgendermaßen vor:

1. Falls $n = 1$, gib $A(\omega_1) = a_0$ zurück.
2. Setze $A_{even}(x) := a_0 + a_2 x + \cdots + a_{n-2}x^{(n-2)/2}$ und $A_{odd}(x) := a_1 + a_3 x + \cdots + a_{n-1}x^{(n-2)/2}$.
3. Berechne rekursiv $A_{even}(\omega_{n/2}^k)$ und $A_{odd}(\omega_{n/2}^k)$ für $k \in \{0, \ldots, n/2 - 1\}$.
4. Berechne für $k \in \{0, \ldots, n/2 - 1\}$:

$$A(\omega_n^k) = A_{even}(\omega_{n/2}^k) + \omega_n^k A_{odd}(\omega_{n/2}^k)$$
$$A(\omega_n^{k+n/2}) = A_{even}(\omega_{n/2}^k) - \omega_n^k A_{odd}(\omega_{n/2}^k)$$

Aufgabe 3: **Dynamic Programming** (15 Punkte)

Gegeben seien zwei Zeichenketten $X, Y \in \Sigma^*$ der Länge $n \geq 1$ über einem endlichen Alphabet Σ.

Eine Zeichenkette $S \in \Sigma^*$ ist Teilsequenz einer anderen Zeichenkette $T \in \Sigma^*$, genau dann, wenn man S durch Löschen von Zeichen aus T erhalten kann. Die Zeichenkette S ist gemeinsame Teilsequenz von X und Y, falls S Teilsequenz von X und von Y ist.

(a) Geben Sie das dynamische Programm aus der Vorlesung an, welches die Länge einer längsten gemeinsamen Teilsequenz von X und Y berechnet (ohne Begründung der Korrektheit). Wie ist die Laufzeit? (5 P)

(b) Wie lässt sich eine längste gemeinsame Teilsequenz berechnen? (3 P)

(c) Adaptieren Sie den Algorithmus, sodass er die Länge einer längsten gemeinsamen Teilsequenz für drei Zeichenketten X, Y, Z der Länge $n \geq 1$ in $\mathcal{O}(n^3)$ Zeit findet. (7 P)

Matr.-Nr.: .....

Aufgabe 4: **Vermischtes** (16 Punkte)

Bei jeder der folgenden Multiple-Choice Fragen ist mindestens eine Antwort korrekt. Jedes richtige Kreuz gibt anteilig Punkte (bei n richtigen Antworten, ist jede $1/n$ der Punkte Wert). Sobald ein falsches Kreuz gesetzt wird, gibt es auf die Frage 0 Punkte! (Ohne Begründung)

(a) Welche der folgenden Aussagen sind korrekt? (3 P)

- ☐ $3^{\log_2 n} \in \mathcal{O}(n)$
- ☐ $3^{\log_2 n} \in \mathcal{O}(n^2)$
- ☐ $3n^2 \log_{10} n \in \mathcal{O}(n^3)$
- ☐ $2^{n^2} \in \mathcal{O}(3^n)$

(b) Gegeben sei ein rekursiver Algorithmus, für dessen Laufzeit $T(n)$ Folgendes gilt: $T(n) = 16T(\lceil n/4 \rceil) + \mathcal{O}(n^3)$. Welche der folgenden Aussagen sind korrekt? (4 P)

- ☐ $T(n) \in \mathcal{O}(n^2)$
- ☐ $T(n) \in \mathcal{O}(n^3 \log n)$
- ☐ $T(n) \in \mathcal{O}(n^4)$
- ☐ $T(n) \in \mathcal{O}(n^4 \log n)$
- ☐ $T(n) \in \mathcal{O}(n^7)$

(c) Im Folgenden nehmen wir P $\neq$ NP an. Welche der folgenden Aussagen sind korrekt? (3 P)

- ☐ KNF-SAT parametrisiert mit der maximalen Klauselgröße ist in FPT.
- ☐ Lineares Programmieren $\leq_p$ Vertex Cover
- ☐ Vertex Cover $\leq_p$ Lineares Programmieren
- ☐ Vertex Cover besitzt ein Polynomzeit-Approximationsschema (PTAS).

(d) Welche der folgenden Aussagen sind korrekt? (6 P)

- ☐ Knapsack ist in P für konstantes Gesamtgewicht W.
- ☐ Der Algorithmus von Ford-Fulkerson hat eine polynomielle Laufzeit.
- ☐ Jedes entscheidbare Problem ist FPT für den Parameter Eingabegröße.
- ☐ Vertex Cover hat eine 2-Approximation.
- ☐ Der Algorithmus von Bellman-Ford zum Finden kürzester Pfade funktioniert auch für negative Kantengewichte.

Aufgabe 5: **Netzwerkflüsse** (15 Punkte)

(a) Eine Menge von n Personen $p_1, \ldots, p_n$ möchte am Semesterende eine Prüfung ablegen. Zur Auswahl stehen m mögliche Termine $t_1, \ldots, t_m$. Jede Person p_i hat nur zu einer bestimmten Menge $T_i \subseteq \{t_1, \ldots, t_m\}$ von möglichen Terminen Zeit für eine Prüfung. Außerdem muss jede Person die Prüfung an genau einem Termin ablegen. Allerdings können Prüfungen in Gruppen von maximal drei Personen gleichzeitig abgelegt werden. Ist es möglich, jeder Person einen ihrer möglichen Prüfungstermine zuzuordnen, sodass an keinem Termin mehr als drei Personen geprüft werden? (10 P)

Modellieren Sie obiges Problem als ein Maximum Flow-Problem, sodass die Anzahl der Knoten und die Anzahl der Kanten im Flussnetzwerk höchstens polynomiell in n und m ist. Geben Sie hierfür die Knoten, Kanten und Kantenkapazitäten Ihres Flussnetzwerks an und erklären Sie, wie die Antwort bestimmt werden kann.

(b) Gegeben sei ein Flussnetzwerk bestehend aus dem gericheten Graphen $G = (V, E)$ mit Kantenkapazitäten $c\colon E \to \mathbb{N}^+$, Quelle $s \in V$ und Senke $t \in V$. Außerdem sei $f\colon E \to \mathbb{N}$ ein größtmöglicher ganzzahliger s–t–Fluss in G. Betrachten Sie das Flussnetzwerk, in dem die Kapazität einer einzigen Kante $x \in E$ um eins erhöht wird, welches also Kantenkapazitäten $c'\colon E \to \mathbb{N}^+$ besitzt mit: (5 P)

$$c'(e) \coloneqq \begin{cases} c(e) + 1, & \text{falls } e = x \\ c(e), & \text{sonst.} \end{cases}$$

Zeigen Sie: Ein größtmöglicher s–t–Fluss im modifizierten Flussnetzwerk kann unter Verwendung von f in $\mathcal{O}(|V| + |E|)$ Zeit gefunden werden.

Matr.-Nr.: .....

Aufgabe 6: **Lineares Programmieren** (15 Punkte)

(a) Betrachten Sie folgendes lineares Programm (LP) mit einer Variable x und $a, b, c \in \mathbb{Q}$. Geben Sie das duale LP an (ohne Begründung). (2 P)

$$\begin{aligned} &\max\ cx \\ \text{N.B.:}\ &ax \leq b \\ &x \geq 0 \end{aligned}$$

(b) Nutzen Sie obiges LP und widerlegen Sie durch eine geeignete Wahl der Koeffizienten a, b, c die folgende Aussage: (5 P)

Falls ein LP eine ganzzahlige Lösung mit optimalem Zielwert besitzt, so besitzt auch das duale LP eine ganzzahlige Lösung mit optimalem Zielwert.

(c) Modellieren Sie folgendes Problem als ein ganzzahliges lineares Programm mit polynomiell (in $|V|$) vielen Variablen und Nebenbedingungen. (8 P)

Erklären Sie die Bedeutung Ihrer Variablen und Nebenbedingungen. (Die Korrekheit muss nicht begründet werden.)

Partial Dominating Set
Eingabe: Graph $G = (V, E)$, $k \in \mathbb{N}$.
Aufgabe: Finde ein Knotenteilmenge $X \subseteq V$ mit $|X| \leq k$, sodass die Anzahl von Knoten, die mindestens einen Nachbarn in X haben oder selber in X enthalten sind, größtmöglich ist.

Zu einem primalen LP mit Vektoren $c \in \mathbb{Q}^n, b \in \mathbb{Q}^m$, Matrix $A \in \mathbb{Q}^{m \times n}$ und Variablen $x \in \mathbb{Q}^n$ wird das duale LP mit Variablen $y \in \mathbb{Q}^m$ wie folgt gebildet:

Primales LP:

$$\begin{aligned} \max\ &c^T x \\ \text{N.B.:}\ &Ax \leq b \\ &x \geq 0 \end{aligned}$$

Duales LP:

$$\begin{aligned} \min\ &y^T b \\ \text{N.B.:}\ &y^T A \geq c^T \\ &y \geq 0 \end{aligned}$$

Aufgabe 7: **FPT / Datenreduktion** (15 Punkte)

Ein parametrisiertes Problem $L \in \{0,1\}^* \times \mathbb{N}$ ist *fixed-parameter tractable* (in FPT), falls ein Algorithmus existiert, der für jede Instanz (x,k) in $f(k) \cdot |x|^{O(1)}$ Zeit entscheidet, ob $(x,k) \in L$, wobei f eine beliebige berechenbare Funktion ist, die nur von k abhängt.

Ein *Problemkern* für ein parametrisiertes Problem L ist ein Polynomzeitalgorithmus, der für eine gegebene Instanz (x,k) eine Instanz (x',k') berechnet, sodass gilt:

$(x,k) \in L \iff (x',k') \in L$ und $|x'| + k' \leq f(k)$ für eine beliebige Funktion f, die nur von k abhängt.

(a) Zeigen Sie, dass jedes entscheidbare parametrisierte Problem L fixed-parameter tractable ist, falls es einen Problemkern besitzt. (6 P)

(b) Das 3-Coloring Problem ist wie folgt definiert: (9 P)

3-Coloring
Eingabe: Ein Graph $G = (V,E)$.
Frage: Ist G 3-färbbar, d.h. existiert eine Knotenfärbung $c\colon V \to \{1,2,3\}$, sodass $c(u) \neq c(v)$ für alle $\{u,v\} \in E$?

Zeigen Sie, dass 3-Coloring einen Problemkern für den Parameter Feedback-Edge-Zahl[1] besitzt.

[1] Die Feedback-Edge-Zahl $a(G)$ eines Graphen G ist die minimale Zahl von Kanten, deren Löschung den Graph kreisfrei macht.

Matr.-Nr.: .....

Aufgabe 8: **Approximation** (14 Punkte)

(a) Das KNAPSACK-Problem ist wie folgt definiert. (7 P)

KNAPSACK
Eingabe: Gegenstände $U = \{1, \ldots, n\}$, wobei Gegenstand $i \in U$ Gewicht $w_i > 0$ und Wert $v_i > 0$ hat, Fassungsvermögen W.
Aufgabe: Finde ein $X \subseteq U$ mit $\sum_{i \in X} w_i \leq W$, das den Wert $\sum_{i \in X} v_i$ maximiert.

Liefert der folgende Greedy Algorithmus eine α-Approximation für KNAPSACK für ein $\alpha \in (0, 1]$, d. h. liefert er immer eine Lösung, die mindestens α mal dem Wert einer optimalen Lösung entspricht?

1: $X \leftarrow \emptyset$
2: **while** $\exists i \in U$ mit $w_i \leq W$ **do**
3: Wähle Gegenstand i mit größtem Wert v_i unter der Bedingung $w_i \leq W$
4: $X \leftarrow X \cup \{i\}$
5: $W \leftarrow W - w_i$
6: $U \leftarrow U \setminus \{i\}$
7: **return** X

(b) Das MAXIMUM MATCHING-Problem ist wie folgt definiert. (7 P)

MAXIMUM MATCHING
Eingabe: Graph $G = (V, E)$.
Aufgabe: Finde eine größtmögliche Kantenmenge $M \subseteq E$, sodass für alle $e, e' \in M$ mit $e \neq e'$ gilt: $e \cap e' = \emptyset$.

Liefert der folgende Algorithmus eine $1/2$-Approximation für MAXIMUM MATCHING, d. h. liefert er immer eine Lösung, die mindestens $1/2$ mal so viele Kanten enthält wie eine optimale Lösung?

1: $M \leftarrow \emptyset$
2: **while** $\exists e \in E(G)$ **do**
3: Wähle beliebige Kante $e = \{u, v\} \in E(G)$
4: $M \leftarrow M \cup \{e\}$
5: $G \leftarrow G[V \setminus \{u, v\}]$
6: **return** M

Problemdefinitionen, Theoreme und Sätze aus der Vorlesung

Vertex Cover
Eingabe: Graph $G = (V, E)$, $k \in \mathbb{N}$
Frage: Existiert $X \subseteq V$ mit $|V| \leq k$, sodass $e \cap X \neq \emptyset$ für jede Kante $e \in E$?

Lineares Programmieren
Eingabe: Vektoren $c \in \mathbb{Q}^n, b \in \mathbb{Q}^m$, Matrix $A \in \mathbb{Q}^{m \times n}$, $k \in \mathbb{N}$.
Frage: Existiert ein $x \in \mathbb{Q}^n$, sodass $x \geq 0$, $Ax \leq b$ und $c^T x \geq k$ gilt?

Theorem 1 (Master Theorem)**.** *Sei $T(n) = aT(\lceil n/b \rceil) + \mathcal{O}(n^d)$ für Konstanten $a > 0$, $b > 1$ und $d \geq 0$. Dann ist*

$$T(n) = \begin{cases} \mathcal{O}(n^d) & \text{falls } d > \log_b a, \\ \mathcal{O}(n^d \log n) & \text{falls } d = \log_b a, \\ \mathcal{O}(n^{\log_b a}) & \text{falls } d < \log_b a. \end{cases}$$

Satz 1. *Eine topologische Sortierung eines DAG (gerichteten kreisfreien Graphen) kann in Linearzeit (d. h. $\mathcal{O}(n + m)$ Zeit) berechnet werden.*

Satz 2. *In dem Restgraph $G_f = (V, E)$ zu einem Fluss f kann in Zeit $\mathcal{O}(|E|)$ ein augmentierender Pfad gefunden werden oder festgestellt werden, dass keiner existiert.*

Theorem 2 (MaxFlow-MinCut)**.** *Ein s–t–Fluss ist genau dann größtmöglich, wenn der Ford/Fulkerson-Algorithmus keine augmentierenden Pfade mehr findet.*

Der Wert eines größtmöglichen s–t–Flusses gleicht dem Wert eines kleinstmöglichen s–t–Schnitts.

Satz 3. *Sei $G = (V, E)$ ein Graph, in dem jeder Knoten mindestens Grad drei hat und sei $a(G)$ die Feedback-Edge-Zahl von G. Dann gilt $|V| \leq 4 \cdot a(G)$.*