

Nachname (Druckschrift): _____

Vorname (Druckschrift): _____

Matrikelnummer: _____

Studiengang: _____

Bitte Hinweise beachten:

- Schreiben Sie Ihren **Namen *nur auf dieses Titelblatt***. Sollten Sie Zusatzpapier bekommen, vermerken Sie darauf unbedingt die Klausurnummer **0000**.
- Die Klausur besteht aus den Aufgaben **1 – 10**.
- Merken oder notieren Sie sich Ihre Klausurnummer **0000**, da nur unter dieser Nummer die Ergebnisse veröffentlicht werden.
- Es dürfen nur **dokumentenechte Stifte** in den Farben blau und schwarz verwendet werden. Insb. ist die Nutzung von Tintenlöschern, Tippex u. Ä. untersagt.
Zugelassene Hilfsmittel:
1 Blatt DIN A4 mit handschriftlichen Notizen (zweiseitig).
- Das Mitbringen nicht zugelassener Hilfsmittel stellt einen Täuschungsversuch dar und führt zum Nichtbestehen der Klausur. **Schalten Sie bitte deshalb alle elektronischen Geräte, insbesondere Handys und Smartwatches, vor Beginn der Klausur aus und packen Sie diese weg.**
- Werden zu einer Aufgabe zwei oder mehr Lösungen angegeben, so gilt die Aufgabe als nicht gelöst. Entscheiden Sie sich also immer für **eine** Lösung. Begründungen sind nur dann notwendig, wenn die Aufgabenformulierung dies verlangt.
- Die Klausur ist mit Sicherheit bestanden, wenn (ohne Bonifikation aus den Übungspunkten) mindestens **50%** der Höchstpunktzahl erreicht wird.
- Die Klausur dauert **180 Minuten**.

**Diese Seite ist nur für den internen Gebrauch bestimmt.
Bitte nicht beschreiben.**

Aufgabe	1	2	3	4	5	6	7	8	9	10
Erreichbar	6	6	7	8	12	13	12	9	11	16
Erreicht										

Klausur	Bonifikation

Betrachten Sie die folgenden Sequenzen der Länge n bestehend aus natürlichen Zahlen. Geben Sie in Abhängigkeit der Länge n asymptotisch exakt an, wie lange die entsprechenden Algorithmen zum Sortieren der Sequenz benötigen.

Anmerkungen: Bubblesort terminiert nach einem Durchlauf ohne Vertauschung. Sie können außerdem davon ausgehen, dass $\log_2 n$ und \sqrt{n} ebenfalls natürliche Zahlen sind.

a)

$$\log_2 n, \dots, 1, 2 \cdot \log_2 n, \dots, 1 + \log_2 n, \dots, n, \dots, n + 1 - \log_2 n$$

d.h. absteigend sortierte Teilfolgen der Länge $\log_2 n$.

Bubblesort: $\Theta(\text{_____})$

Insertionsort: $\Theta(\text{_____})$

Selectionsort: $\Theta(\text{_____})$

↑↑↑ _____ / 3 Punkt(e) ↑↑↑

b)

$$1, \dots, \sqrt{n} - 1, \sqrt{n} + 1, \dots, n, \sqrt{n}$$

d.h. die Folge ist sortiert bis auf das Element \sqrt{n} , das sich am Ende befindet.

Bubblesort: $\Theta(\text{_____})$

Insertionsort: $\Theta(\text{_____})$

Selectionsort: $\Theta(\text{_____})$

↑↑↑ _____ / 3 Punkt(e) ↑↑↑



Gegeben seien n natürliche Zahlen der Form N^{k^N} für $k \leq n$ und ein beliebig großes, aber bekanntes, $N \in \mathbb{N}$.

Gibt es für solche Sequenzen eine Transformation, die jedem Element einen neuen Schlüssel zuordnet, sodass die transformierte Sequenz in Zeit $\mathcal{O}(n)$ sortiert werden kann? Geben Sie eine Transformation sowie einen Sortieralgorithmus an, mit dem dies möglich ist, oder begründen Sie, warum eine solche Transformation nicht existiert.

↑↑↑ _____ / 6 Punkt(e) ↑↑↑



Bubblesort ist mit seiner Worst-Case Laufzeit von $\Theta(n^2)$ normalerweise nicht der Sortieralgorithmus der Wahl. Mit den Erfahrungen aus Quicksort wollen wir versuchen, Bubblesort zu verbessern.

Sei A eine Liste von n Elementen. Sie können davon ausgehen, dass n eine Zweierpotenz ist.

QUICKBUBBLESORT:

- Falls $n = 1$: A ist sortiert, **return**.
- Falls $n > 1$:
 1. Sortiere $A[1, \dots, \frac{n}{2}]$ und $A[\frac{n}{2} + 1, \dots, n]$ rekursiv mit QUICKBUBBLESORT.
 2. Mische die beiden sortierten Teilfolgen mit Bubblesort.

Die Hoffnung: Da QUICKBUBBLESORT die Liste in der Mitte teilt, hat es eine Rekursionstiefe von $\log n$ und somit eine Gesamtlaufzeit von $\mathcal{O}(n \log n)$.

Ist QUICKBUBBLESORT wirklich schneller als Bubblesort? Begründen Sie Ihre Antwort.

↑↑↑ _____ / 7 Punkt(e) ↑↑↑



Betrachten Sie folgende Aussage:

Wenn L eine entscheidbare Sprache ist, ist auch ihr Komplement \bar{L} entscheidbar.
Da entscheidbare Sprachen rekursiv aufzählbar sind, folgt: Ist eine Sprache L rekursiv aufzählbar, dann auch ihr Komplement \bar{L} .

Ist diese Aussage korrekt? Begründen Sie Ihre Antwort.

↑↑↑ _____ / 8 Punkt(e) ↑↑↑



Gegeben sei die folgende Turingmaschine $M = (Q, \Sigma, \delta, q_0, \Gamma, F)$ mit

$$Q = \{q_0, q_1, q_2\}$$

$$\Sigma = \{0, 1\}$$

$$\Gamma = \{B\} \cup \Sigma$$

$$F = \{q_2\}$$

und der Übergangsfunktion δ definiert als

$$(q_0, \alpha) \mapsto (q_0, \alpha, rechts) \quad \forall \alpha \in \Sigma$$

$$(q_0, B) \mapsto (q_1, B, links)$$

$$(q_1, 1) \mapsto (q_0, 1, rechts)$$

$$(q_1, 0) \mapsto (q_2, 0, bleib)$$

- a) Geben Sie ein Wort $w \in \Sigma^*$ an, das von M akzeptiert wird.

$w =$ _____

↑↑↑ _____ / 1 Punkt(e) ↑↑↑

- b) Welche Sprache wird von M akzeptiert?

↑↑↑ _____ / 4 Punkt(e) ↑↑↑

- c) Hält M auf allen Eingaben? Begründen Sie Ihre Antwort und geben Sie ggf. eine Eingabe an, auf der M nicht hält.

↑↑↑ _____ / 5 Punkt(e) ↑↑↑

- d) Gibt es Eingaben, auf denen M in einem nicht-akzeptierenden Zustand hält? Geben Sie ein Beispiel an oder begründen Sie, warum es keine solche Eingabe gibt.

↑↑↑ _____ / 2 Punkt(e) ↑↑↑



Gegeben seien zwei identische Maschinen M_1 und M_2 und fünf Aufgaben $1, \dots, 5$ mit den Laufzeiten t_1, \dots, t_5 :

i	1	2	3	4	5
t_i	9	8	4	5	10

Alle Aufgaben sollen auf die Maschinen verteilt werden mit dem Ziel, den Makespan (frühester Zeitpunkt, zu dem alle Aufgaben fertiggestellt sind) zu minimieren.

Betrachten Sie die folgenden zwei Strategien:

1. Die Aufgaben werden nach absteigenden Laufzeiten sortiert. Die Aufgaben werden dann nacheinander auf die Maschinen verteilt, wobei die aktuelle Aufgabe der Maschine mit der bisher geringsten Last zugeteilt wird.
2. Die Aufgaben werden in irgendeiner Reihenfolge nacheinander auf die Maschinen verteilt, wobei die aktuelle Aufgabe der Maschine mit der bisher geringsten Last zugeteilt wird.

Falls beide Maschinen die gleiche Last haben, dürfen Sie die aktuelle Aufgabe einer beliebigen Maschine zuteilen.

- a) Bestimmen Sie eine optimale Lösung. Zeigen Sie die Optimalität dieser Lösung.

↑↑↑ _____ / 3 Punkt(e) ↑↑↑

- b) Verteilen Sie die Aufgaben an die Maschinen unter Verwendung der 1. Strategie. Geben Sie dabei für jede Aufgabe an, welcher Maschine sie zugeteilt wird. Berechnen Sie die 1. Strategie eine optimale Lösung? Begründen Sie Ihre Antwort.

↑↑↑ _____ / 3 Punkt(e) ↑↑↑



Aufgabe 6: Scheduling (Fortsetzung)

c) Hier ist eine Kopie der Tabelle:

i	1	2	3	4	5
t_i	9	8	4	5	10

Geben Sie eine Reihenfolge der Aufgaben an, sodass mit der 2. Strategie der Makespan 23 oder schlechter ist. Geben Sie dabei für jede Aufgabe an, welcher Maschine sie zugeteilt wird.

↑↑↑ _____ / 7 Punkt(e) ↑↑↑



Entscheiden Sie, ob die folgenden Aussagen stimmen, und begründen Sie kurz Ihre Antwort. Es ist kein Beweis gefordert.

- a) Für das Rucksackproblem gibt es einen $\frac{3}{2}$ -approximativen Algorithmus mit polynomialer Laufzeit in n .

↑↑↑ _____ / 3 Punkt(e) ↑↑↑

- b) Sei SP das Problem zu entscheiden, ob eine Sequenz bestehend aus n natürlichen Zahlen aufsteigend sortiert ist. Dann gibt es eine polynomielle Reduktion von SP auf KNF-SAT.

↑↑↑ _____ / 3 Punkt(e) ↑↑↑

- c) Angenommen $\mathcal{P} \neq \mathcal{NP}$. Dann gibt es eine polynomielle Reduktion von CLIQUE auf 3-SAT.

↑↑↑ _____ / 3 Punkt(e) ↑↑↑

- d) Gegeben sei ein ungerichteter Graph $G = (V, E)$ mit nicht-negativen Kantengewichten und ein Knoten $s \in V$. Es gibt einen effizienten Algorithmus, der zu jedem Knoten $v \in V$ einen Distanzwert berechnet, der höchstens doppelt so groß ist wie die Länge des kürzesten Weges von s nach v .

↑↑↑ _____ / 3 Punkt(e) ↑↑↑



Im \mathcal{NP} -vollständigen SUBSETSUM Problem ist die Eingabe durch eine Menge M von n ganzen Zahlen gegeben. Die Eingabe gehört genau dann zur Sprache SUBSETSUM, wenn eine nicht-leere Teilmenge $M' \subseteq M$ existiert mit folgender Eigenschaft:

$$\sum_{m \in M'} m = 0.$$

Wir wollen eine Einschränkung betrachten: Im 3-SUM Problem ist die Eingabe auch durch eine Menge M von n ganzen Zahlen gegeben. Die Eingabe gehört genau dann zur Sprache, wenn 3 paarweise verschiedene Zahlen $a, b, c \in M$, das heißt $a \neq b, a \neq c$ und $b \neq c$, mit folgender Eigenschaft existieren:

$$a + b + c = 0.$$

- a) Zeigen Sie, dass 3-SUM zur Sprache \mathcal{NP} gehört.

↑↑↑ _____ / 2 Punkt(e) ↑↑↑

- b) Alice behauptet:

„3-SUM ist so ähnlich zu SUBSETSUM, also ist 3-SUM bestimmt auch \mathcal{NP} -vollständig.“

Ist Alice's Behauptung wahr?

Zeigen Sie die \mathcal{NP} -Vollständigkeit mit SUBSETSUM als Reduktionspartner oder geben Sie einen effizienten deterministischen Algorithmus an, der 3-SUM entscheidet.

↑↑↑ _____ / 7 Punkt(e) ↑↑↑



Luca geht in die erste Klasse. Er hat ein Freundebuch, in das sich alle seine Klassenmitglieder eintragen sollen. Dazu soll das Freundebuch herumgereicht werden, sodass jedes Klassenmitglied das Freundebuch genau ein Mal erhält und es am Ende wieder bei Luca ist. Leider gibt es in der Klasse Konflikte. Daher darf jedes Klassenmitglied das Freundebuch nur an ein Klassenmitglied weiterreichen, mit dem es keinen Konflikt hat.

Luca bezeichnet die Klassenmitglieder als K_1, \dots, K_n . Luca selbst ist K_1 . Nun erstellt er eine Konfliktliste L bestehend aus zweielementigen Mengen $\{K_i, K_j\}$ von Klassenmitgliedern, die einen Konflikt miteinander haben.

Die Sprache FB sei wie folgt definiert:

$$FB = \left\{ (n, L) \left| \begin{array}{l} \text{Bei } n \text{ Klassenmitgliedern mit Konfliktliste } L \text{ gibt es eine Reihenfolge,} \\ \text{angefangen und beendet bei Klassenmitglied } K_1, \\ \text{in der jedes } K_i \text{ das Freundebuch genau ein Mal erhält und} \\ \text{eine Übergabe von } K_i \text{ an } K_j \text{ nur erlaubt ist, wenn } \{K_i, K_j\} \notin L. \end{array} \right. \right\}$$

- a) Gehört $(7, \{\{K_1, K_2\}, \{K_1, K_3\}, \{K_4, K_6\}\})$ zur Sprache FB? Begründen Sie Ihre Antwort.

↑↑↑ _____ / 2 Punkt(e) ↑↑↑

- b) Zeigen Sie, dass die Sprache FB \mathcal{NP} -vollständig ist.

Hinweis: Das Hamiltonische Kreis-Problem HC ist \mathcal{NP} -vollständig.

↑↑↑ _____ / 9 Punkt(e) ↑↑↑



Ein VERTEXCOVER in einem ungerichteten Graphen $G = (V, E)$ ist eine Teilmenge C von Knoten, sodass jede Kante einen Endpunkt in der Menge C besitzt. Das Ziel ist, die Anzahl der Knoten in C zu minimieren.

Gegeben sei ein ungerichteter Graph $G = (V, E)$ mit $n = |V|$ Knoten und $m = |E|$ Kanten, wobei $V = \{1, \dots, n\}$. Beachten Sie, dass ungerichtete Graphen keine Eigenkanten haben.

Betrachten Sie nun den folgenden Algorithmus.

```
guter-greedy {
   $C_g = \emptyset$ ;
  Solange  $E \neq \emptyset$  {
    Wähle eine beliebige Kante  $\{u, v\} \in E$ ;
    Füge Knoten  $u$  und  $v$  zu  $C_g$  hinzu;
    Entferne alle zu  $u$  und  $v$  inzidenten Kanten aus  $E$ ;
  };
  Ausgabe:  $C_g$ ;
};
```

- a) Zeigen Sie, dass die Ausgabe C_g von Algorithmus guter-greedy immer höchstens 2-mal größer als eine optimale Lösung ist.

↑↑↑ _____ / 4 Punkt(e) ↑↑↑



Aufgabe 10: VertexCover (Fortsetzung)

- b) Bob hat einen Verbesserungsvorschlag für den Algorithmus `guter-greedy`: Statt beide Knoten u, v einer Kante zu C_g hinzuzufügen, kann man auch nur einen der beiden Knoten hinzufügen. Er schlägt vor immer den Knoten mit dem kleineren Index auszuwählen. Daraus ergibt sich der folgende Algorithmus:

```
schlechter-greedy {
   $C_s = \emptyset$ ;
  Solange  $E \neq \emptyset$  {
    Wähle eine beliebige Kante  $\{u, v\} \in E$ ;
    Finde den Knoten mit dem kleineren Index  $w = \min(u, v)$ ;
    Füge Knoten  $w$  zu  $C_s$  hinzu;
    Entferne alle zu  $w$  inzidenten Kanten aus  $E$ ;
  };
  Ausgabe:  $C_s$ ;
};
```

Zeigen Sie, dass der Algorithmus `schlechter-greedy` eine Ausgabe C_s produzieren kann, die $(n - 1)$ -mal größer ist als die optimale Lösung. Der Approximationsfaktor $n - 1$ soll für eine beliebige Anzahl an Knoten $n \geq 3$ gelten.

↑↑↑ _____ / 7 Punkt(e) ↑↑↑

- c) Zeigen Sie, dass für Ausgabe C_s von `schlechter-greedy` aus Aufgabenteil b) gilt: $|C_s| \leq n - 1$. Folgern Sie, dass C_s garantiert höchstens $(n - 1)$ -mal größer ist als die optimale Lösung. Sie dürfen annehmen, dass $E \neq \emptyset$.

↑↑↑ _____ / 5 Punkt(e) ↑↑↑



Wichtig: Lösungen auf dieser Seite werden nur dann berücksichtigt, wenn bei der entsprechenden Aufgabe ein Hinweis auf Seite 15 platziert wurde.

