

Einführung in die Theoretische Informatik
Klausur — SoSe 2025 — 7.8.2025

Haupttermin

Gruppe: Trajan/Claudius

Unbedingt ausfüllen

Matrikelnummer	Studiengang/Abschluss	Fachsemester
<input style="width: 100%;" type="text"/>	<input style="width: 100%;" type="text"/>	<input style="width: 100%;" type="text"/>
Nachname	Vorname	
<input style="width: 100%;" type="text"/>	<input style="width: 100%;" type="text"/>	
Unterschrift	Identifikator <small>(Beliebiges Wort zur Identifikation im anonymen Notenaushang)</small>	
<input style="width: 100%;" type="text"/>	<input style="width: 100%;" type="text"/>	

Grundregeln

- Die Bearbeitungszeit der Klausur beträgt **120 Minuten**.
- Sie schreiben diese Klausur **vorbehaltlich** der Erfüllung der **Zulassungsvoraussetzung**. Das heißt: Wir werden Ihre Zulassung vor Korrektur prüfen; die Tatsache, dass Sie die Klausur mitschreiben, bedeutet keine implizite Zulassung.
- Es sind **keine Unterlagen** und auch **keine** anderen **Hilfsmittel** erlaubt.
- Benutzen Sie nur dokumentenechten (blauen oder zur Not schwarzen) **Kugelschreiber!** Bleistiftlösungen werden nicht gewertet!
- Es zählt die Antwort, die sich im dafür vorgesehenen Kästchen befindet! Soll eine andere Antwort gewertet werden, so ist diese **eindeutig** zu kennzeichnen!
- Jegliches Schummeln, und auch der Versuch desselben, führt zum Ausschluss von der Klausur und einer Bewertung mit **5,0**.

Wird vom Korrektor/Prüfer ausgefüllt

Aufgabe	1	2	3	4	5	6	7	Σ
Punkte (max)	8	8	10	6	18	10	12	72
Punkte (erreicht)								

Punkte	0.. 35	36..37	38..39	40..41	42..44	45..47	48..50	51..53	54..57	58..60	61..72
Note	5,0	4,0	3,7	3,3	3,0	2,7	2,3	2,0	1,7	1,3	1,0

Note:

Aufgabe 1: Information – Caesar Chiffre**(8 Punkte)**

Sei $\Sigma = \{\sigma_0, \dots, \sigma_{n-1}\}$ ein Alphabet, in dem jedes Symbol einen Index hat. Bei der *Caesar Chiffre* $\mathcal{C}(\sigma_i) = \sigma_{(i+1) \bmod n}$ wird jedes Symbol durch sein (zyklisches) Nachfolgersymbol im Alphabet ersetzt. Für eine Quelle (Σ, p) ist $(\Sigma, p^{\mathcal{C}})$ die gleiche Quelle, aber jedes ursprüngliche Symbol ist mit der Caesar Chiffre \mathcal{C} verschlüsselt.

Beispiel: Betrachte das (natürlich sortierte) Alphabet der Kleinbuchstaben $\Sigma = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots, \mathbf{z}\}$. Dann ist $\mathcal{C}(\mathbf{a}) = \mathbf{b}$ und $\mathcal{C}(\mathbf{z}) = \mathbf{a}$.

(a) Erwarteter Informationsgewinn**(4 Punkte)**

Zeigen Sie mathematisch, dass für die Quellen (Σ, p) und $(\Sigma, p^{\mathcal{C}})$ der erwartete Informationsgewinn gleich ist.

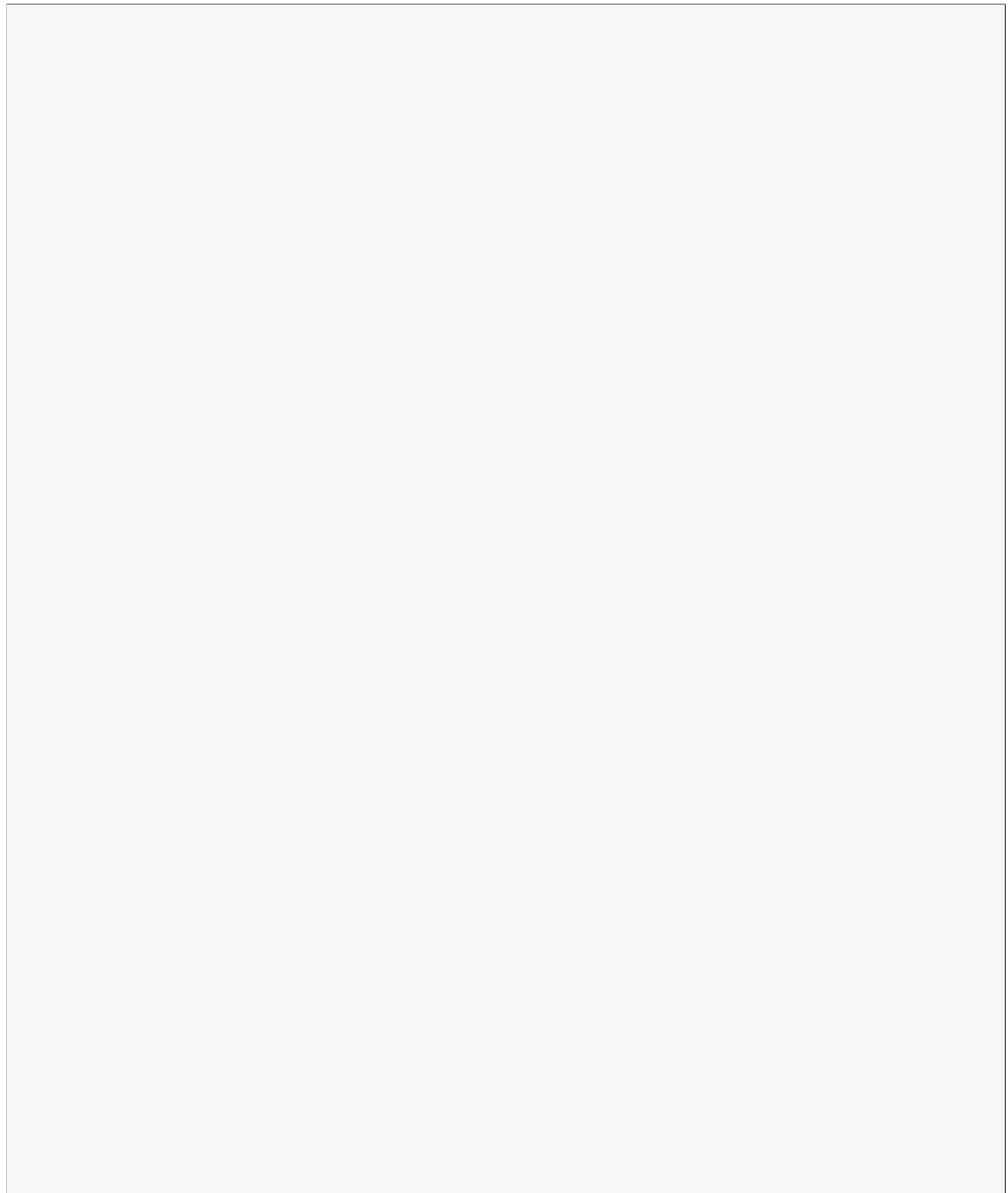
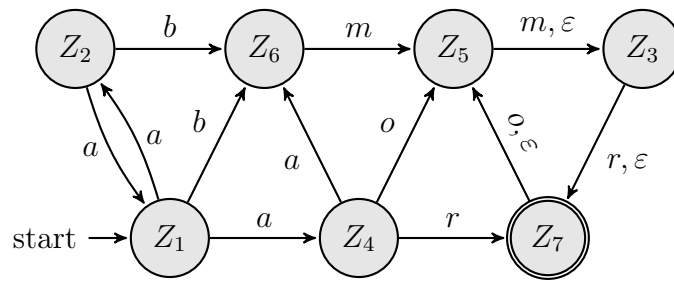
(b) Präfixcode**(4 Punkte)**

Sei \mathbb{O} ein optimaler Präfixcode für die Quelle (Σ, p) . Zeigen Sie mithilfe eines Beispiels, dass \mathbb{O} für die Quelle $(\Sigma, p^{\mathcal{C}})$ nicht notwendigerweise optimal ist.

Aufgabe 2: Umwandlung NDEA \rightarrow DEA

(8 Punkte)

Wandeln Sie den folgenden nichtdeterministischen endlichen Automaten – gemäß dem Vorgehen aus der Vorlesung – in einen deterministischen endlichen Automaten um.



Aufgabe 3: Rechnende Turingmaschine – Add Two Brute**(10 Punkte)**

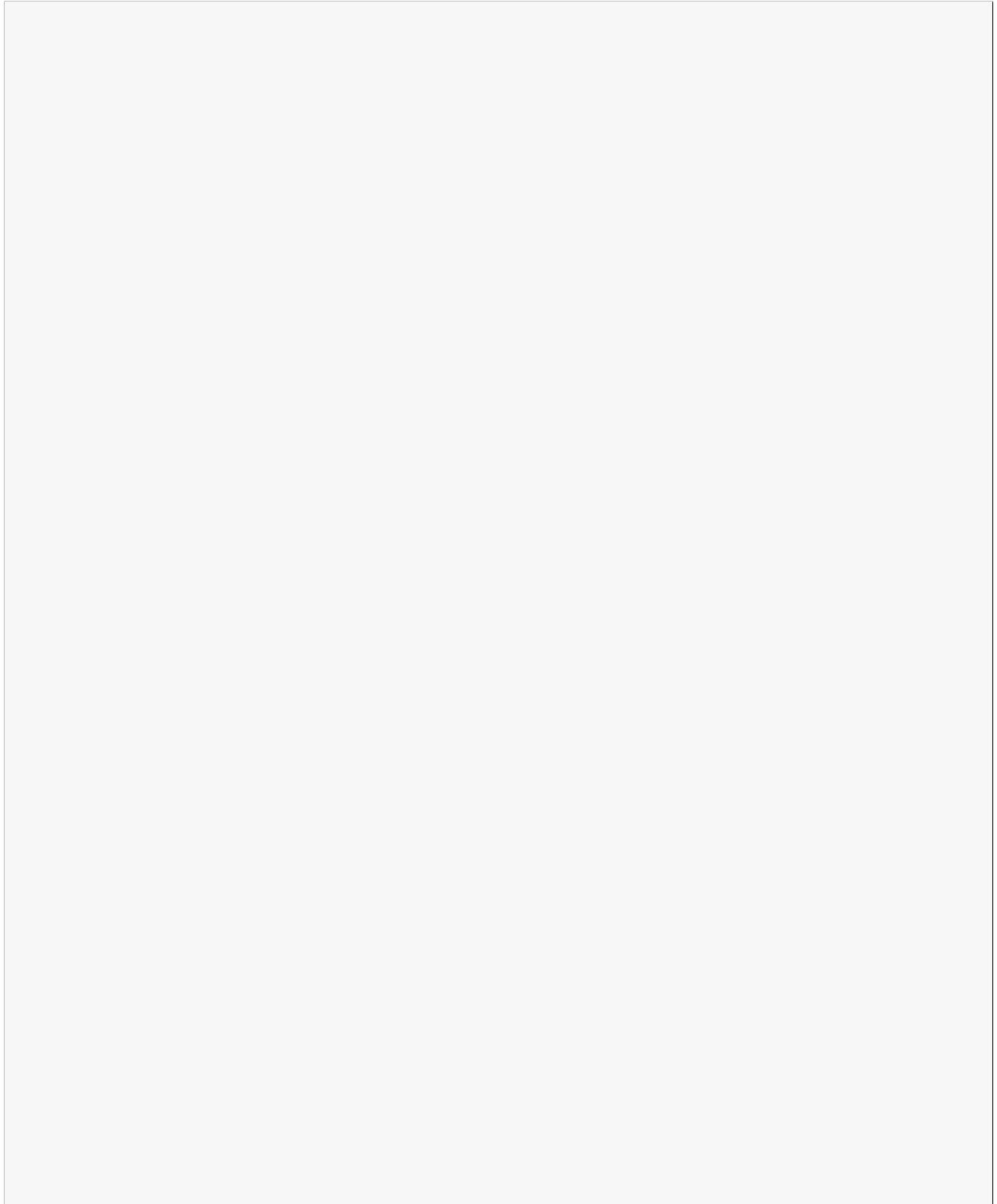
Sei $\mathbb{R}(\mu) = \mathbb{V}^{\lfloor \mu/5 \rfloor} \mathbb{I}^{\mu \bmod 5}$ eine vereinfachte römische Schreibweise einer Zahl $\mu \in \mathbb{N}$.

Sei $\mathbb{Q}(\mu) \in \{0, 1, 2, 3, 4\}^+$ die Darstellung einer Zahl im Quinärsystem (also zur Basis 5).

Beispiel: Für $\mu = 17$ ist $\mathbb{R}(\mu) = \mathbb{VVVII}$ und $\mathbb{Q}(\mu) = 32$.

Gegeben die Funktion $b(\mu) := \begin{cases} \mu + 2 & \text{falls } \mu \text{ ein Vielfaches von } 5 \text{ ist,} \\ \text{undef} & \text{sonst.} \end{cases}$

Geben Sie eine Turingmaschine mit höchstens 10 Zuständen an, die aus der Eingabe $\mathbb{R}(\mu)$, für $\mu \in \mathbb{N}$, die Ausgabe $\mathbb{Q}(b(\mu))$ berechnet. (*Lösungen mit mehr als 12 Zuständen geben 0 Punkte.*)



Aufgabe 4: Co-Semi-Entscheidbarkeit

(6 Punkte)

Eine rechnende Turingmaschine \mathcal{R} mit einer Eingabe w heißt *hinterhältig*, wenn \mathcal{R} für w mit der Ausgabe „brutus“ terminiert.

CAESAR

Gegeben: Rechnende Turingmaschine \mathcal{R} .

Gefragt: Ist \mathcal{R} mit keiner Eingabe *hinterhältig*?

Ist CAESAR co-semi-entscheidbar? Begründen Sie (ggf. mit Pseudocode).

Aufgabe 5: NP-Vollständigkeit**(18 Punkte)****(a) Lückentext:** Vervollständigen Sie.**(6 Punkte)**

Eine Reduktion von auf ist eine Funktion f ,
 die eine \mathcal{Y} -Instanz $f(I)$ aus einer beliebigen \mathcal{X} -Instanz I erzeugt, sodass gilt:
 $f(I)$ ist eine Nein-Instanz I ist eine -Instanz.

Seien $\mathcal{A} \in \mathbf{P}$ und \mathcal{B} NP-schwer. Da in NP liegt, existiert dann eine
 Reduktion von auf . Falls
 eine solche Reduktion auch in die umgekehrte Richtung existiert, ist \mathbf{P} NP.

Unabhängig vom letzten Satz: Wenn $\mathcal{B} \notin \mathbf{NPC}$, dann ist \mathcal{B} nicht .

(b) Reduktion**(12 Punkte)**

Eines der ältesten Feste der römischen Kultur waren die *Ludi Romani* zu Ehren Jupiters, deren Kosten jedoch (tatsächlich!) durch den Senat per Dekret begrenzt werden mussten.

Problem: LUDIROMANI

Gegeben: Eine Tagesanzahl $d \in \mathbb{N}$, ein Tagesbudget $s \in \mathbb{N}$ und eine Multimenge von Attraktionen $\{(r_1, c_1), \dots, (r_n, c_n)\}$: eine Attraktion (r_i, c_i) findet in einer Region $r_i \in \{1, 2, \dots, 14\}$ statt und verursacht Kosten $c_i \in \mathbb{N}$.

Gefragt: Mehrere Attraktionen können gleichzeitig in der gleichen Region stattfinden. Ist es möglich, alle Attraktionen an d Tagen zu veranstalten, sodass

- (a) an jedem Tag das Tagesbudget s nicht überschritten wird,
- (b) an jedem Tag mindestens eine Attraktion stattfindet,
- (c) jede Attraktion an genau einem Tag stattfindet und
- (d) in keiner Region an zwei Tagen hintereinander eine Attraktion stattfindet?

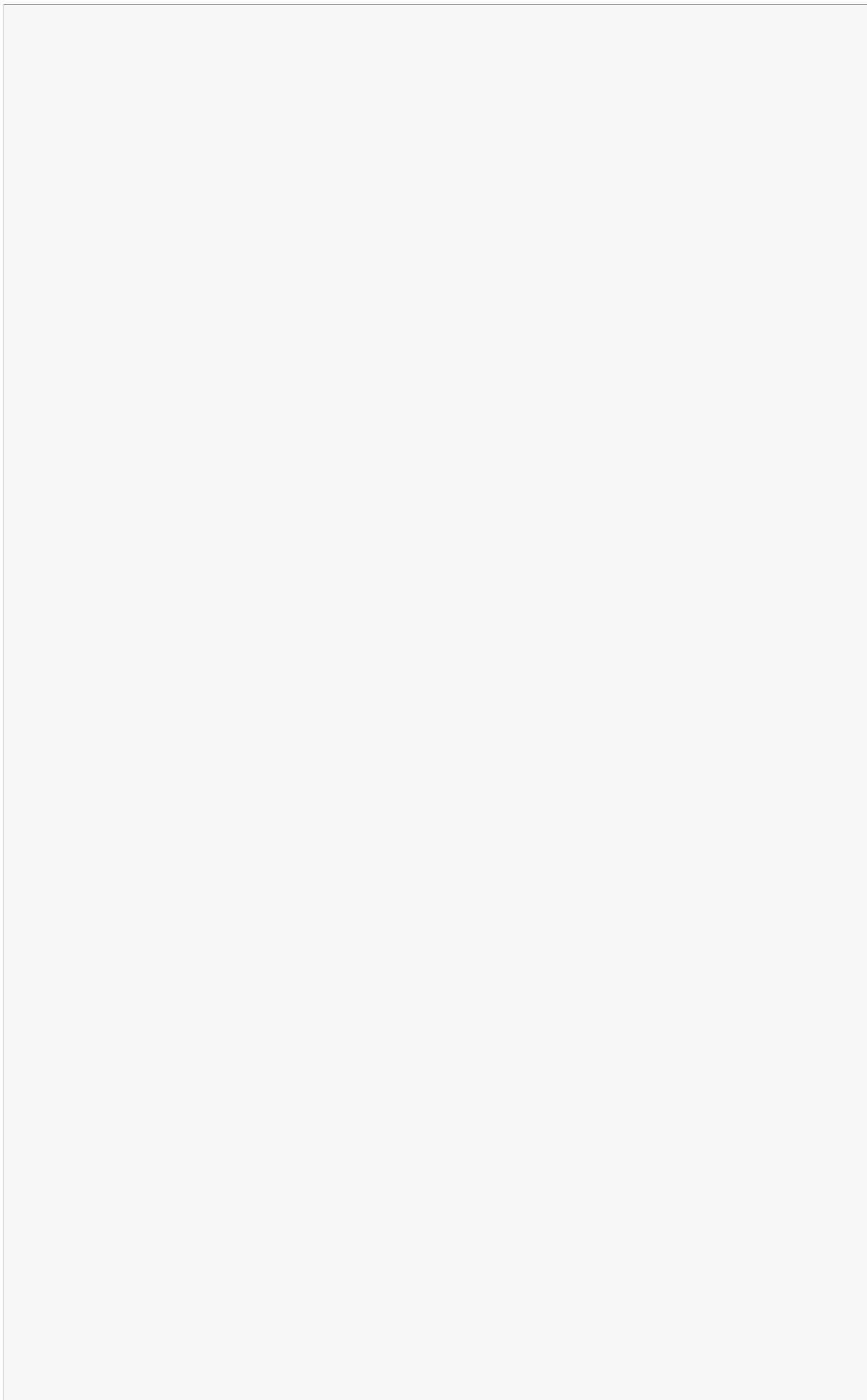
Sie sollen im Folgenden zeigen, dass LUDIROMANI NP-schwer ist. Definieren Sie hierfür SUBSETSUM, PARTITION, BINPACKING oder 3SAT.

Name:

Gegeben:

Gefragt:

Zeigen Sie nun mit Hilfe dieses Problems, dass LUDIROMANI **NP**-schwer ist.



Aufgabe 6: Fixed Parameter Tractibility

(10 Punkte)

(a) Problemarten

(4 Punkte)

Ein Kreis C in einem Graphen ist *römisch*, wenn es einen Knoten r gibt, der zu jedem Knoten von C eine Kante hat. Betrachten Sie folgendes Optimierungsproblem:

MINRÖMCYCLE

Gegeben: Ungerichteter Graph G .

Gesucht: Kürzester römischer Kreis in G .

Geben Sie das *zugehörige Entscheidungsproblem* an:

Geben Sie das *Co-Problem* zu dem Problem Ihrer obigen Antwort an:

Geben Sie das *FPT-Problem* zu MINRÖMCYCLE mit *Zielfunktionsparametrisierung* an:

(b) **Brennt Rom?**

(6 Punkte)

Nachdem Nero Rom abgefackelt hat, soll besser erkannt werden können, ob Rom brennt. Deswegen sollen f Freiwillige so auf Straßen positioniert werden, dass egal in welcher Straße ein Feuer ausbricht, immer eine Freiwillige höchstens eine Straße weit entfernt ist und Alarm auslösen kann. In Rom treffen sich an jeder Kreuzung höchstens vier Straßen. Als Graphproblem modelliert haben wir:

BRANDSCHUTZ

Gegeben: Ungerichteter Graph G mit Maximalgrad vier, natürliche Zahl $f \in \mathbb{N}$

Gefragt: Existiert eine Kanten-Teilmenge $E' \subseteq E$ mit $|E'| \leq f$, so dass für jede Kante $\{u, v\} \in E$ eine benachbarte Kante in E' existiert, d.h. $\forall \{u, v\} \in E: \exists w \in V: \{u, w\} \in E' \vee \{v, w\} \in E'$

Zeigen Sie: BRANDSCHUTZ ist in **FPT** in Bezug auf f .

Hinweis: Sowohl Kernelization als auch ein tiefenbeschränkter Suchbaum sind gut machbar, aber eines davon ist mit weniger Schreibaufwand verbunden.

Aufgabe 7: Randomisierte Algorithmen

(12 Punkte)

(a) Definitionen

(6 Punkte)

Vervollständigen Sie die folgenden Definitionen.

Co-RP enthält alle Entscheidungsprobleme, für die ein Monte-Carlo-Algorithmus mit $\left\{ \begin{array}{l} \square \text{ im Worst Case} \\ \square \text{ im Erwartungswert} \end{array} \right\}$ polynomieller Laufzeit existiert, der

Ja-Instanzen mit Wahrscheinlichkeit $\left\{ \begin{array}{ll} \square = 1 & \square = 0 \\ \square \geq \frac{1}{2} & \square \leq \frac{1}{2} \end{array} \right\}$ und

Nein-Instanzen mit Wahrscheinlichkeit $\left\{ \begin{array}{ll} \square = 1 & \square = 0 \\ \square \geq \frac{1}{2} & \square \leq \frac{1}{2} \end{array} \right\}$ korrekt beantwortet.

Falls er nicht korrekt antwortet, antwortet er $\left\{ \begin{array}{l} \square \text{ mit dem Gegenteil} \\ \square \text{ „keine Ahnung“} \\ \square \text{ gar nicht} \end{array} \right\}$.

ZPP enthält alle Entscheidungsprobleme, für die ein Las-Vegas-Algorithmus mit $\left\{ \begin{array}{l} \square \text{ im Worst Case} \\ \square \text{ im Erwartungswert} \end{array} \right\}$ polynomieller Laufzeit existiert, der

(b) **Brutus-Im-Land-Produkt**

(6 Punkte)

Der notorische Steuerbetrüger Brutus verdient sein Geld damit, für Unternehmen Berichte so zu verändern, dass diese weniger Steuern zahlen müssen. Von ihm veränderte Berichte lassen sich daran erkennen, dass mindestens die Hälfte der vorkommenden Zahlen mit der Ziffernfolge „44“ beginnen. In nicht von ihm veränderten Berichten, ist dies nur bei höchstens $1/100$ der vorkommenden Zahlen der Fall. Aus Datenschutzgründen kann der Steuerprüfer Benfordus immer nur eine zufällige Zahl eines Berichtes anfordern (das aber auch mehrfach unabhängig für den selben Bericht).

Benfordus möchte mit $k \in \mathbb{N}$ solcher Anfragen mit einer Fehlerwahrscheinlichkeit $< 1/7$ für einen Bericht entscheiden, ob dieser von Brutus verändert wurde. Wie kann er dies erreichen, sodass k konstant und möglichst klein ist?

Hinweise:

- Es ist *nicht* hilfreich zu prüfen, ob $\geq k/2$ der abgefragten Zahlen mit „44“ beginnen.
- Denken Sie zunächst über die Wahrscheinlichkeiten für $k = 1$ nach.
- Die Fehlerw'keit muss für veränderte und auch für unveränderte Berichte gelten.
- *Vielleicht* hilft: Für eine Wahrscheinlichkeit $0 \leq p \leq 1$ gilt $(1 - p)^k \geq 1 - pk$.

Notizen: