↑ Für deinen Klausurcode ✂

Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Abteilung Algorithmik

Prof. Dr. Sándor Fekete
Ramin Kosfeld
Chek-Manh Loi

Sommersemester 2024

Klausur
Algorithmen und Datenstrukturen
07.08.2024

Nachname: .....

Vorname: .....

Matr.-Nr.: .....

Studiengang: .....

☐ Bachelor ☐ Master ☐ Andere

Klausurcode:
Dieser wird benötigt, um das Ergebnis der Klausur abzurufen.

Hinweise:

- Bitte das Deckblatt in Druckschrift vollständig ausfüllen.
- Die Klausur besteht aus 13 Blättern, bitte auf Vollständigkeit überprüfen.
- Die Bearbeitungszeit für die Klausur beträgt 120 Minuten.

- Erlaubte Hilfsmittel: keine
- Eigenes Papier ist nicht erlaubt.
- Die Rückseiten der Blätter dürfen beschrieben werden.
- Die Klausur ist mit 50 % der Punkte bestanden.
- Antworten, die *nicht* gewertet werden sollen, bitte deutlich durchstreichen. Kein Tippex verwenden!
- Mit *Bleistift* oder in *Rot* geschriebene Klausurteile können nicht gewertet werden.
- Werden mehrere Antworten gegeben, werten wir die mit der geringsten Punktzahl.
- Sämtliche Algorithmen, Datenstrukturen, Sätze und Begriffe beziehen sich, sofern nicht explizit anders angegeben, auf die in der Vorlesung vorgestellte Variante.
- Sofern nicht anders angegeben, sind alle Graphen als einfache Graphen zu verstehen.

Aufgabe	1	2	3	4	5	6	7	8	Σ
Max	18	15	12	12	10	11	10	12	100
Erreicht									

Aufgabe 1: Graphen **(7+2+3+6 Punkte)**

a) Betrachte den Graphen G aus Abbildung 1. Wende den Algorithmus von Fleury auf G an, um einen Eulerweg in G zu bestimmen. Gib den resultierenden Eulerweg als Knotenliste an.

Kommen in einem Schritt des Algorithmus mehrere Knoten in Frage, wähle denjenigen mit dem kleinsten Index. Starte bei v_0.

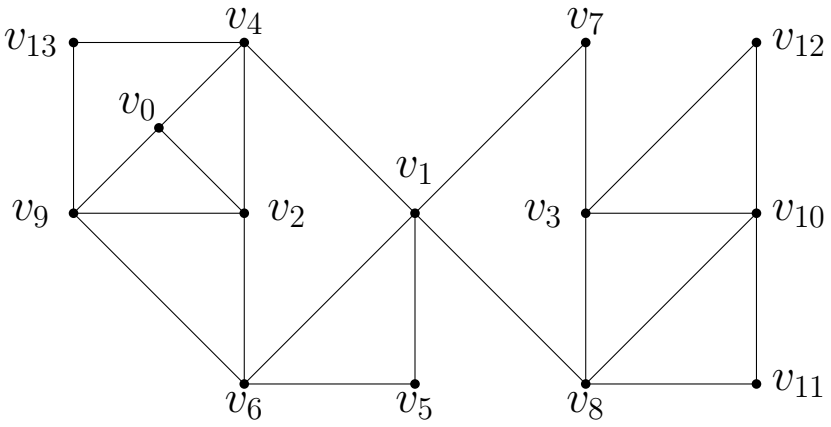


Abbildung 1: Der Graph G

b) Welche zwei Bedingungen müssen erfüllt sein, damit ein Graph **immer** eine Eulertour besitzt?

c) Sei H ein zusammenhängender einfacher Graph mit n Knoten. Zeige oder widerlege: In H gibt es nicht mehr als n verschiedene Hamiltonpfade.

d) Gib die Adjazenzliste und Adjazenzmatrix des Graphen G' aus Abbildung 2 an. Sortiere alle Einträge in der Liste *und* in der Matrix aufsteigend nach Knoten- und Kantenindizes. Du kannst deine Lösung in den dafür vorgegebenen Bereichen eintragen.

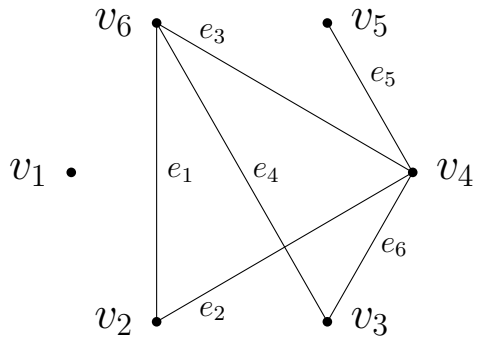


Abbildung 2: Der Graph G'

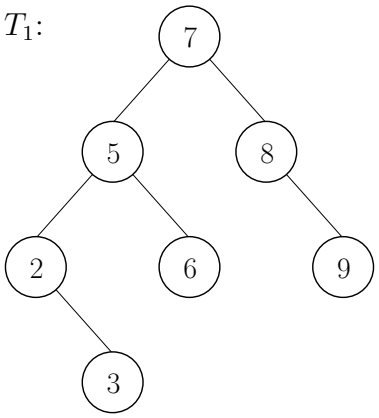
Adjazenzliste:

Adjazenzmatrix:

Aufgabe 2: Dynamische Datenstrukturen **(2+3+2+4+4 Punkte)**

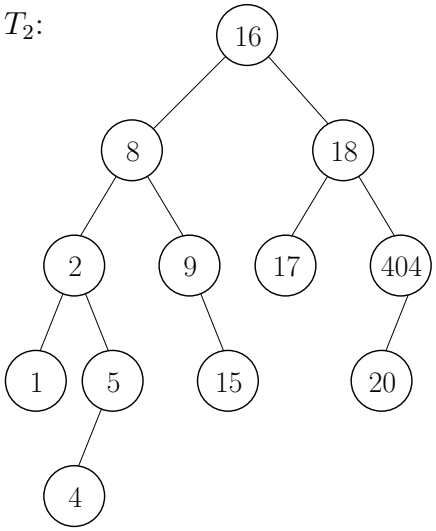
a) Führe INSERT(T_1, 4) auf dem AVL-Baum aus. Gib den Baum nach der Einfügeoperation sowie nach jeder Restructure-Operation an.
(Hinweis: Die Einfügeoperation darf in dem angegebenen Baum durchgeführt werden.)

T_1:



b) Führe DELETE(T_2, 17) auf dem AVL-Baum aus. Gib den Baum nach der Löschoperation sowie nach jeder Restructure-Operation an.

T_2:



c) Sei $M = [17, 10, 16, 7, 3, 11, 8, 6, 5, 1]$ ein Max-Heap. Gib M in Baumstruktur an.

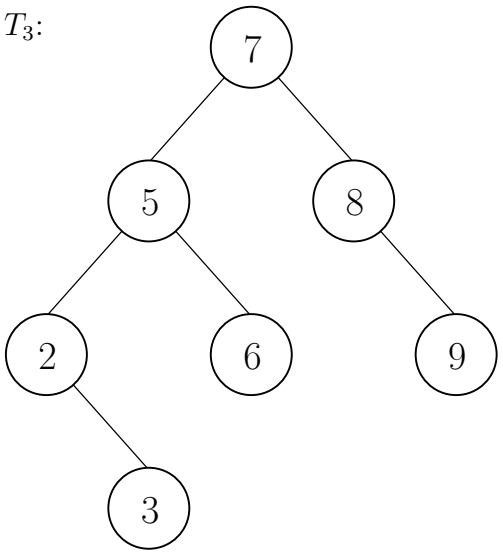
d) Sei B ein *vollständiger* binärer Baum der Höhe h. Zeige mithilfe von vollständiger Induktion, dass für die Anzahl der Knoten n in B gilt: $n = 2^h - 1$.

e) Algorithmus 1 gibt einen binären Baum in preorder aus. Wende diesen Algorithmus auf den binären Baum T_3 an. Gib dabei die Reihenfolge an, in der die Knoten ausgegeben werden, wenn PreOrder mit dem Wurzelknoten von T_3 ausgeführt wird.

```
function PreOrder(v)
    if v ≠ Nil then
        Gib S[v] aus
        PreOrder(l[v])
        PreOrder(r[v])
```

Algorithmus 1: Preorder Ausgabe eines binären Baumes.

T_3:



Aufgabe 3: Wachstum von Funktionen **(4+4+4 Punkte)**

a) Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.
Zeige oder widerlege: $\mathcal{O}(f(n)) \cap \Omega(g(n)) \neq \emptyset \Rightarrow f(n) \in \Theta(g(n))$

b) Bestimme geeignete Konstanten, um zu zeigen, dass folgende Aussage korrekt ist.

$$f(n) \coloneqq 2n(n-3) + n^3 - 8 \in \mathcal{O}(n^3)$$

c) Ordne folgende Klassen nach Inklusion. Markiere außerdem identische Klassen.

(Hinweis: Schreibe $A \subset B$, wenn jedes Element aus A auch in B vorkommt. Schreibe $A = B$ für identische Klassen. Schreibe alle Klassen in einer Zeile hintereinander, also zum Beispiel $A \subset B = C \subset D \subset E \dots$)

$$\mathcal{O}(n \log n), \quad \mathcal{O}(2^n), \quad \mathcal{O}(n^2), \quad \mathcal{O}(5n), \quad \mathcal{O}(\sqrt{2}), \quad \mathcal{O}\left(\frac{n}{2}\right), \quad \mathcal{O}(n^2{+}n), \quad \mathcal{O}(\log n)$$

Aufgabe 4: Rekursionen **(12 Punkte)**

Bestimme das asymptotische Wachstum der folgenden Funktionen mithilfe des in der Vorlesung vorgestellten Mastertheorems, oder begründe in einem Satz, warum man das Theorem nicht anwenden kann. Gib beim Anwenden alle im Mastertheorem auftretenden Parameter an.

a) $R(n) = 7n + 27 \cdot R\left(\frac{n}{3}\right) - 21 \log n - 5 + n^2$

b) $S(n) = 8n - 4 \cdot S\left(\frac{n}{4}\right) - 21 \log n - 6 + n^2$

c) $T(n) = 3 \cdot T\left(\frac{n}{3}\right) + 8\sqrt{n} + 3n \cdot T\left(\frac{n}{9}\right) + 4n^3$

d) $U(n) = 3 \cdot U\left(\frac{n}{3}\right) + 2\sqrt{n} + 54 \cdot U\left(\frac{n}{9}\right) + 7n^2$

e) $V(n) = 7n + 2 \cdot V\left(\frac{n}{2}\right) - 21 \log n - 5 + n^4$

Aufgabe 5: Mediane **(1+4+5 Punkte)**

a) Sei X eine Menge von paarweise verschiedenen natürliche Zahlen. Wie lautet die Definition eines Rang-k Elements $m \in X$?

b) Welche Schritte werden benötigt, um das Rang-k Element in einer Menge von n paarweise verschiedenen Zahlen in $O(n)$ Zeit zu finden? Kreuze in jeder Teilaufgabe den richtigen Schritt an.

(i) Teile die n Zahlen gleichmäßig in
- ◯ $t = \lceil n/2 \rceil$ viele Gruppen auf.
- ◯ $t = \lceil n/3 \rceil$ viele Gruppen auf.
- ◯ $t = \lceil n/5 \rceil$ viele Gruppen auf.

(ii) Suche in jeder Gruppe j $(1 \leq j \leq t)$
- ◯ das Minimum m_j.
- ◯ den Median m_j.
- ◯ das Maximum m_j.

(iii) Sei M die Menge aller m_j mit $1 \leq j \leq t$. Suche in M
- ◯ das Rang-$\lceil t/2 \rceil$ Element x.
- ◯ das Rang-$\lceil \sqrt{t} \rceil$ Element x.
- ◯ das Rang-$\lceil \log_2 t \rceil$ Element x.

(iv) Angenommen, es gibt nun $(i-1)$ Zahlen kleiner als x und es gilt $k > i$. Um das Rang-k Element zu finden, suchen wir in der Menge der Zahlen größer als x rekursiv nach dem
- ◯ Rang-$(k-i)$ Element.
- ◯ Rang-k Element.
- ◯ Rang-$(n-i)$ Element.

c) Sei $\ell \in \{1, \ldots, n\}$ und X ein unsortiertes Array von n paarweise verschiedenen Zahlen.

Zeige: Die ℓ kleinsten Zahlen aus X können in Zeit $O(n + \ell \log \ell)$ in sortierter Reihenfolge ausgegeben werden. (Hinweis: Die Laufzeit zum Finden eines Rang-k-Elementes kann ohne Beweis aus der Vorlesung übernommen werden.)

Aufgabe 6: Sortieren **(5+3+3 Punkte)**

a) Wir betrachten RADIXSORT zum Sortieren von acht in Abbildung 3 angegebenen Zahlen mit jeweils vier Ziffern. Sortiere das Array A aus Abbildung 3 mit RADIXSORT. Gib A nach jeder Iteration an; nutze dazu Abbildung 3:

123			
237			
532			
982			
635			
007			

Abbildung 3: RADIXSORT auf Array A (linke Spalte).

b) Welche Eigenschaft erfüllt ein stabiler Sortieralgorithmus? Ist QUICKSORT in der aus der Vorlesung bekannten Version stabil? Begründe deine Antwort.

c) Welche Laufzeit besitzt Bubblesort für n Zahlen im Worst-Case? Begründe deine Antwort.

Aufgabe 7: Algorithmenentwurf **(4+1+2+3 Punkte)**

a) Sei A ein Array mit n Elementen. Wir wollen bestimmen, ob die Elemente in A *absteigend* sortiert sind. Gib dafür einen geeigneten Algorithmus mit dem Namen IsDescending(A) in Pseudocode mit maximal 10 Zeilen an.

(Hinweis: Ein Korrektheitsbeweis des Algorithmus ist nicht erforderlich.)

b) Was gibt dein Algorithmus aus, wenn ein leeres Array eingegeben wird?

c) Welche Laufzeit in Abhängigkeit von n hat dein Algorithmus? Erkläre kurz, wie diese Laufzeit zustande kommt.

d) Kann es einen korrekten Algorithmus IsDescending geben, der das Problem *schneller* löst als in $\mathcal{O}(n)$? Begründe deine Antwort in einem Satz.

Aufgabe 8: Kurzfragen **(2×6 Punkte)**

Kreuze die korrekten Aussagen an. Es gibt nur Punkte für vollständig korrekt angekreuzte Teilaufgaben.

(Hinweis: In jeder Teilaufgabe ist immer mindestens eine Aussage korrekt.)

a) In einem einfachen Graphen ...

- ... gibt es maximal so viele Knoten, wie der Graph Kanten enthält. ☐
- ... gibt es weder doppelte Kanten noch Schleifen. ☐
- ... ist die Anzahl der Knoten mit ungeradem Grad immer ungerade. ☐

b) In welcher Datenstruktur kann das größte Element in $\mathcal{O}(n \log n)$ Zeit gefunden werden?

- AVL-Baum ☐
- Einfach verkettete Liste ☐
- Max-Heap ☐

c) Welche Laufzeitschranken sind korrekt?

- Vergleichsbasiertes Sortieren von n Zahlen: $\Omega(n^2)$ ☐
- Test auf Zusammenhang eines Graphen: $\mathcal{O}(n + m)$ ☐
- Finden einer Eulertour in einem Graphen: $\Omega(m)$ ☐

d) Welche der folgenden Sortieralgorithmen besitzen eine Laufzeit von $\mathcal{O}(n \log n)$?

- Mergesort ☐
- Quicksort ☐
- Bubblesort ☐

e) Wird eine Subroutine, welche die Laufzeit $\mathcal{O}(n^2)$ besitzt, $\mathcal{O}(n)$ mal wiederholt, so ist die asymptotische Gesamtlaufzeit der wiederholten Durchführung der Subroutine...

- ... unbekannt, da zunächst die absolute Laufzeit bekannt sein muss. ☐
- ... $\mathcal{O}(n^3)$. ☐
- ... $\mathcal{O}(n^3 + n^2)$. ☐

f) Wir wollen nacheinander eine große Menge von Zahlen in eine Datenstruktur einfügen. Welche der folgenden Datenstrukturen ist laufzeittechnisch am besten für diese Anwendung geeignet?

- Doppelt verkettete Liste ☐
- AVL-Baum ☐
- Binärer Suchbaum ☐

Viel Erfolg ☺