Algorithmen und Datenstrukturen, SS21 Termin 1

Online, open book

Zeit: 90min
(laut Prof geschätzt 80min + 10min Reserve)

1 Positionsbasierte Liste – Mittlere Frage (ca 3min)

2 Punkte

Führen Sie auf einer leeren Positionsbasierten Liste (L) die folgenden Operationen aus. Geben Sie nach Ablauf aller Operationen die resultierende Liste an.

1. L.addFirst(8)
2. L.addLast(77)
3. p = L.addBefore(L.last(), 64)
4. L.addBefore(p, 9)
5. L.remove(L.before(L.before(p)))
6. L.addAfter(L.first(), 97)
7. L.addAfter(L.first(), 22)
8. L.addAfter(p, 78)

2 Laufzeitanalyse – Kurze Frage (ca. 2min)

2 Punkte

Bestimmen Sie die asymptotische Laufzeitkomplexität des folgenden Codes:

```
public static int example(int n) {
    int counter = 0;
    for (int i = 2; i <= n; i = i * 2) {
        for (int j = i; j <= n; j = j * 2) {
            counter++;
        }
    }
    return counter;
}
```

Single Choice:

a) $O((\log n)^2)$

b) $O(n \log n)$

c) $O(2 \log n)$

d) $O(n^2)$

e) $O(n^3)$

f) $O(n)$

g) $O(2n)$

h) $O(\log n)$

3 Problemaufgabe – Lange Frage (ca. 14min)

4 Punkte

Das Informatikstudium an der Universität Tusnicht (haha) besteht aus N Pflicht-Lehrveranstaltungen (LVen), die jeweils ein Semester lang dauern. Einige dieser LVen sind Voraussetzungen anderer dieser LVen. Hängt beispielsweise LV C von den LVen A und B ab, dann müssen LVen A und B vor LV C absolviert werden. Besteht keine Abhängigkeit zwischen LVen A und B, dann können beide in demselben Semester belegt werden. Innerhalb desselben Semesters können unbegrenzt viele LVen gleichzeitig belegt werden.

Halten Sie im Folgenden Ihre Antworten so kurz und präzise wie möglich, indem Sie sich auf in unserer (realen) LV besprochene abstrakte Datentypen, Datenstrukturen, Algorithmen und andere Konzepte beziehen. Beschreiben Sie in deutscher Sprache, wie Sie diese einsetzen und ggf. modifizieren würden, um die gewünschte Antwort zu berechnen. Schreiben Sie keinen Code, und geben Sie keine detaillierten Schritt-für-Schritt-Beschreibungen.

Gliedern Sie Ihre Antwort klar durch Zwischenüberschriften *Aufgabe 1, Aufgabe2,* usw.

1. Wie würden Sie eine Instanz dieses Problems repräsentieren, um die folgenden Fragen zu beantworten? Welche abstrakten Datentypen und Datenstrukturen würden Sie einsetzen?

2. Wie würden Sie folgendes Problem algorithmisch lösen: Ist dieses Studium studierbar, oder gibt es unerfüllbare Abhängigkeiten?

3. Wie würden Sie folgendes Problem algorithmisch lösen: Wie viele Semester werden für dieses Studium mindestens benötigt?

4. Wie würden Sie folgendes Problem algorithmisch lösen: Sie möchten das Studium in minimaler Semesterzahl abschließen, und Sie möchten alle LVen möglichst früh im Studium belegen. Wie viele LVen müssen Sie maximal innerhalb desselben Semesters absolvieren?

➔ Freitext Antwortfeld

4 Groß-O – Kurze Frage (ca. 2min)

4 Punkte

Wählen Sie die beste Groß-O-Charakterisierung der folgenden Funktionen:

- $\mathbf{0.01}n\ \log_2 n + n(\log_2 n)^2$
  - $O(n(\log n)^2)$
  - $O(n(\log_2 n)^2)$
  - $O(n \log n)$
  - $O((\log_2 n)^2)$

- $(n+1)^5$
  - $O(n^4)$
  - $O(n^5)$
  - $O(5n^5)$
  - $O(n^5 + 5)$

- 2^{n+1}
  - $O(2^n)$
  - $O\left(2^{\log n}\right)$
  - $O(n^2)$
  - $O(2^{n+1})$

- $\sum_{i=1}^{n} \log i$
  - $O(n)$
  - $O(\log n)$
  - $O(n \log n)$
  - $O(n^2)$

- $\sum_{i=1}^{n} i^2$
  - $O(n^3)$
  - $O(n^2)$
  - $O(n)$
  - $O(n \log n)$

- $2n + 100 \log n$
  - $O(\log n)$
  - $O(n)$
  - $O(2n)$
  - $O(n \log n)$

- $5n^4 + 3n^3$
  - $O(n^3)$
  - $O(5n^4)$
  - $O(3n^3)$
  - $O(n^4)$

- $5n^2 + 3n \log n$
  - $O(5n^2)$
  - $O(n^2)$
  - $O(n^2 \log n)$
  - $O(n \log n)$

5 Suchbäume – Mittlere Frage (ca. 3min)

2,5 Punkte

Welche der folgenden Aussage sind wahr? (Multiple Choice)

a) Die Sentinel-Nodes eines Rot-Schwarz-Baumes sind immer schwarz

b) Ein binärer Suchbaum mit n Elementen hat immer genau n + 1 externe Knoten

c) Jeder AVL-Baum ist auch ein binärer Suchbaum

d) Rot-Schwarz-Bäume erlauben eine asymptotisch effizientere Einfügung als AVL-Bäume, wenn man die Anzahl der Rechenoperationen betrachtet

e) Das Entfernen aus einem Rot-Schwarz-Baum benötigt im schlechtesten Fall O(n) Zeit, wobei der Baum vor dieser Entfernung n Elemente enthielt

6 Mergesort – Mittlere Frage (ca. 4min)

2 Punkte

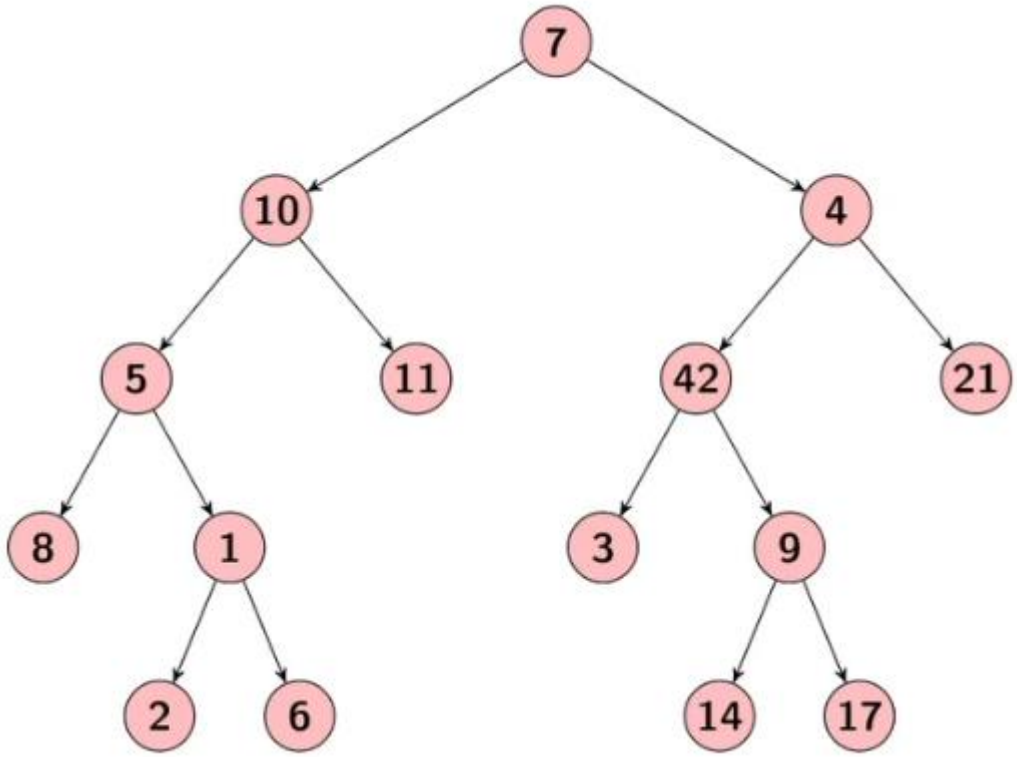
Gegeben ist folgende unsortierte Liste. Sortieren Sie diese aufsteigend mit Mergesort (Variante aus der Vorlesung). Wie viele Vergleichsoperationen von Listenelementen werden dabei benötigt?

37, 34, 44, 40, 86, 66, 80, 46

7 Traversierung Bäume I – Kurze Frage (ca. 2min)

3 Punkte

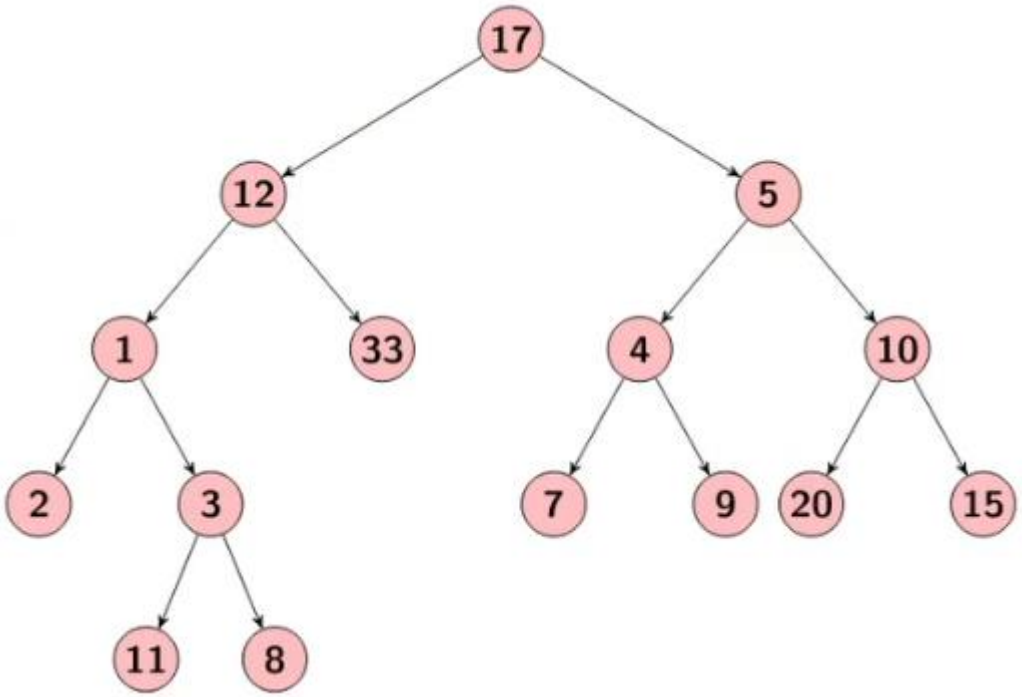
In welcher Reihenfolge werden die Knoten im folgenden Baum bei einer **Präorder** Traversierung besucht?



8 Traversierung Bäume II – Kurze Frage (ca. 2min)

3 Punkte

In welcher Reihenfolge werden die Knoten im folgenden Baum bei einer **Postorder** Traversierung besucht?



9 Heap – Mittlere Frage (ca. 3min)

2 Punkte

Gegeben ist folgende Liste von Elementen. Verwenden Sie Bottom-up Heap Construction um daraus einen Heap zu erstellen und geben Sie dessen Arraydarstellung an.

93, 64, 56, 72, 82, 22, 14

10 Iterator – Kurze Frage (ca. 2min)

1 Punkt

Gegeben sei die Liste [46, 64, 73, 9, 91, 47, 76] von Zahlen. Diese Liste wird als Parameter an die Methode doSomething() übergeben. Geben Sie die Liste nach Ausführung der Methode an. Wenn mehr Felder zur Verfügung stehen, als die Liste am Ende lang ist, dann füllen Sie die leeren Felder mit `-` (Minus ohne Anführungszeichen).

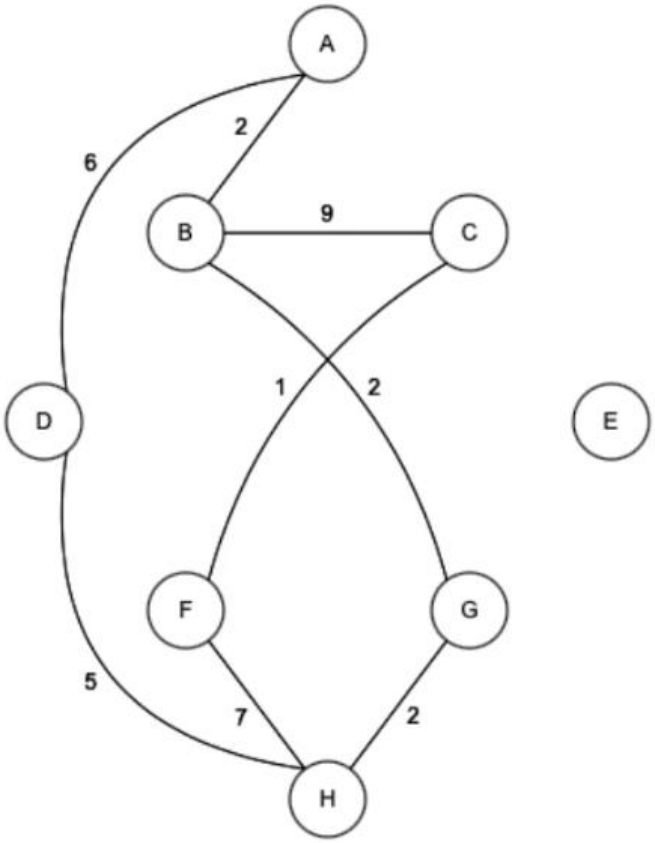
```
doSomething(numbers) {
    numbersIt = numbers.iterator();

    while (numbersIt.hasNext()) {
        if ((numbersIt.next() % 2) == 1) {
            numbersIt.remove();
        }
    }
}
```

(Es waren Felder von a – g gegeben mit je einem Eingabefeld)

11 Dijkstra – Mittlere Frage (ca. 5min)

4 Punkte

Führen Sie den Dijkstra Algorithmus auf dem gegebenen Graphen durch. Starten Sie bei Knoten D und füllen Sie die Tabelle nach Abschluss des Algorithmus mit den finalen Werten aus. Im Fall von gleichen Distanzen (Schlüsseln), wird die Vorrangwarteschlange alphabetisch nach der Knotenbezeichnung sortiert (aufsteigend). Die Reihenfolge beginnt bei 0 mit dem Startknoten. Wenn ein Knoten nicht besucht wird geben sie ein ´-´ (Minus ohne Anführungszeichen) bei der Reihenfolge an und als Distanz ´INF´ (ohne Anführungszeichen).

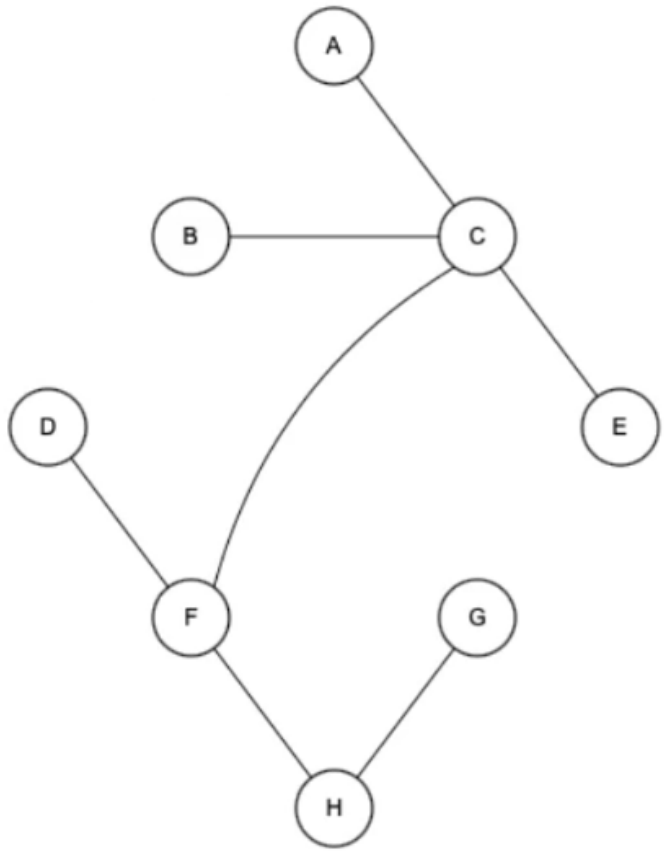


Knoten	Reihenfolge	Distanz
A		
B		
C		
D		
E		
F		
G		
H		

12 Traversierung Graphen – Kurze Frage (ca. 2min)

4 Punkte

In welcher Reihenfolge (beginnend mit 0 für den Startknoten) werden die Knoten im folgenden Graphen bei einer **Breitentraversierung** beginnend mit Knoten **C** besucht? Geben Sie außerdem das Niveau der Besuchten Knoten an. Wenn zwischen mehreren Folgeknoten entschieden werden muss, wählen Sie diese in alphabetischer Reihenfolge (aufsteigend). Falls nicht alle Knoten besucht werden, füllen Sie die Tabelle mit ´-´ (Minus ohne Anführungszeichen) auf.



Reihenfolge	Knoten	Niveau
1		
2		
3		
4		
5		
6		
7		
8		

13 AVL Baum – Mittlere Frage (ca. 6min)

4 Punkte

beginnen Sie mit einem leeren AVL-Baum und führen Sie folgende Operationen aus. Vergleichen Sie lexikografisch (alphabetisch). extttaaa wäre dabei z.B. kleiner als extttaaz. Halten Sie sich strikt an die Algorithmen aus der Vorlesung. Zählen Sie dabei die Anzahl der Operationen, die eine Restrukturierung erfordern.

1. insert(„aps“)
2. insert(„fqk“)
3. insert(„nsb“)
4. insert(„fqe“)
5. insert(„tot“)
6. remove(„fqk“)
7. insert(„jhf“)
8. remove(„fqe“)
9. insert(„rsh“)
10. insert(„fqk“)
11. insert(„epw“)
12. insert(„mnu“)

- Was ist die Höhe des resultierenden Baumes?
- Bei wie vielen Operationen ist eine Restrukturierung nötig?
- Geben Sie die Präorder Traversierung des resultierenden Baumes ohne leere Blätter, mit Beistrich (Komma) getrennt und ohne Anführungszeichen oder Leerzeichen an.

14 Hashing – Lange Frage (ca. 8min)

4 Punkte

Führen Sie die folgenden Operationen auf einer mittels Array implementierten Menge mit der primären (komprimierenden) Hashfunktion $\boldsymbol{c(h(i)) = 3i \bmod 7}$ durch. Die Kollisionsbehandlung findet mit doppeltem Hashing statt und verwendet die sekundäre Hashfunktion**:** $\boldsymbol{h'(i) = 5 - (i \bmod 5)}$. Bilden Sie die gegebenen Werte auf eine Zuordnungstabelle der Größe 7 ab.

1. add(26)
2. add(62)
3. add(48)
4. add(73)
5. add(5)
6. add(17)
7. add(4)

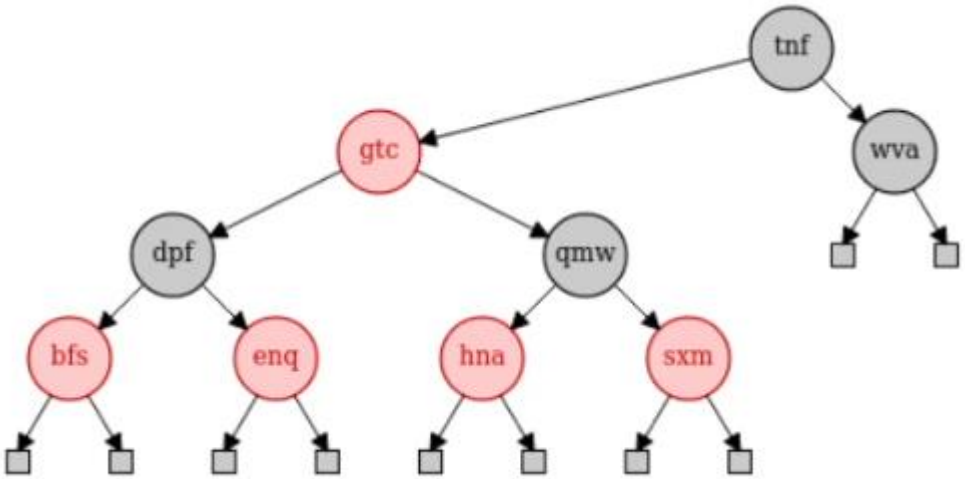
Geben Sie das resultierende Array an.

15 Rot-Schwarz-Bäume – Lange Frage (ca. 17min)

6,5 Punkte

Fügen Sie die folgenden Werte in den ebenfalls gegebenen Rot-Schwarz-Baum ein. Vergleichen Sie lexikografisch (alphabetisch. aaa wäre dabei z.B. kleiner als aaz. Halten Sie sich strikt an die Algorithmen aus der Vorlesung. Zählen Sie dabei die Anzahl der Einfügeoperationen, die eine Restrukturierung erfordern.

fks, oxm, fun, ulh, qed



- Handelt es sich bei dem Ausgangsbaum um einen korrekten Rot-Schwarz-Baum?
- Was ist die Höhe des resultierenden Baumes?
- Bei wie vielen Einfügeoperationen ist eine Restrukturierung (keine reinen Umfärbungen) nötig?
- Geben Sie die Präorder Traversierung des Resultierenden Baumes ohne leere Blätter, mit Beistrich (Komma) getrennt und ohne Leerzeichen an.

16 Huffman – Mittlere Frage (ca. 5min)

4 Punkte

Erstellen Sie den **Huffman-Baum** für die gegebenen Häufigkeiten. Beim Zusammenfügen sollte das **rechte Kind** stets der Teilbaum mit der **größeren Häufigkeit** sein.

- M: 1
- U: 2
- I: 4
- F: 5
- S: 6
- T: 8

Geben Sie die Präorder-Traversierung der Häufigkeiten im Huffman-Baum an (Bsp. a:4, b:2, c:3 -> 9,4,5,2,3):

Codieren Sie das Wort STIFT mit dem Huffman-Baum: