Vorlesungsprüfung

703010 VO Algorithmen und Daten Strukturen 2019

16. Januar 2020

Vorname: _____ Nachname: _____

Matr. Nr.: _____ Note: _____

Diese Klausur besteht aus **??** Aufgaben und Sie können maximal 50 Punkte erreichen. Es sind keine elektrotechnischen Geräte oder andere Hilfsmittel zulässig. Sie haben 90 Minuten Zeit und müssen *alle* Zettel nach Abschluss der Klausur abgeben. Wenn nötig, benennen Sie Ihre Annahmen. Geben Sie präzise und knappe Antworten.

This exam consists of **??** questions with a total of 50 points. The use of electronic devices is not allowed. The exam is closed book and closed notes. The total duration of this exam is 90 minutes. Participants have to return this copy and *all* additional sheets at the end of the exam. Specify your own assumptions if needed, and provide precise and concise answers.

1 Pseudocode und Komplexität (10 Punkte)

1. **[5 Punkte]** Folgendes Codefragment ist gegeben:
   Consider the following code fragment:

```
int start(int N) {
    return inner(0, N, 1);
}

int inner(int s, int N, int k) {
   if (2*k >= N)
     return s;
   else
     inner(s+k, N, 2*k);
}
```

   Welcher Groß-O Komplexitätsklasse gehört die Anzahl der Funktionsaufrufe dieses Codefragments bezüglich der Problemgröße N an? Begründen Sie ihre Lösung.
   State the Big-Oh complexity class of the number of function calls in this code fragment in terms of the problem size N. Justify your answer.

2. **[5 Punkte]** Das folgende Polynom is gegeben:
   Consider the following polynomial:

$$3 \log n + 2n \cos n^2 + \frac{1}{n}$$

   Welcher Groß-O Komplexitätsklasse gehört es bezüglich der Problemgröße n an? Beweisen Sie ihre Lösung.
   State its Big-Oh complexity class in terms of the problem size n. Prove your answer.

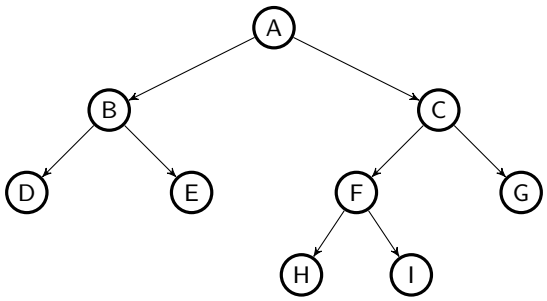
2 Tree Construction (10 Punkte)

Fügen Sie die folgenen Zahlen schrittweise in einen anfangs leeren Rot-Schwarz-Baum [**5 Punkte**] bzw. einen Heap [**5 Punkte**] ein. Geben Sie die resultierende Datenstruktur nach jeder eingefügten Zahl an. (Nutzen Sie den Platz links gegenüber.)

Starting from an empty AVL tree and heap, add the following numbers. Show the resulting data structure after every insertion. (Use the space on the facing page.)

5, 3, 4, 2, 1

3 Tree Traversal (10 Punkte)



Geben Sie die Reihenfolge an, in der die Knoten das Baums für folgende Traversierungsarten ausgegeben werden:

Write down the order in which the nodes are returned using the following tree traversal strategies:

1. **Präorder [2 Punkte]**

2. **Inorder [2 Punkte]**

3. **Postorder [2 Punkte]**

4. **Euler Traversal [4 Punkte]**

   Geben Sie für den Prä-Besuch eine öffnende Klammer "(", beim In-Besuch den gespeicherten Buchstaben und beim Post-Besuch eine schließende Klammer ")" aus.

   Return an opening parenthesis "(" at pre-visit, the label of the node at the in-visit, and a closing parenthesis ")" at the post-visit.

4 Quicksort (10 Punkte)

1. [**4 Punkte**] Sortieren Sie die folgende Sequenz mit dem Quicksort-Algorithmus in aufsteigender Reihenfolge, wobei immer das *letzte* Element einer (Teil-)Sequenz als Pivot dient. Schreiben Sie die bei der Ausführung entstehenden Teilsequenzen Schritt für Schritt untereinander auf, und stellen Sie klar, wie sie jeweils zustande kommen.

   Sort the following sequence using the Quicksort algorithm in ascending order, always using the *last* element of a (sub)sequence as the pivot. Write the resulting subsequences in one line per step of the algorithm, and indicate how they arise.

   8 3 6 1 4 7 6 5 2 5

2. [**3 Punkte**] Welchem algorithmischen Paradigma folgt Quicksort? Nennen Sie einen weiteren Sortieralgorithmus, der demselben Paradigma folgt, sowie drei weitere Sortieralgorithmen, die diesem Paradigma *nicht* folgen.

   Which algorithmic paradigm does Quicksort exemplify? Name one other sorting algorithm that follows the same paradigm, and three more sorting algorithms that do *not* follow this paradigm.

3. [**3 Punkte**] Was sind die *erwartete* und die im *schlechtesten* Fall erreichte Laufzeitkomplexitäten von Quicksort? Sind diese nicht identisch, geben Sie eine Beispielsequenz an, für die der obige Sortiervorgang außerhalb der erwarteten Laufzeitkomplexität von Quicksort liegt.

   What are the *expected* and *worst-case* running-time complexities of Quicksort? If they differ, give an example of a sequence that is outside the expected running-time complexity class if sorted according to the above procedure.

5 Stichwortverzeichnis (10 Punkte)

Spezifizieren Sie einen effizienten Algorithmus zur Erstellung eines Stichwortverzeichnis L für ein Dokument D. Hier, D ist ein Array von N (nicht unbedingt verschiedenen) Wörtern, die (sequenziell gelesen) den Text darstellen; L ist eine alphabetisch sortierte Liste der Länge n, die jedes der n verschiedenen Wörter $w \in D$ enthält, jeweils mit einer Liste der Indizes von w im Array D. **[6 Punkte]**

Ermitteln Sie die asymptotische Laufzeitkomplexität Ihres Algorithmus **[4 Punkte]**.

Sie müssen als Antwort nicht unbedingt Pseudocode formulieren. Ihre Erklärung des Algorithmus muss aber ausreichend detailliert sein, um Ihre Antworten auf die gestellten Fragen hinreichend zu untermauern. Beziehen Sie sich so weit wie möglich auf in der Lehrveranstaltung besprochene ADTs, Datenstrukturen und/oder Algorithmen. Schreiben Sie *keinen* C-, Java- oder ähnlichen Code.

Specify an efficient algorithm that creates an index L for a document D. Here, D is an array of N (not necessarily distinct) words that, read in sequence, constitute the text; L is an alphabetically-sorted list of length n that contains each of the n distinct words $w \in D$, each along with a list of the indices of w in the array D.

Determine the asymptotic running-time complexity of your algorithm.

You do not necessarily need to provide pseudo-code, but you do need to specify your algorithm with sufficient detail to justify your answers. To the extent possible, refer to ADTs, data structures and/or algorithms discussed in class as appropriate. Do *not* write C, Java or similar code.