

**Aufgabe 1 [2.0]**

- a. Gegeben ist folgendes rekursive Programm:

```
int f(n) {
    int s = 0;
    if(n<=1) return 1;
    else for(int i=0; i<4; i++) s = s + f(n/4);
    return s;
}
```

[0.2] Entwickeln Sie die *Rekurrenzgleichung*, die das Laufzeitverhalten der Funktion  $f(n)$  beschreibt.

[0.5] Bestimmen Sie das *asymptotische Laufzeitverhalten* der Funktion  $f(n)$  mit Hilfe des Master Theorems.

- b. [0.4] Definieren Sie die Begriffe  $\Theta(n)$  und  $\Omega(n)$  bei der Laufzeitabschätzung eines Programms?
- c. [0.4] Was verstehen Sie unter dem Begriff der *Algorithmischen Lücke*? Geben Sie jeweils ein Beispiel für eine Problem an, wo die Algorithmische Lücke geschlossen bzw. offen ist.
- d. [0.5] Skizzieren Sie den Programmgraphen des obigen Programms und berechnen Sie dessen zyklomatische Komplexität nach der Metrik von McCabe.

**Aufgabe 2 [2.0]**

- a. [1.0] Fügen Sie die Werte 5, 8, 3, 4, 6, 7, 1, 2, 9, 11, 10 in dieser Reihenfolge in einen ursprünglich leeren  $B+$  - Baum der Ordnung 1 ein.
- b. [0.5] Löschen Sie die Werte 5, 6 und 4 (in dieser Reihenfolge) in dem oben erstellten  $B+$  - Baum.

Geben Sie alle Zwischenschritte bei der Veränderung des Baums an.

- c. [0.5] Beschreiben Sie die beiden Situationen, die dazu führen, dass ein  $B+$ -Baum zu 50 Prozent oder zu 100 Prozent gefüllt ist.

**Aufgabe 3 [1.6]**

Die Werte 5, 7, 8, 1, 6, 3, 2, 4 sind in dieser Reihenfolge in einem Vektor der Länge 8 gespeichert.

Sortieren Sie diesen Vektor jeweils mit dem Verfahren

- a. [0.2] Bubblesort,  
b. [0.4] Quicksort und  
c. [0.4] Bucketsort.

Geben Sie alle Zwischenschritte bei den Sortievorgängen an.

- d. [0.6] Geben Sie für jeden der obigen Sortieralgorithmen eine Situation in der Praxis an, bei der es sinnvoll ist, gerade ihn einzusetzen.

**Aufgabe 4 [1.5]**

- a. [0.5] Tragen Sie die folgenden Zeichenketten (Strings) in die Datenstruktur eines Tries ein. Skizzieren Sie den schrittweisen Aufbau des digitalen Suchbaumes (Trie).

- SAMBA
- SALSA
- CCC
- TANGO
- PASO
- PASODOBLE

Stellen Sie die Struktur des Tries graphisch dar und geben Sie alle Zwischenschritte an.

- b. [0.2] Geben Sie eine geeignete Datenstruktur in C++ zur Speicherung eines Tries an.
- c. [0.8] Skizzieren Sie in Pseudocode einen Algorithmus zur Suche aller gespeicherten Strings im Trie, die mit einem gegebenen Teilstring (Praefix) beginnen.  
Dokumentieren Sie Ihr Programm ausführlich.

**Aufgabe 5 [2.0]**

- a. [0.7] Fügen Sie die Werte 8, 4, 3, 6, 7, 5, 1 und 2 in dieser Reihenfolge in einen ursprünglich leeren Heap ein (das Minimum ist in der Wurzel gespeichert). Stellen Sie den Heap als binären Baum dar.
- b. [0.3] Löschen Sie 3x das Minimum.

Geben Sie alle Zwischenschritte bei der Veränderung des Baums an.

- c. [0.6] Traversieren Sie den entstandenen Baum mit der *Preorder*, *Inorder* und *Postorder* Methode und geben Sie die entsprechenden Reihenfolgen der Werte der besuchten Knoten aus.
- d. [0.4] Geben Sie die Aufwände in O-Notation für die Heap-Operationen Einfügen, Löschen und Zugriff auf das Minimum an und leiten Sie diese informell ab.

### Aufgabe 6 [0.9]

Erläutern Sie die Algorithmenentwurfsverfahren *greedy*, *divide-and-conquer* und *dynamic programming* jeweils anhand ihrer

- zugrundeliegenden Ideen,
- ihrer Eigenschaften und
- eines Beispiels.