FAKULTÄT FÜR INFORMATIK UND MATHEMATIK
OTH Regensburg · D-93053 Regensburg
http://www.oth-regensburg.de/

Prof. Dr. Carsten Kern

Algorithmen und Datenstrukturen

– Übungsblatt 5 –

WS 2018/19

Votierung und Abgabe bis zum **05.11.2018, 23:00 Uhr**

Hinweis:
- Falls Sie bei der Bearbeitung einer Aufgabe größere Schwierigkeiten hatten und deswegen die Bearbeitung abgebrochen haben, so versuchen Sie bitte, Ihre Schwierigkeiten in Form von Fragen festzuhalten. Bringen Sie Ihre Fragen einfach zur Vorlesung oder zur Übung mit!

Aufgabe 1 (MergeSort und HeapSort)

Demonstrieren Sie die Funktionsweise von *Mergesort* und *Heapsort* anhand des Feldes:

$$\mathtt{a}[] = [\,-5, 13, -32, 7, -3, 17, 23, 12, -35, 19\,].$$

a) Führen Sie die Demonstration von *Mergesort* anhand eines Rekursionsbaums durch, indem Sie den Baum in der Phase „Teilen" von oben nach unten aufbauen und anschließend in der Phase „Mischen" von unten nach oben durchlaufen. Nennen Sie jeweils die entstandenen Teilfolgen.
b) Führen Sie bei der Demonstration von *Heapsort* sowohl entstehende Bäume als auch entstehende Felder mit.
c) Überprüfen Sie Ihre Ausführungen aus a) und b) mithilfe eines `C, C++, C#` oder `Java`-Programms, das die beiden Sortierverfahren implementiert und die wesentlichen Informationen ausgibt.

Aufgabe 2 (Implementierung eines Algorithmus)

Entwickeln Sie einen Algorithmus mit Laufzeit $\Theta(n \log n)$, der folgende Spezifikation erfüllt:

- Eingabe: $\mathtt{a}[\,] = [a_0, \ldots, a_{n-1}], s \quad$ mit $a_i \in \mathbb{Z},\ s \in \mathbb{Z},\ n \in \mathbb{N}$ (also $n+1$ ganze Zahlen)
- Ausgabe: `true`, falls es zwei Elemente a_i, a_j gibt mit $s = a_i + a_j, i \neq j$ und `false` sonst

a) Erläutern Sie zunächst die zugrundeliegende Idee.
b) Weisen Sie nach, dass Ihr Algorithmus die Laufzeit $\Theta(n \log n)$ hat.
c) Implementieren und testen Sie Ihren Algorithmus in `C, C++, Java` oder `C#`.

Aufgabe 3 (Heaps)

Zeigen Sie folgende Aussagen:

a) Ein Heap mit n Elementen hat die Höhe $\lfloor \log n \rfloor$.
b) Ein Heap mit n Elementen hat höchstens $\left\lceil \frac{n}{2^{h+1}} \right\rceil$ viele Knoten der Höhe h.

c) Für alle x mit $|x| < 1$: $\sum_{k=0}^{\infty} kx^k = \frac{x}{(1-x)^2}$.

Tipp: Differenzieren Sie beide Seiten der geometrischen Reihe und multiplizieren Sie mit x)

d) Kann die Reihenfolge der `heapify`-Aufrufe in der Operation `buildMaxHeap` ohne weitere Modifikation vertauscht werden? Begründen Sie Ihre Antwort!

Aufgabe 4 (Matrixprodukt)

Stellen Sie sich vor, Sie sollen zwei quadratische Matrizen $M, N \in R^{n \times n}$ $\quad (R:$ Ring) miteinander multiplizieren. Sei $n = 2^i$ für ein $i \in \mathbb{N}$, dann kann man M, N und $O = M \cdot N$ wie folgt zerlegen:

$$M := \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}, \quad N := \begin{pmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{pmatrix}, \quad O := \begin{pmatrix} O_{11} & O_{12} \\ O_{21} & O_{22} \end{pmatrix}$$

a) Zeigen Sie, dass die beiden folgenden Varianten die Produktmatrix O korrekt berechnen:

- Variante 1:

$$\begin{aligned} O_{11} &:= M_{11} \cdot N_{11} + M_{12} \cdot N_{21} \\ O_{12} &:= M_{11} \cdot N_{12} + M_{12} \cdot N_{22} \\ O_{21} &:= M_{21} \cdot N_{11} + M_{22} \cdot N_{21} \\ O_{22} &:= M_{21} \cdot N_{12} + M_{22} \cdot N_{22} \end{aligned}$$

- Variante 2:

$$\begin{aligned} H_1 &:= (M_{11} + M_{22}) \cdot (N_{11} + N_{22}) \\ H_2 &:= (M_{21} + M_{22}) \cdot N_{11} \\ H_3 &:= M_{11} \cdot (N_{12} - N_{22}) \\ H_4 &:= M_{22} \cdot (N_{21} - N_{11}) \\ H_5 &:= (M_{11} + M_{12}) \cdot N_{22} \\ H_6 &:= (M_{21} - M_{11}) \cdot (N_{11} + N_{12}) \\ H_7 &:= (M_{12} - M_{22}) \cdot (N_{21} + N_{22}) \end{aligned}$$

$$\begin{aligned} O_{11} &:= H_1 + H_4 - H_5 + H_7 \\ O_{12} &:= H_3 + H_5 \\ O_{21} &:= H_2 + H_4 \\ O_{22} &:= H_1 - H_2 + H_3 + H_6 \end{aligned}$$

b) Bestimmen Sie die asymptotische Laufzeitkomplexität beider Varianten und vergleichen Sie diese mit der Komplexität der Standardmethode zur Multiplikation zweier Matrizen.