Prof. Dr. Ir. Joost-Pieter Katoen

Sebastian Junges, Benjamin Kaminski, David Korzeniewski, Tim Quatmann

Übung 5

Hinweise:

- Die Lösungen müssen bis **Donnerstag, den 17. Mai um 16:00 Uhr** in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Die Übungsblätter **müssen** in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung abgegeben werden.
- Drucken Sie ggf. digital angefertigte Lösungen aus. Abgaben z.B. per Email sind nicht zulässig.
- Namen und Matrikelnummer sowie die **Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. Abgaben, die aus mehreren Blättern bestehen **müssen geheftet bzw. getackert** werden! Die **Gruppennummer muss sich auf der ersten Seite oben links** befinden.
- **Bei Nichtbeachten der obigen Hinweise müssen Sie mit erheblichen Punktabzügen rechnen!**

Aufgabe 1 (Rekursionsgleichung): (10 + 10 = 20 Punkte)

Hinweise:
- Manchmal bietet es sich an, die Rekursiongleichung zu vereinfachen.

a) Beweisen Sie per Induktion, dass $T(n) \in \mathcal{O}(n \log n)$ für

$$T(n) = \begin{cases} 1 & 1 \leq n < 16 \\ 2 \cdot T(\frac{n}{4}) + T(\frac{n}{2}) + n & n \geq 16 \end{cases}$$

b) Beweisen Sie per Induktion, dass $T(n) \in \Omega(3^{\frac{n}{3}})$ für

$$T(n) = \begin{cases} 1 & 1 \leq n \leq 3 \\ T(n-1) + T(n-2) + T(n-3) & n > 3 \end{cases}$$

Aufgabe 2 (Master Theorem): (8 + 8 + 8 + 8 + 8 = 40 Punkte)

Geben Sie für die folgenden Rekursionsgleichungen an, ob diese mit dem Master-Theorem aus der Vorlesung gelöst werden können. Begründen Sie ihre Antwort! Geben Sie auch die resultierende Komplexitätsklasse an, sofern das Master-Theorem angewendet werden kann.

a) $T(n) = 8 \cdot T(\frac{n}{2}) + 2^n$

b) $T(n) = 64 \cdot T(\frac{n}{4}) + (n^2 + 1) \cdot (n + 7)$

c) $T(n) = 27 \cdot T(\frac{n}{3}) + n \cdot (n^2 \log(n) + n)$

d) $T(n) = 16 \cdot T(\frac{n}{4}) + n \cdot (\log(n) + n)$

e) $T(n) = T(\frac{n}{3}) + f(n)$, wobei $f(n) = \begin{cases} 3n + 2^{3n} & \text{falls } n \in \{2^i \mid i \in \mathbb{N}\} \\ 3n & \text{sonst} \end{cases}$

Aufgabe 3 (Analyse): (5 + 4 + 6 + 5 + 5 = 25 Punkte)

Betrachten Sie folgenden Algorithmus.

```
T(Liste L der Laenge n mit natuerlichen Zahlen) -> Liste:
    if length(L) == 1:
        return L
    print "T: " + L
    i = S(L, 0, 0, -1)
    swap(L[0], L[i])
    return [L[0]] + T(L.tail)

S(Liste L, integer a, integer b, integer c) -> integer:
    print "S: " + L + " " + [a, b, c]
    if length(L) == 0:
        return b
    if L.head > c:
        return S(L.tail, a+1, a, L.head)
    else:
        return S(l.tail, a+1, b, c)
```

Hinweise:

- Die Liste L.tail beschreibt die Liste L ohne das erste Element.
- Für die Komplexität zählen Sie bitte die Anzahl Aufrufe von S.
- + zwischen Listen beschreibt das konkatenieren,

a) Bestimmen Sie die ersten 11 Ausgaben von print beim Aufruf $T([3, 1, 2, 4])$. Was ist die Rückgabe von $T([3, 1, 2, 4])$?

b) Beschreiben Sie in Stichpunkten die Bedeutung von S(...), und den Variablen a, b, c, und i.

c) Geben Sie eine Rekursionsgleichung für $T(n)$ und bestimmen Sie die Asymptotische Komplexität von $T(n)$ mittels der Rekursionsgleichung.

d) Geben Sie eine Liste L der Länge 6, sodass T(L) 5 Swaps ausführt.

e) Gibt es auch eine Liste L der Länge n, sodass T(L) n Swaps ausführt? Geben Sie eine kurze Begründung.

Aufgabe 4 (Sortieren): (7+8=15 Punkte)

a) Sortieren Sie das folgende Array mithilfe von Insertionsort. Geben Sie dazu das Array nach jeder Iteration der äußeren Schleife an.

Hinweise:

- Geben Sie *genau* die gewünschten Schritte an.

9	3	5	1	10	4	6

b) Wir definieren das UCI-Flaggen Problem. Die UCI Flagge besteht aus 5 Streifen mit den Farben mit den Farben Blau, Rot, Schwarz,Gelb, Grün. Gegeben ein Array mit gefärbten Einträgen (Blau, Rot, Schwarz, Gelb, Grün). Das Problem besteht darin, in linearer Laufzeit und konstantem Speicher, das Array so zu sortieren, wie sie in UCI Flagge auftreten. Dabei darf angenommen weden, dass Pointer in das Array nur konstantem Speicher benötigen.

Beschreiben Sie, wie die Lösung zum Dutch Flag Problem genutzt werden kann, um das UCI-Flaggen Problem zu lösen.

Hinweise:

- Sie sollen keine anderen Algorithmen zum sortieren nutzen.
- Eine einfache, elegante Lösung ergibt mehr Punkte.