Lösungshinweise zur Probeklausur

Aufgabe 1

a) $T(n) = 2T(n/2) + n^3$

Hier ist das Master-Theorem anwendbar mit $a = 2, b = 2, f(n) = n^3$.

$$\begin{aligned}
&n^{\log_b a} = n^{\log_2 2} = n \\
&f(n) = n^3 = \Omega(n^{\log_2 2+2}) \\
&af(n/b) = 2(n/2)^3 = 1/4\, n^3 \leq cf(n),\ c = 1/4 \text{ (Regularitätsbedingung)}
\end{aligned}$$

$\Rightarrow$ Fall 3 Master-Theorem: $T(n) = \Theta(n^3)$.

b) $T(n) = T(n-1) + n$

Die Master-Methode ist hier ungeeignet. Rekursionsbaum aufstellen:

$$\begin{array}{ccc}
n & \longrightarrow & n \\
| & & \\
n-1 & \longrightarrow & n-1 \\
| & & \\
n-2 & \longrightarrow & n-2 \\
| & & \\
\vdots & & \\
| & & \\
2 & \longrightarrow & 2 \\
| & & \\
1 & \longrightarrow & 1 \\
 & & \overline{}
\end{array}$$

Insgesamt: $\Theta(n^2)$

Die Vermutung $T(n) = \Theta(n^2)$ muß bewiesen werden! Substitutionsmethode, um $T(n) = \Omega(n^2)$ und $T(n) = O(n^2)$ zu zeigen, also $T(n) = \Theta(n^2)$.

c) $T(n) = T(\sqrt{n}) + 1$

Variablentransformation: $m = \log_2 n,$ also $n = 2^m$. Setze $S(m) := T(2^m)$.

$$S(m) = T(2^m) = T(n) = T(\sqrt{n}) + 1 = T(2^{m/2}) + 1 = S(m/2) + 1$$

Anwendung des Master-Theorems auf die Rekursionsgleichung $S(m) = S(m/2) + 1$ mit den Parametern $a = 1,\ b = 2, f(m) = 1$.

$$\begin{aligned}
&m^{\log_b a} = m^{\log_2 1} = 1 \\
&f(m) = 1 = \Theta(m^{\log_2 1})
\end{aligned}$$

$\Rightarrow$ Fall 2 Master-Theorem: $S(m) = \Theta(\log_2 m)$, also

$$T(n) = T(2^m) = S(m) = \Theta(\log_2 m) = \Theta(\log_2 \log_2 n).$$

Aufgabe 2

Eine iterative Variante der Prozedur MAX-HEAPIFY(A, i) ist:

```
MAX-HEAPIFY(A, i)
fertig ← FALSE
while fertig ≠ TRUE
      do
         l ← LEFT(i)
         r ← RIGHT(i)
         if l ≤ heap-size[A] and A[l] > A[i]
            then largest ← l
            else largest ← i
         if r ≤ heap-size[A] and A[r] > A[largest]
            then largest ← r
         if largest = i
            then fertig ← TRUE
            else vertausche A[i] ↔ A[largest]
                 i ← largest
```

Aufgabe 3

Idee: Berechne das Feld $C[0, \ldots, k]$ wie in COUNTING-SORT. $C[i]$ enthält dann die Anzahl der Elemente im Feld $A[1, \ldots, n]$, die kleiner oder gleich i sind, $i = 0, \ldots, k$. Vorverarbeitungszeit: $\Theta(n + k)$. Nun gilt:

$$
\begin{aligned}
|\{x \in A : a \leq x \leq b\}| &= |\{x \in A : x \leq b\}| - |\{x \in A : x < a\}| \\
&= |\{x \in A : x \leq b\}| - |\{x \in A : x \leq a - 1\}| \\
&= C[b] - C[a - 1],
\end{aligned}
$$

wobei $C[-1] := 0$.

Aufgabe 4

Idee: Stelle die Zahlen aus dem Bereich $0, \ldots, n^2 - 1$ als zweistellige Zahlen zur Basis n dar, denn für $z \in \{0, \ldots, n^2 - 1\}$ gilt

$$z = z_1 * n + z_0, \quad z_0, z_1 \in \{0, \ldots, n - 1\}.$$

Algorithmus:

Bilde $z_0 = z \mod n$ und $z_1 = z/n$. Sortiere danach die zweistelligen Zahlen $z_1 z_0$ mit RADIX-SORT.

Kosten: Zwei Aufrufe von COUNTING-SORT mit Laufzeit $O(n + n - 1) = O(n)$.

Aufgabe 5

a) Korrektheitsbeweis (Induktion über $n = length(A)$)
Damit der rekursive Aufruf von DUCK-SORT (Z. 6 - 8) erfolgt, muß $j - i + 1 \geq 3$, also $j - i \geq 2$.
Daher Basisfälle $n = 1$ und $n = 2$.

- $n = 1$: $\text{DUCK-SORT}(A, 1, 1)$ o.k., Ende Z. 4.
- $n = 2$: $\text{DUCK-SORT}(A, 1, 2)$ vertauscht bei Bedarf die Feldelemente $A[1]$ und $A[2]$, Ende Z. 4.
- Annahme: DUCK-SORT sortiert Eingabefelder der Länge $k,\ 1 \leq k < n,$ korrekt.
  Sei nun $A[1, \ldots, n]$ ein Eingabefeld der Länge n.
  Dann sortiert $\text{DUCK-SORT}(A, 1, n-k)$ (Z. 6) die ersten $2n/3$ Elemente von A korrekt, danach werden die letzten $2n/3$ Elemente von A sortiert (Z. 7).
  Ergebnis: Die letzten $n/3$ Elemente sind danach die $n/3$ größten von A und befinden sich an der richtigen Position.
  Der erneute Aufruf von $\text{DUCK-SORT}(A, 1, n-k)$ (Z. 8) sortiert nun die restlichen $2n/3$ Elemente von A.

b) Rekursionsgleichung für die Worst-Case-Laufzeit von DUCK-SORT:

$$T(n) = 3T(2n/3) + \Theta(1)$$

Die Master-Methode mit $a = 3, b = 3/2$ und $f(n) = \Theta(1)$ angewendet:

$$n^{\log_b a} = n^{\log_{3/2} 3}$$
$$f(n) = \Theta(1) = O(n^{\log_{3/2} 3 - 2})$$

$\Rightarrow$ Fall 1 Master-Theorem: $T(n) = \Theta(n^{\log_{3/2} 3}), \quad \log_{3/2} 3 \approx 2.710.$

c)

	Worst-Case-Laufzeit
INSERTION-SORT	$\Theta(n^2)$
MERGE-SORT	$\Theta(n \log_2 n)$
HEAP-SORT	$\Theta(n \log_2 n)$
QUICK-SORT	$\Theta(n^2)$
DUCK-SORT	$\Theta(n^{2.7\cdots})$