Musterlösungen
zur Hauptklausur
„1663 Datenstrukturen“
12. August 2000
Aufgabe 1

(a)

Einfügen von 55:

Rotation. Einfügen von 58:
Doppelrotation. Einfügen von 45:

```
30
  /  \
20   55
 |    |*
10   50
 |     |
25    60
 |     |
58    70
 |  |
40  45
 | |
```

Doppelrotation:

```
30
 /  \
20   55
 |    |
10   50
 |     |
25    60
 |     |
45    70
 |     |
40    58
 |     |
```

Wir erhalten zunächst:

Rotation:

```plaintext
Aufgabenblatt 4, Seite 4

Kurs 1663 „Datenstrukturen“

Musterlösungen zur Klausur vom 12.08.2000

(b)

Wir erhalten zunächst:

Rotation:
```
Doppelrotation:

(c)


**Aufgabe 2**

Ausgangsfolge: 35 62 28 50 11 45

nach $i = 1$: 11 62 28 50 35 45

nach $i = 2$: 11 28 62 50 35 45

nach $i = 3$: 11 28 35 50 62 45

nach $i = 4$: 11 28 35 45 62 50

nach $i = 5$: 11 28 35 45 50 60
Aufgabe 3

Die Knoten werden in lexikographischer Reihenfolge betrachtet. $G$ nach Bearbeitung von $A$:

$G$ nach Bearbeitung von $B$:
$G$ nach Bearbeitung von $C$:

$G$ nach Bearbeitung von $D$:

Aufgabe 4

(a)
Es gibt im Wesentlichen drei Möglichkeiten, wie zwei Rechtecke zueinander liegen können:


(b)
Sei $I$ eine Datenstruktur, die eine Menge von Intervallen speichert und als Operationen das Einfügen und Löschen von Intervallen und das Auffinden aller Intervalle, die eine Query-Koordinate enthalten, erlaubt. Während des Sweeps werden dann folgende Aktionen durchgeführt:

1. Falls eine linke Rechteckkante angetroffen wird, füge das Intervall-Objekt $([y_1, y_2], r)$ in $I$ ein. Hierbei bezeichnen $y_1$, $y_2$ die $y$-Koordinaten der linken Rechteckkante und $r$ den Namen des Rechtecks.

2. Falls eine rechte Rechteckkante angetroffen wird, entferne das Intervall $([y_1, y_2], r)$ aus $I$.

3. Falls ein Punkt $([x, y], r, Punkt)$ angetroffen wird, bestimme $A := \pi_2((([y_1, y_2], r') \in I \mid y \in [y_1, y_2]));$ gib alle Paare in $A \times \{ r \}$ aus.

Der Segmentbaum ist gerade eine solche Datenstruktur, wie sie in 1. beschrieben ist.
Aufgabe 5

(a)

(i) \( f(n) = 2n, n > 0 \)

\[ f(n) = O(n) \]

(ii) \( f(n) = 2^n - 1, n > 0 \)

\[ f(n) = O(2^n) \]

(iii) \( f(n) = n + 1, n > 1 \)

\[ f(n) = O(n) \]

(b)

\( M(1) = 0 \)

\[ M(n) = 2 M(n/2) + 1, n > 1, n \text{ Zweierpotenz} \]

(c)

\[ M(n) = n - 1. \]

Die Aussage ist laut (b) korrekt für \( n = 1 \). Unter der Annahme der Korrektheit auch für alle \( n > 1, n \text{ Zweierpotenz} \), ergibt sich dann der Induktionsschritt von \( n \) nach \( 2n \) durch

\[ M(2n) = 2 M(2n/2) + 1 = 2 M(n) + 1 = 2(n - 1) + 1 = 2n - 1. \]
Aufgabe 6

(a)
Wir benutzen den temporären Speicherplatz der Länge $k$ für ein Array $B$ mit den Indizes 1, ..., $k$. Zweck von $B$ ist es, die Häufigkeit des Auftretens eines Elements aus \{1, ..., $k$\} festzuhalten. $B$ repräsentiert also eine Multimenge. Anschließend wird dann jedes Element $i$ aus \{1, ..., $k$\} $B[i]$-mal nach $A$ geschrieben.

\textbf{algorithm} CountingSort($A$, $B$, $n$, $k$);
\{Initialisierung.\}
\begin{align*}
&\text{for } i := 1 \text{ to } k \text{ do } B[i] := 0 \text{ od}; \\
&\text{\{Die Anzahl der Elemente, die gleich } i \text{ sind, wird in } B[i] \text{ festgehalten.\}} \\
&\text{for } j := 1 \text{ to } n \text{ do } B[A[j]] := B[A[j]] + 1 \text{ od}; \\
&\text{\{Zurückschreiben der Elemente nach } A.\} \\
&l := 1; \\
&\text{for } i := 1 \text{ to } k \text{ do } \\
&\quad \text{for } j := 1 \text{ to } B[i] \text{ do } \\
&\quad\quad A[l] := i; \\
&\quad\quad l := l + 1 \\
&\quad \text{od} \\
&\text{od} \\
\text{end CountingSort.}
\end{align*}

(b)
Die erste \textit{for}-Schleife benötigt $O(k)$ Zeit, die zweite \textit{for}-Schleife $O(n)$ Zeit. Obwohl zuletzt zwei ineinander verschachtelte \textit{for}-Schleifen auftreten, ist deren Laufzeit $O(n+k)$, da die Summe aller $B[i]$ für $i$ aus \{1, ..., $k$\} gleich $n$ ist. Da gemäß Aufgabenstellung $k < n$ angenommen wird, ist $k = O(n)$. Insgesamt ergibt sich somit eine Laufzeit von $O(n)$, die die Forderung, $\Omega(n \log n)$ zu unterschreiten, erfüllt.