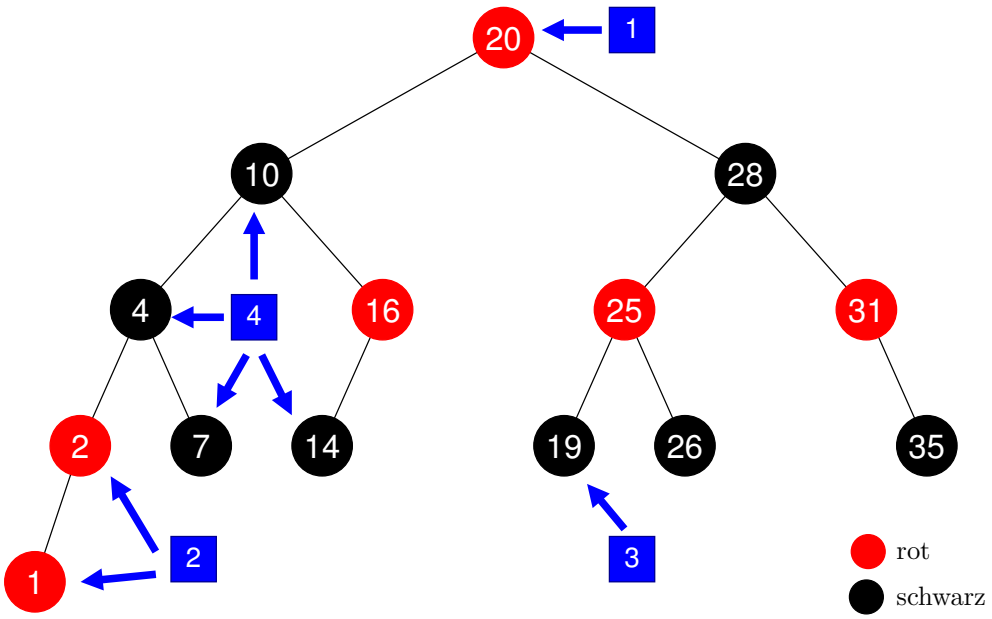
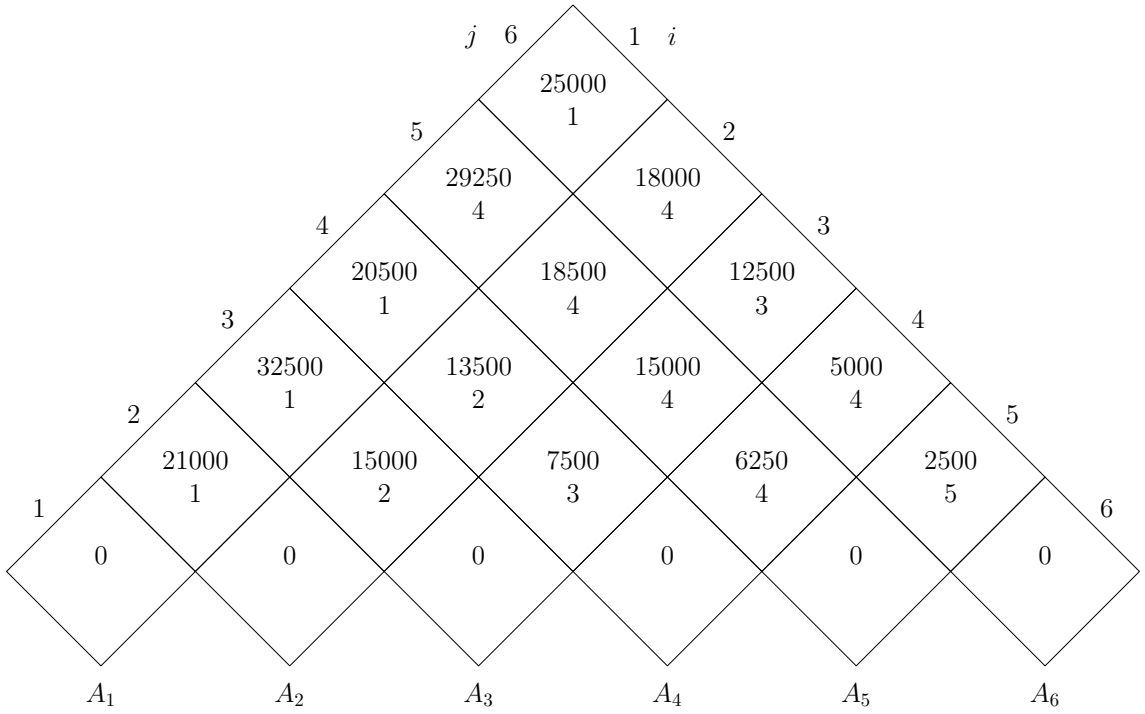
Klausur zur Vorlesung
Algorithmen und Datenstrukturen 2 (SS 2006)
Lösungshinweise

Aufgabe 1.



1. Die Wurzel muss schwarz sein.
2. Zwei benachbarte Knoten dürfen nicht rot sein..
3. Die Binärbaum-Eigenschaft wird verletzt.
4. Zwei verschiedene Pfade von der Wurzel zu einem Blatt haben eine unterschiedliche Anzahl schwarzer Knoten.

Aufgabe 2.



a) Kosten der Multiplikation von A_5 und A_6:

$$\begin{aligned} m[5,6] &= 10 \cdot 25 \cdot 10 = 2500 \\ s[5,6] &= 5 \end{aligned}$$

b) Kosten der Multiplikation von A_2, A_3, A_4, A_5 und A_6 hängt ab von der Position der Klammerung:

$$\begin{aligned} k = 2 &: 0 + 12500 + 20 \cdot 30 \cdot 10 = 18500 \\ k = 3 &: 15000 + 5000 + 20 \cdot 25 \cdot 10 = 25000 \\ k = 4 &: 13500 + 2500 + 20 \cdot 10 \cdot 10 = 18000 \\ k = 5 &: 18500 + 0 + 20 \cdot 25 \cdot 10 = 23500 \end{aligned}$$

Das Minimum liegt bei $k = 4$. Somit ist:

$$\begin{aligned} m[2,6] &= 18000 \\ s[2,6] &= 4 \end{aligned}$$

c) Die optimale Klammerung lautet:

$$(A_1)((A_2(A_3A_4))(A_5A_6))$$

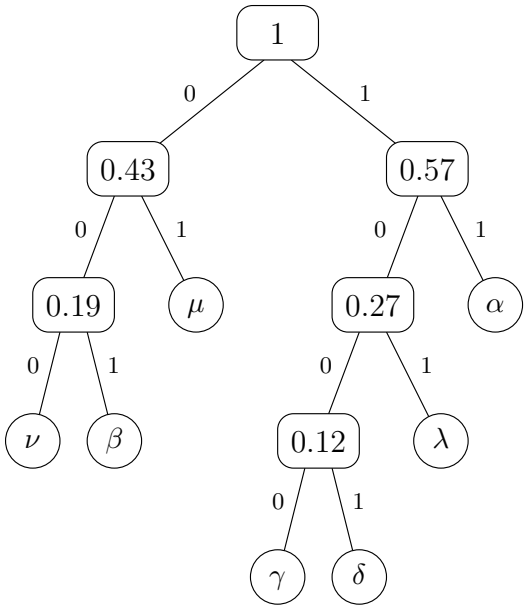
Aufgabe 3.

THREE(x)
Input: Zeiger x auf die Wurzel eines Binärbaums
Output: Zeiger auf das drittkleinste Element, falls ein solches existert, NIL, sonst.

```
1  y := TreeMinimum(x);
2  if (y ≠ NIL) then
3     y := TreeSuccessor(y);
4     if (y ≠ NIL) then
5        return TreeSuccessor(y);
6     else
7        return NIL;
8  else
9     return NIL;
```

Aufgabe 4.

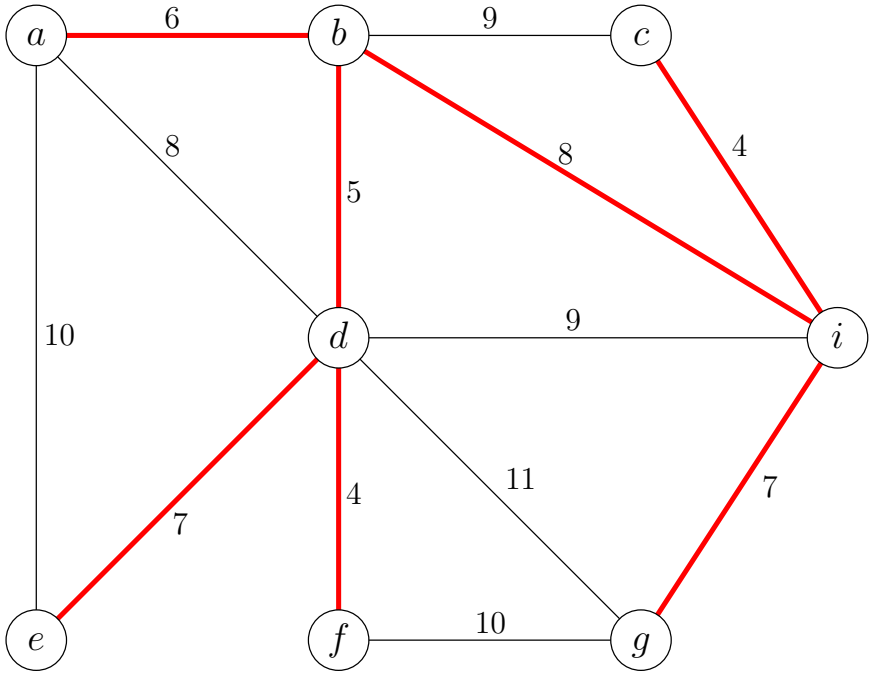
a) Der Huffman Algorithmus liefert diesen Präfixcode:



b) Die Kodierung der Buchstaben lautet:

$$
\begin{array}{lcl}
\alpha & : & 11 \\
\beta & : & 001 \\
\gamma & : & 1000 \\
\delta & : & 1001 \\
\lambda & : & 101 \\
\mu & : & 01 \\
\nu & : & 000
\end{array}
$$

Aufgabe 5.



Reihenfolge	1	2	3	4	5	6	7	8
Knoten	d	f	b	a	e	i	c	g

Aufgabe 6.

a) $T(n)$ bezeichnet die Worst Case Laufzeit von $\textsc{Maximum}(A, \ell, r)$ für ein Array A mit $n = r - \ell + 1$ Elementen. Es gilt:

$$T(n) = \begin{cases} c_{\text{if}} & n = 1 \\ c_{\text{else}} + 2 \cdot T\left(\frac{n}{2}\right) & n > 1 \end{cases}$$

Hierbei steht c_{if} für die Anzahl der Rechenschritte zur Ausführung des if-Teils von $\textsc{Maximum}(A, \ell, r)$ und c_{else} für die Anzahl der Rechenschritte zur Ausführung des else-Teils ohne die rekursiven Funktionsaufrufe.

Der schlimmste Fall tritt ein, wenn $n = 2^k$. Hieraus folgt:

$$\begin{aligned} T(n) &= \underbrace{(c_{\text{else}} + 2 \cdot (c_{\text{else}} + 2 \cdot (c_{\text{else}} + \ldots 2 \cdot (c_{\text{else}} + 2 \cdot c_{\text{if}})))}_{k\text{-mal}} \\ &= \sum_{i=0}^{k-1} c_{\text{else}} \cdot 2^i + 2^k \cdot c_{\text{if}} \\ &= c_{\text{else}} \cdot (2^k - 1) + c_{\text{if}} \cdot 2^k \\ &= (c_{\text{else}} + c_{\text{if}}) \cdot 2^k - c_{\text{else}} \\ &\geq (c_{\text{else}} + c_{\text{if}}) \cdot 2^k \end{aligned}$$

Da $k = \log_2 n$, folgt $T(n) = (c_{\text{else}} + c_{\text{if}}) \cdot n = O(n)$, wobei $n_0 = 1$ und $c = c_{\text{else}} + c_{\text{if}}$.

b) Zur Korrektheit führen wir einen Induktionsbeweis über die Anzahl der Elemente im Array A.

Induktionsbehauptung: Für jedes Array A und $n = r - \ell + 1$ Elementen liefert $\textsc{Maximum}(A, \ell, r)$ das Maximum der Elemente in $\{A[\ell], \ldots, A[r]\}$.

Induktionsanfang: $n = 1$. In diesem Fall ist $\ell = r$, d.h., das Array $A[\ell..rv]$ enthält nur ein Element. Dieses wird in Zeile 2 zurückgegeben.

Induktionsschritt: $n \rightsquigarrow n + 1$. Da $\ell < r$ ist, wird der else-Teil des Algorithmus ausgeführt. Da $m = \ell + \left\lfloor \frac{r-\ell+1}{2} \right\rfloor$, gilt $\ell \leq m-1 < m \leq r$. Also enthält sowohl $A[\ell..m-1]$ als auch $A[m..r]$ höchstens n Elemente. Also ist laut Induktionsbehauptung w_1 das Maximum in $A[\ell..m-1]$ und w_2 das Maximum in $A[m..r]$. In den Zeilen 7 bis 10 wird das Maximum von w_1 und w_2 berechnet und zurück gegeben. Dieser Wert ist das Maximum in $A[\ell..r]$.