



Universität Karlsruhe (TH)

Institut für Programmstrukturen und Datenorganisation (IPD)

Informatik I WS 20003/04

Dozent: Prof. Dr.rer.nat. G. Goos

Übungsleiter: Tom Gelhausen

<http://www.infoeins.de>

goos@ipd.info.uni-karlsruhe.de

gelhausen@fzi.de

Musterlösung 1 - (60T / 0P)

Binärarithmetik, Huffman-Codierung

Ausgabe: 17.10.2003

Abgabe: 24.10.2003

13:30 Uhr

Einwurf im Keller des Informatik-Hauptbaus (Geb. 50.34)

Aufgabe 1: Zahldarstellungen (10T)

1.1 Basiswechsel (6T)

Füllen Sie die Lücken in der Tabelle aus:

Dezimalzahl	Binärzahl	Oktalzahl	Hexadezimalzahl
180			
	1100,001		
		5,71	
			A8

Lösung:

Jede richtige Zahl ist 0,5T wert.

Dezimalzahl	Binärzahl	Oktalzahl	Hexadezimalzahl
180	10110100	264	B4
12,125	1100,001	14,1	C,2
5,890625	101,111001	5,71	5,E4
168	10101000	250	A8

1.2. Zweierkomplement (4T)

Stellen Sie folgende Dezimalzahlen als Binärzahlen im Einer- und Zweierkomplement dar. Ergänzen Sie eventuell fehlende Stellen, so dass deren Anzahl durch vier teilbar ist:

- -43
- -664
- -273,125
- +499

Lösung:

Korrektes Einer- bzw. Zweierkomplement je 0,5T.

- 43 = 0010 1011
Einerkomplement = 1101 0100
Zweierkomplement = 1101 0101
- 664 = 0010 1001 1000
Einerkomplement = 1101 0110 0111
Zweierkomplement = 1101 0110 1000

- 273,125 = 0001 0001 0001, 0010
 Einerkomplement = 1110 1110 1110, 1101
 Zweierkomplement = 1110 1110 1110, 1110
- 499 = 0001 1111 0011
 Einerkomplement = 0001 1111 0011
 Zweierkomplement = 0001 1111 0011

Aufgabe 2: Addieren und Subtrahieren (10T)

2.1 Addition (4T)

Addieren Sie die folgenden Zahlen im Binärsystem und kontrollieren Sie danach Ihr Ergebnis im Dezimalsystem:

- 100010 + 001101
- 110101 + 010110
- 010001,01 + 100010,00
- 101000,10 + 010101,11

Lösung:

$$\begin{array}{r}
 100010 \\
 + 001101 \\
 \hline
 101111
 \end{array}
 \qquad
 \begin{array}{r}
 34 \\
 + 13 \\
 \hline
 47
 \end{array}
 \quad (1T)$$

$$\begin{array}{r}
 110101 \\
 + 010110 \\
 \hline
 1001011
 \end{array}
 \qquad
 \begin{array}{r}
 53 \\
 + 22 \\
 \hline
 75
 \end{array}
 \quad (1T)$$

$$\begin{array}{r}
 010001,01 \\
 + 100010,00 \\
 \hline
 110011,01
 \end{array}
 \qquad
 \begin{array}{r}
 17,25 \\
 + 34,00 \\
 \hline
 51,25
 \end{array}
 \quad (1T)$$

$$\begin{array}{r}
 101000,10 \\
 + 010101,11 \\
 \hline
 111110,01
 \end{array}
 \qquad
 \begin{array}{r}
 40,5 \\
 + 21,75 \\
 \hline
 62,25
 \end{array}
 \quad (1T)$$

2.2 Subtrahieren (6T)

Subtrahieren Sie folgende positiven Zahlen im Binärsystem und kontrollieren Sie danach Ihr Ergebnis im Dezimalsystem:

- 100011 - 010110
- 001101,00 - 000110,01

Führen Sie nun folgende Rechnungen durch, indem Sie die Dezimalzahlen zunächst in Zweierkomplement-Darstellung mit sechs Ziffern umwandeln:

- (+21) + (-13)
- (-23) + (+11)

Lösung:

100011	35	
- 010110	- 22	
<u>111</u>		
001101	13	(1,5T)
001101,00	13,00	
- 000110,01	- 6,25	
<u>1111</u>		
000110,11	6,75	(1,5T)
010101	21	
+ 110011	+ (-13)	
<u>111</u>		
1001000	8	(1,5T)
101001	-23	
+ 001011	+ 11	
<u>11</u>		
0110100	-12	(1,5T)

Aufgabe 3: Multiplizieren und Dividieren (20T)**3.1 Multiplikation (10T)**

Multiplizieren Sie folgende positive Zahlen im Binärsystem und kontrollieren danach die Ergebnisse:

- 111100 * 1010
- 001110 * 10111

Führen Sie nun folgende Multiplikationen in Zweierkomplementdarstellung durch (verwenden Sie hierbei eine ausreichende Anzahl Ziffern) und überprüfen danach das Ergebnis im Zehnersystem:

- (-10) * (+10)
- (-14) * (-18)

Lösung:

$$\begin{array}{r}
 111100 * 1010 \\
 111100 \\
 111100 \\
 \hline
 0
 \end{array}$$

$$\begin{array}{r}
 111 \\
 1001011000
 \end{array}$$

$$60 * 10 = 600 \quad (2T)$$

$$\begin{array}{r}
 001110 * 10111 \\
 001110 \\
 001110 \\
 001110 \\
 \hline
 001110
 \end{array}$$

$$\begin{array}{r}
 1121 \\
 101000010
 \end{array}$$

$$14 * 23 = 322 \quad (2T)$$

```

111110110 * 1010
...111110110
...11111110110
-----
0
...111110011100

```

$$(-10) * (+10) = -100 \quad (3T)$$

```

111110010 * 111101110
...111110010
...1111110010
...11111110010
...111111110010
...1111111110010
...11111111110010
...111111111110010
...1111111111110010
-----
0
...00000000011111100

```

$$(-14) * (-18) = 252 \quad (3T)$$

3.2 Division (10T)

Führen Sie folgende Divisionen im Binärsystem durch (positive Zahlen) und kontrollieren Sie die Ergebnisse danach im Dezimalsystem:

- 1010010111 : 110011
- 100100001 : 10001

Lösung:

```

1010010111 : 110011 = 01101
-110011
-----
11111
-110011
-----
1100
  -0
-----
11001
-110011
-----
0

```

$$663 : 51 = 13 \quad (5T)$$

```

100100001 : 10001 = 10001
-10001
-----
10001
-10001
-----
0

```

$$289 : 17 = 17 \quad (5T)$$

Aufgabe 4: Kodierung (20T)

- Nennen Sie zwei Anwendungsgebiete, für die Kodierungsverfahren eingesetzt werden. (1T)
- Wie viele Bits werden mindestens benötigt, um alle Groß- und Kleinbuchstaben der deutschen Sprache sowie die Ziffern 0-9 und das Leerzeichen deterministisch binär zu codieren? (1T)
- Mit wie vielen Bits kann demnach der Satz "Blaukraut bleibt Blaukraut und Brautkleid bleibt Brautkleid" binär kodiert werden? Wie viele Bits wären nötig, wenn man sich auf Kleinbuchstaben und das Leerzeichen beschränkt? (2T)
- Kodieren Sie obigen Satz nach dem Huffman-Verfahren aus der Vorlesung. Sie können dabei alle Buchstaben als Kleinbuchstaben interpretieren. Konstruieren Sie hierzu zunächst den zugehörigen Binärbaum und beschriften Sie die einzelnen Kanten und Blätter. Wie viele Bits werden nun zur Darstellung des Satzes benötigt? (10T)
- Was bedeutet Präfixcodierung? (1T)
- Wie groß ist also der mittlere Informationsgehalt eines Zeichens (nach Shannon)? Setzen Sie die entsprechenden Werte in die Gleichung ein und berechnen Sie das Ergebnis. (5T)

Lösung:

a. Datenkompression, Verschlüsselung, Datenübertragung,...

b. $26 + 26 + 10 + 1 = 63$ Zeichen

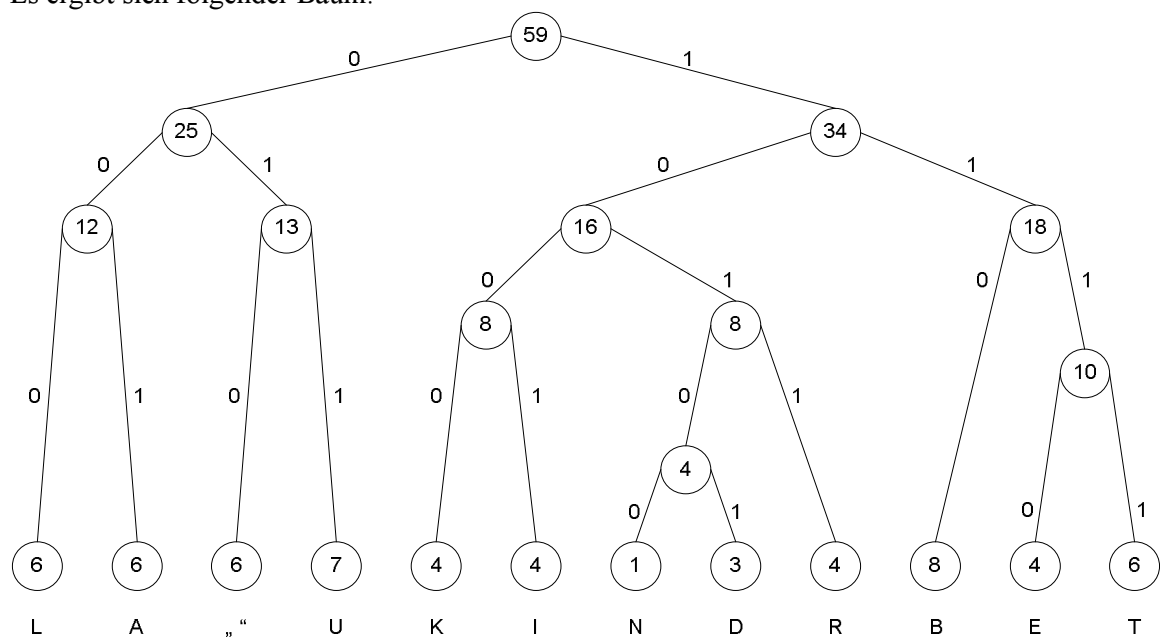
$$\Rightarrow \lceil \lg 63 \rceil = 6$$

$\Rightarrow 6$ Bits sind nötig.

c. Man benötigt $6 * 59 = 354$ Bits.

Verwendet man nur Kleinbuchstaben und das Leerzeichen, reichen 5 Bit zur Darstellung jedes Zeichens aus und man benötigt für den Satz $5 * 59 = 294$ Bits.

d. Es ergibt sich folgender Baum:



d. Damit ergeben sich folgende Kodierungen:

L 000
A 001
 010
U 011
K 1000
I 1001
N 10100
D 10101
R 1011
B 110
E 1110
T 1111

Und schließlich der gesuchte Satz, bestehend aus 207 Bit:

```
110|000|001|011|1000|1011|001|011|1111|  
010|110|000|1110|1001|110|1111|010|  
110|000|001|011|1000|1011|001|011|1111|  
010|011|10100|10101|010|  
110|1011|001|011|1111|1000|000|1110|1001|10101|  
010|110|000|1110|1001|110|1111|010|  
110|1011|001|011|1111|1000|000|1110|1001|10101
```

(Baum + Buchstabenkodierung 6T, Anzahl Bits 1T, Satz 3T)

e. Präfixkodierung, d.h. kein Code für ein Zeichen ist gleichzeitig Anfang eines anderen Codes (z.B. auch beim Morsecode).

f. Es ergibt sich:

$H(L) = - 6/59 * \lg (6/59) = 0,335$
 $H(A) = - 6/59 * \lg (6/59) = 0,335$
 $H() = - 6/59 * \lg (6/59) = 0,335$
 $H(U) = - 7/59 * \lg (7/59) = 0,365$
 $H(K) = - 4/59 * \lg (4/59) = 0,263$
 $H(I) = - 4/59 * \lg (4/59) = 0,263$
 $H(N) = - 1/59 * \lg (1/59) = 0,100$
 $H(D) = - 3/59 * \lg (3/59) = 0,219$
 $H(R) = - 4/59 * \lg (4/59) = 0,263$
 $H(B) = - 8/59 * \lg (8/59) = 0,391$
 $H(E) = - 4/59 * \lg (4/59) = 0,263$
 $H(T) = - 6/59 * \lg (6/59) = 0,335$

Demnach ist der mittlere Informationsgehalt eines Zeichens:

$H = H(A) + H() + H(U) + H(K) + H(I) + H(N) + H(D) + H(R) + H(B) + H(E) + H(T) = 3,468 \text{ bit.}$