4.6 Üben und Vertiefen

Aufgabe 1

a) $1010101_2 = 85_{10}$ $10111_2 = 23_{10}$
$1101010101_2 = 853_{10}$ $1011101101_2 = 749_{10}$

b) $A07_{16} = 2567_{10}$ $1BD_{16} = 445_{10}$
$D01_{16} = 3329_{10}$ $FFFF_{16} = 65535_{10}$

Aufgabe 2

a) $19_{10} = 10011_2$ $123_{10} = 1111011_2$
$1023_{10} = 1111111111_2$ $1000_{10} = 1111101000_2$

b) $A_{16} = 10_{10} = 1010_2$ $FF_{16} = 255_{10} = 11111111_2$
$AB_{16} = 171_{10} = 10101011_2$ $123_{16} = 291_{10} = 100100011_2$

Aufgabe 3

a) $19_{10} = 13_{16}$ $123_{10} = 7B_{16}$
$1023_{10} = 3FF_{16}$ $1000_{10} = 3E8_{16}$

b) $1010101_2 = 55_{16}$ $10111_2 = 17_{16}$
$1101010101_2 = 355_{16}$ $1011101101_2 = 2ED_{16}$

Aufgabe 4

a) Die Addition und Subtraktion wird ähnlich dem Dezimalsystem durchgeführt, nur dass ein Übertrag im Hexadezimal- bzw. Binärsystem bei 16 bzw. 2 stattfindet.

b)

```
  6B2        155        1001001        111011
 +578       - 77       +1011010       - 10110
  C2A         DE        10100011        100101
```

c)

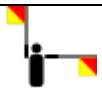
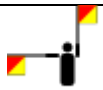
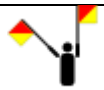
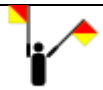
```
  1714       341         73            59
 +1400      -119       + 90           -22
  3114       222        163            37
```

© C.C.Buchner Verlag, Bamberg

Aufgabe 5

a) Das Winkeralphabet wird zur Nachrichtenübermittlung auf Schiffen oder auch auf Land verwendet.

Winkeralphabet:

A (1)	B (2)	C (3)	D (4)	E (5)	F (6)	G (7)
H (8)	I (9)	 J	K (0)	L	M	N
O	 P	Q	R	S	 T	U
V	W	X	Y	Z	 Zahlen folgen	

b)

S E H R G U T D U

K A N N S T E S

© C.C.Buchner Verlag, Bamberg

Aufgabe 6

a)

Name	Code	Beschreibung
GS1 Data-Bar	(01) 01234567890128 (15) 057072	https://www.etikettenwissen.de/wiki/GS1_DataBar
Code 39	BAR57072	https://www.activebarcode.de/codes/code39
Code128	Barcode507072	https://www.activebarcode.de/codes/code128
UPC-A GTIN-12 UPC-E	1 23601 05707 2	https://www.activebarcode.de/codes/upca_upce
SCC-14	3 07 12345 00001 0	https://www.activebarcode.de/codes/itf14
PZN (Pharma-Zentral-Nummer)	PZN - 2357072	https://www.activebarcode.de/codes/pzn
ISSN	ISSN 1144-875X 9 771144 875007	https://www.activebarcode.de/codes/issn
PDF417		https://www.activebarcode.de/codes/pdf417

b) Die Barcodes geben folgende Zahlen aus:
90311017, - 12345678, A+A-1234, 9783661380117

Die 2D-Codes ergeben folgende Informationen:

- Cool oder? Ist so ähnlich wie ein QR-Code
- Ich bin kein QR-Code, sondern ein Data Matrix-Code
- Überweisung QR-Code für 99,99€

© C.C.Buchner Verlag, Bamberg

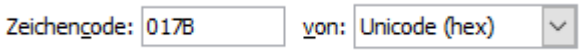
Aufgabe 7

a) Der Unicode ist ein Zeichensatz für 145 000 Zeichen, welcher sowohl Schriftzeichen verschiedener Sprachen als auch Emojis und weitere Symbole enthält.

b)

U+1F4BB	ergibt	💻
U+002B	ergibt	+
U+1F3EB	ergibt	🏫
U+21D2	ergibt	⇒
U+1F60E	ergibt	😎

c) In Word gibt es z. B. eine Tabelle mit Sonderzeichen. Diese sind unter „Symbol“ und „Weitere Symbole“ zu finden. Dort kann man den Unicode der Zeichen sehen.



Um den Code umzuwandeln gibt man diesen in der Form U+2764 ein und drückt anschließend *ALT+C*, sodass der Code umgewandelt wird, im Beispiel ❤.

Aufgabe 8

a) 09511234 ➔ Zahlen Modus ➔ `0001`

b) 8 Ziffern ➔ `0001 0000001000`

c) Im Ziffernmodus werden jeweils drei Ziffern als eine Zahl zusammengefasst und dann in eine 10-Bit-Dualzahl umgewandelt:

095 = 95 = 0001011111_2
112 = 0001110000_2
34 = 0000100010_2

`00010000001000 0001011111 0001110000 0000100010`

d) Es kann auch die Alphanummerische oder Byte-Codierung verwendet werden. Dies verändert die QR-Codes im Aufbau.

© C.C.Buchner Verlag, Bamberg

e)

🡐 Eingetragene Bitfolge aus c)

f)

g) Es wird individuell gescannt und überprüft.

h) Veränderung von einigen Pixeln ergeben immer noch die Telefonnummer 09511234. Dies ist durch die Fehlerkorrektur gewährleistet. Erst bei einer größeren Änderung wird der Code nicht mehr erkannt.

Aufgabe 9

a) Es handelt sich um Länderkennzeichen, z.B. bei Internetadressen.

b) AF = Afghanistan
DE = Deutschland
BH = Bahrain
IN = Indien
MU = Mauritius

Aufgabe 10

a) Programmiersprache: Lisp

```
(defun fat (n)
  (cond
    ((= n 1) 1)
    (t (* n (fat(- n 1))))))
```

Es wird die Fakultät von n berechnet. Dies kann in einem Online-Compiler für LISP (https://onecompiler.com/commonlisp/3zfpudama) und dem zusätzlichen Aufruf `(princ (fat 5))` getestet werden.

© C.C.Buchner Verlag, Bamberg

Programmiersprache: C

```
double fkt(double r, double w) {
double wz, kw;
if ((r ==0 ) || (w == 0)) goto fin;
kw = 1/ w ;
wz =  pow(r, kw)
fin: return wz;}
```

Es wird die w-te Wurzel aus der Zahl r berechnet. Dies kann in einem Online-Compiler für C (https://onecompiler.com/c/) und dem folgenden Programm getestet werden.

```
#include <stdio.h>
#include <math.h>
double fkt(double r, double w)
{
    double wz, kw;
    if ((r ==0 ) || (w == 0)) goto fin;
    kw = 1/ w ;
    wz = pow(r, kw);
    fin: return wz;
}

int main()
{
  printf("%f",fkt(9,2));
}
```

Programmiersprache: Rust

```
mod squares {
    pub fn sfz( n: i32) -> i32 {
        (0 .. n+1).fold(0, |a, b| a + b) }
}
```

Es werden die natürlichen Zahlen von 0 bis n+1 addiert. Dies kann in einem Online-Compiler für Rust (https://onecompiler.com/rust/) und dem folgenden Programm getestet werden.

```
fn main()
{
  println!("Ausgabe:{}",squares::sfz(5));
}

mod squares
{
 pub fn sfz( n: i32) -> i32 {
 (0 .. n+1).fold(0, |a, b| a + b) }
}
```

© C.C.Buchner Verlag, Bamberg

Programmiersprache: Python

```
def fkt(a,b):
    if a==b: return a
    else:
        if a>b: a = a-b
        else: b = b-a
        fkt(a,b)
    return a
```

- Es wird der ggT (größte gemeinsamer Teiler) der beiden Zahlen a und b berechnet. Dies kann in einem Online-Compiler für Python (https://onecompiler.com/python/) und dem folgenden Programm getestet werden.

```
def fkt(a,b):
 if a==b: return a
 else:
   if a>b: a = a-b
   else: b = b-a
   fkt(a,b)
 return a

result = fkt(15, 10)
print("Ausgabe:",result)
```

Aufgabe 11

a) Die Position des Orts auf der Erde wird mit zwei Gradangaben bestimmt. Dabei gibt die erste Zahl den Breitengrad von -90° bis +90° an und die zweite Zahl den Längengrad von -180° bis +180° an.

b) Eingabe in eine Onlinekartensuche liefert folgende Ortsnamen, evtl. muss etwas gezoomt werden.

48.22935694097753, 7.5218317967583115 ➔ Mussig
47.744550294567645, 12.134816585075894 ➔ Inn
49.88620782758452, 9.83339953216241 ➔ Main
48.222481189487674, 10.17626802069046 ➔ Buch
51.94591396554895, 10.714839461752582 ➔ Schauen

Nachricht: „Muss ich in mein Buch schauen.“

© C.C.Buchner Verlag, Bamberg

Aufgabe 12

a) Olivia hat für jede Farbe einen Buchstaben verwendet, so steht ein W für einen weißen, G für einen grünen, B für einen braun und S für einen schwarzen Pixel. Jeder Block entspricht einer Zeile im Bild.

b) Da in Olivias Zeichenfolge sehr viele gleiche Buchstaben stehen, hat Mattis dies verkürzt und statt 13 – mal W zu schreiben dies auf 13W verkürzt. So steht z. B. 1G für einen grünen Pixel, 9S für neun schwarze Pixel.

c) Schreibt man Olivias Darstellung direkt untereinander kann man anhand der Buchstaben gleich die Form des Bilds erkennen. Man muss aber viel mehr Zeichen aufschreiben.
Mattis‘ Schreibweise ist kürzer zu notieren, man kann aber daraus nicht direkt das Bild erahnen.

d) Olivia: 8 · 9 = 72 Zeichen ➔ 72 Byte = 576 Bit
Mattis: 15 Ziffern und 14 Zeichen ➔ 15 · 4 Bit + 14 Byte = 172 Bit

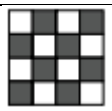
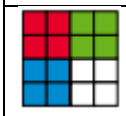
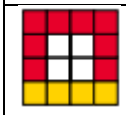
e) 172 : 576 ≈ 0,2986 ➔ Es können 70,14 % an Bit eingespart werden.

Aufgabe 13

a) 42 4d ist der Header (Kopf) der Datei und gibt an, dass es sich um eine BMP-Datei handelt
66 gibt die Größe des Bilds in Byte an, 66_{16} = 102 (Byte)
00 00 00 04 00 00 00 04 bestimmt Länge und Breite des Bild, hier je 4 Pixel.
01 Anzahl der Farbebenen, hier nur eine Ebene
18 Anzahl der Bit um die Farbe eines Pixels zu beschreiben,
hier 18_{16} = 24 (Bit).
FF FF … Hier sind die Farbinformationen des Bildes gespeichert. Drei Byte ergeben jeweils einen Pixel. FF FF FF steht bspw. für die Farbe Weiß und 00 00 00 für Schwarz.

b) Z. B. einen weißen Pixel in Rot ändern. Dann ändert sich der FF FF FF - Block in eine andere Zahlenkombination, nämlich 00 00 FF.

c) Der Gelbe Bereich muss angepasst werden:
Hinweis: Die 1. Zeile entspricht der letzten Zeile des Bilds, d.h. sie werden von unten nach oben aufgebaut! Auch der RGB-Code ist Rückwärts gespeichert.

	FF FF FF 00 00 00 FF FF FF FF FF FF 00 00 00 FF FF FF 00 00 00 00 00 00 FF FF FF 00 00 00 FF FF FF FF FF FF 00 00 00 FF FF FF 00 00 00 FF FF FF
	FF 00 00 FF 00 00 FF FF FF FF FF FF FF 00 00 FF 00 00 FF FF FF FF FF FF 00 00 FF 00 00 FF 00 FF 00 00 FF 00 00 00 FF 00 00 FF 00 FF 00 00 FF 00
	00 FF FF 00 FF FF 00 FF FF 00 FF FF 00 00 FF FF FF FF FF FF FF 00 00 FF 00 00 FF FF FF FF FF FF FF 00 00 FF 00 00 FF 00 00 FF 00 00 FF 00 00 FF

© C.C.Buchner Verlag, Bamberg

Aufgabe 14

a) Stellenwerttafel:

Stufenzahl	32 768	4096	512	64	8	1

b) 800 : 512 = 1 R 288
288 : 64 = 4 R 32
32 : 8 = 4 R 0
→ $800_{10} = 1440_8$

64 : 64 = 1 R 0
→ $64_{10} = 100_8$

1024 : 512 = 2 R 0
→ $1024_{10} = 2000_8$

© C.C.Buchner Verlag, Bamberg