

Codierung Text /Töne und Bilder mit Übungen und Aufgaben

Textcodierung (ASCII): Diese Tabelle wurde 1967 erstmals als Standard veröffentlicht unter American Standard Code for Information Interchange (ASCII)

	0100000	0	0110000	@	1000000	P	1010000	`	1100000	p	1110000
!	0100001	1	0110001	A	1000001	Q	1010001	a	1100001	q	1110001
"	0100010	2	0110010	B	1000010	R	1010010	b	1100010	r	1110010
#	0100011	3	0110011	C	1000011	S	1010011	c	1100011	s	1110011
\$	0100100	4	0110100	D	1000100	T	1010100	d	1100100	t	1110100
%	0100101	5	0110101	E	1000101	U	1010101	e	1100101	u	1110101
&	0100110	6	0110110	F	1000110	V	1010110	f	1100110	v	1110110
'	0100111	7	0110111	G	1000111	W	1010111	g	1100111	w	1110111
(0101000	8	0111000	H	1001000	X	1011000	h	1101000	x	1111000
)	0101001	9	0111001	I	1001001	Y	1011001	i	1101001	y	1111001
*	0101010	:	0111010	J	1001010	Z	1011010	j	1101010	z	1111010
+	0101011	;	0111011	K	1001011	[1011011	k	1101011	{	1111011
,	0101100	<	0111100	L	1001100	\	1011100	l	1101100		1111100
-	0101101	=	0111101	M	1001101]	1011101	m	1101101	}	1111101
.	0101110	>	0111110	N	1001110	^	1011110	n	1101110	~	1111110
/	0101111	?	0111111	O	1001111	_	1011111	o	1101111	DEL	1111111

Musikcodierung

1/2 c	00000	1/2 c'	01000	1/4 b	10000	1/8 g	11000
1/2 d	00001	1/2 d'	01001	1/4 h	10001	1/8 a	11001
1/2 e	00010	1/4 c	01010	1/4 c'	10010	1/8 b	11010
1/2 f	00011	1/4 d	01011	1/4 d'	10011	1/8 h	11011
1/2 g	00100	1/4 e	01100	1/8 c	10100	1/8 c'	11100
1/2 a	00101	1/4 f	01101	1/8 d	10101	1/8 d'	11101
1/2 b	00110	1/4 g	01110	1/8 e	10110	1/2 Pause	11110
1/2 h	00111	1/4 a	01111	1/8 f	10111	1/4 Pause	11111

Codierung von Pixelgrafiken

Ein Rasterbild mit verschiedenen Auflösungen (dpi – dots per inch -> Pixel pro Zoll):



1 Zoll = 2,54 cm

Für die Codierung der Farben wird, wie bei allen Codierungen (außer Zahlenwerte) eine Codetabelle benötigt. Sie weist jeder Farbe einen Binärcode zu.

Pixelgrafiken mit Graustufen besitzen 256 Grautöne: 0 => schwarz bis 255 => weiß.

Für jeden Pixel wird 1 Byte benötigt => $2^8 = 256$ Grautöne um ihn abzuspeichern.

P2	24	36	255
Codierungsinformationen -> siehe unten			
129	136	178	157
160	174	182	85
202	204	207	179
218	219	222	225
225	226	227	228
228	228	230	229
232	232	233	232
236	237	235	234
235	235	233	234
233	233	231	232
230	233	232	232
231	231	232	233
232	232	233	234
235	234	233	233
237	235	231	232
235	234	233	233
236	237	235	234
238	231	230	227
228	228	226	225
229	229	228	227
230	230	229	228
231	231	230	229
232	232	231	230
233	233	232	231
234	234	233	232
235	235	234	233
236	236	235	234
237	237	236	235
238	238	237	236
239	239	238	237
240	240	239	238
241	241	240	239
242	242	241	240
243	243	242	241
244	244	243	242
245	245	244	243
246	246	245	244
247	247	246	245
248	248	247	246
249	249	248	247
250	250	249	248
251	251	250	249
252	252	251	250
253	253	252	251
254	254	253	252
255	255	254	253



Farbige Pixelgrafiken

Für jeden Farbton wird im Farbraum R-G-B ein Byte benötigt, also benötigt ein Pixel eines True-Color-Bildes 3 Byte => 24 Bit für $2^{24} = 16,7$ Mio. Farben.

Die Farben werden durch die Anteile von Rot, Grün und Blau im Bereich von 0 bis 255 erzeugt.

z.B. Schwarz: 0, 0, 0 Weiß: 255, 255, 255 Veilchen-Lila: 115, 71, 255

Der Speicherbedarf eines Pixelbildes ist abhängig von der Pixelzahl des Bildes.

Breite bzw. Höhe [in Pixel] = Höhe bzw. Breite [in cm] x Auflösung [in dpi] / 2,54

Speicher [in Byte] = Breite [in Pixel] * Höhe [in Pixel] * Farbwert [in Byte]

Auflösung: dots per inch (dpi) => Anzahl der Pixel pro Zoll (2,54 cm)

Der tatsächliche Speicherbedarf ist meist kleiner als der berechnete, weil die Bilder normalerweise in einem Bildformat abgespeichert (z.B. jpg) werden, dass die Daten komprimiert (→ nächstes Stoffgebiet).

Codierung unkomprimierter Bilder

Quelle Wikipedia:

Portable Anymap ist eine Familie von einfachen Dateiformaten zur Speicherung von Rastergrafiken.

Zu den PNM-Formaten gehören: Portable Bitmap (PBM), Portable Graymap (PGM) und Portable Pixmap (PPM)

Die **Codierungsinformationen** im Dateikopf aller drei Dateiformate sind gleich.

Der Dateikopf ist folgendermaßen aufgebaut:

1. **Magic Number:** beschreibt das Format der Bilddaten
2. Breite des Bildes Höhe des Bildes (dezimal)
3. maximale Farbwertzahl (dezimal)



Magic Number	Dateityp	Kodierung
P1	Portable Bitmap	ASCII
P2	Portable Graymap	ASCII
P3	Portable Pixmap	ASCII
P4	Portable Bitmap	Binär
P5	Portable Graymap	Binär
P6	Portable Pixmap	Binär
P7	Portable Anymap	Binär

Das PGM-Format

Vorbereitung:

- Kopiere dir den Ordner **Bildcodierung** auf deinen Datenträger.

Übung:

1a: Erläutere die Bildcodierungsinformationen des Eselbildes auf der Vorderseite und berechne den Speicherbedarf des Bildes in kB.

1b: Öffne die Datei `bitmap-graustufen.pgm` mit  [03_4 Pixelbilder_erstellen_Schüler.html](#) .

Achte darauf, den Link unbedingt mit Chrome zu öffnen!

Erläutere die Codierung dieses Bildes und berechne den Speicherplatz.

1c: Verändere den Code mit dem Editor so, dass der Smiley eine schwarze Sonnenbrille trägt und der Mund lachend und besser sichtbar ist.

Übung 2: Erzeuge das folgende Bild und speichere es.



PPM-Format

Bei der additiven Farbmischung kann das RGB-Modell das gesamte Farbspektrum erzeugen:

255, 0, 0 -> Rot, 0, 255, 0 -> Grün, 0, 0, 255 -> Blau, 0, 0, 0 -> Schwarz, 255, 255, 255 -> Weiß

Alle anderen Farbtöne ergeben sich durch Mischen, z.B. 130, 65, 160 -> Lila

Nutze zur Veranschaulichung das Programm `farbwahl.exe`

Eine Pixelgrafik wird im PPM-Format so beschrieben:

P3

3 4

255

0 0 0 0 0 0 0 0 0

255 255 255 255 255 255 255 255 255

0 0 255 0 255 0 255 0 0

255 255 0 255 0 255 0 255 255



Übung 3: Erläutere am oben gezeigten Beispiel die Bestandteile des PPM-Formats und notiere dir die Tripel der vorkommenden Farben.

Übung 4: Skizziere das Bild.

P3

4 2

255

255 0 0 255 0 0 0 255 0 0 0 255

255 0 0 255 0 0 255 0 0 255 0 0

- Verringere den maximalen Farbwert des Bildes in kleinen Schritten ohne die Farbwerte des Bildes zu verändern, wie wirkt sich das aus?

- Stelle den maximalen Farbwert auf 50 und erzeuge alles grüne Pixel.

Übung 5: Ein quadratisches Bild soll aus 7 x 7 Pixeln aufgebaut werden. Die Randpixel sollen alle blau, die inneren Pixel die gelb sein und das mittige Pixel schwarz. Erstelle den Quelltext im PPM-Format.

Übung 6: Erstelle mit dem Editor das vorgegebene Bild.



Übung 7: Wie viele verschiedene Farben lassen sich mit dem Zahlentripel beschreiben, wenn alle Farbwerte im Bereich 0 . . . 255 liegen dürfen?

Übung 8: Eine Farbe kann (z.B. in HTML) auch Hexadezimal #FF6825; beschrieben werden. Rechne den Farbcode FF|68|25 in Dezimalwerte um. Welcher Farbe entspricht das? Gib den Farbwert 130,65,160 --> Lila hexadezimal an!

Übung 9: Decodiere den folgenden Bitcode:

a) als Zeichenkette b) als Audiocode c) als PGM mit: P2 5 1
0 1 1 0 1 1 1 1 | 0 1 1 1 0 0 1 1 | 0 1 1 0 0 1 1 0 | 0 1 0 1 0 1 1 0 | 0 1 0 1 0 1 1 1

Übung 10: Für ein Logo wurde eine Grafik erstellt. Hier der Quelltext zu dieser Grafik:

P2 4 5 3 0 2 2 2 0 3 3 3 0 2 2 3 0 3 3 3 0 0 0 0

- Was wird hier dargestellt? Male es.
- Erstelle den Quelltext zum „Negativ“ der Grafik.

Übung 11:

- Digitalkameras bieten das Pixelformat 2560 x 1920 an. Gib den Speicherbedarf im True Color-Format in sinnvoller Einheit an.
- Ein Foto von 16 x 9 cm hat eine Auflösung von 300 dpi und wird in True Color abgespeichert. Berechne den (unkomprimierten) Speicherbedarf.
- Ein 70-minütiger Film hat Full-HD-Auflösung (1920 x 1080 Pixel) und eine Framerate von 24 FPS. Berechne den unkomprimierten Speicherbedarf des Filmes.

Vektorgrafik

Diese Grafiken setzen sich aus ‚Formbausteinen‘ zusammen, deren Eigenschaften (Farbe, Größe, Koordinaten...) genau beschrieben werden.
Das Aussehen des Bildes wird mittels Befehle (ähnlich wie HTML) beschrieben.
Der folgende Quelltext beschreibt eine Vektorgrafik.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd" >
<svg height="100" width="100" xmlns="http://www.w3.org/2000/svg">
<rect fill="yellow" height="100" width="100" x="0" y="0"></rect>
<circle cx="50" cy="50" fill="red" r="10"></circle>
</svg>
```

Übung 12:

- Skizziere die beschriebene Vektorgrafik.