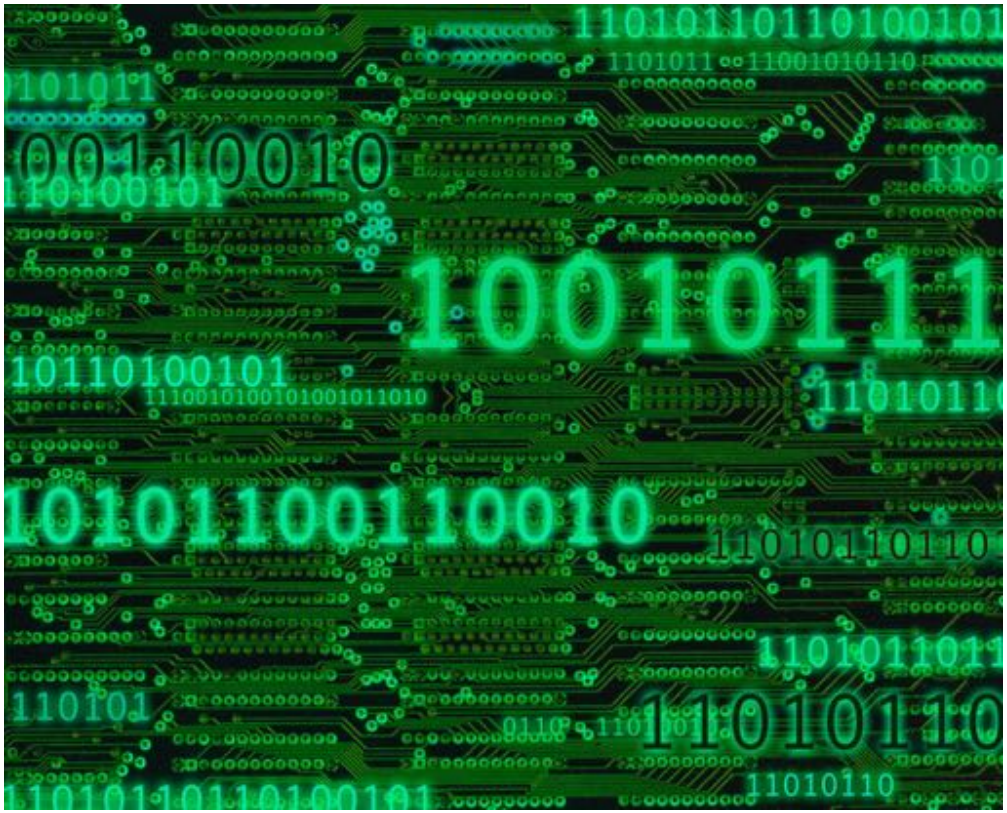
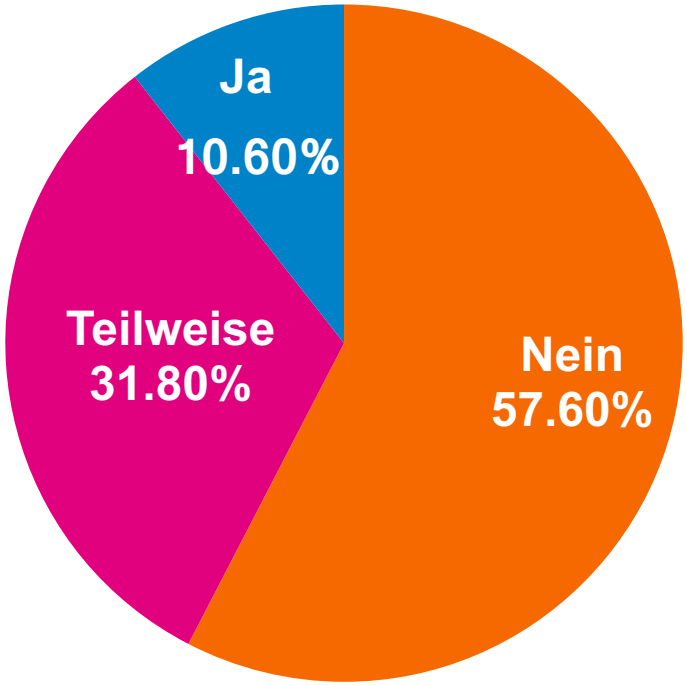
Codes und Codierung



Dr. Michael Hielscher

Aus der Eingangsbefragung von Studierenden

Ich weiss, wie Codes in unserem Alltag (z.B. Barcodes auf der Milchpackung) funktionieren.



n = 85

Codes im Alltag

Codes im Alltag

Welche Informationen stecken im Code?
Preis? Gewicht? Grösse?

Sind die Codes in allen Geschäften gleich?

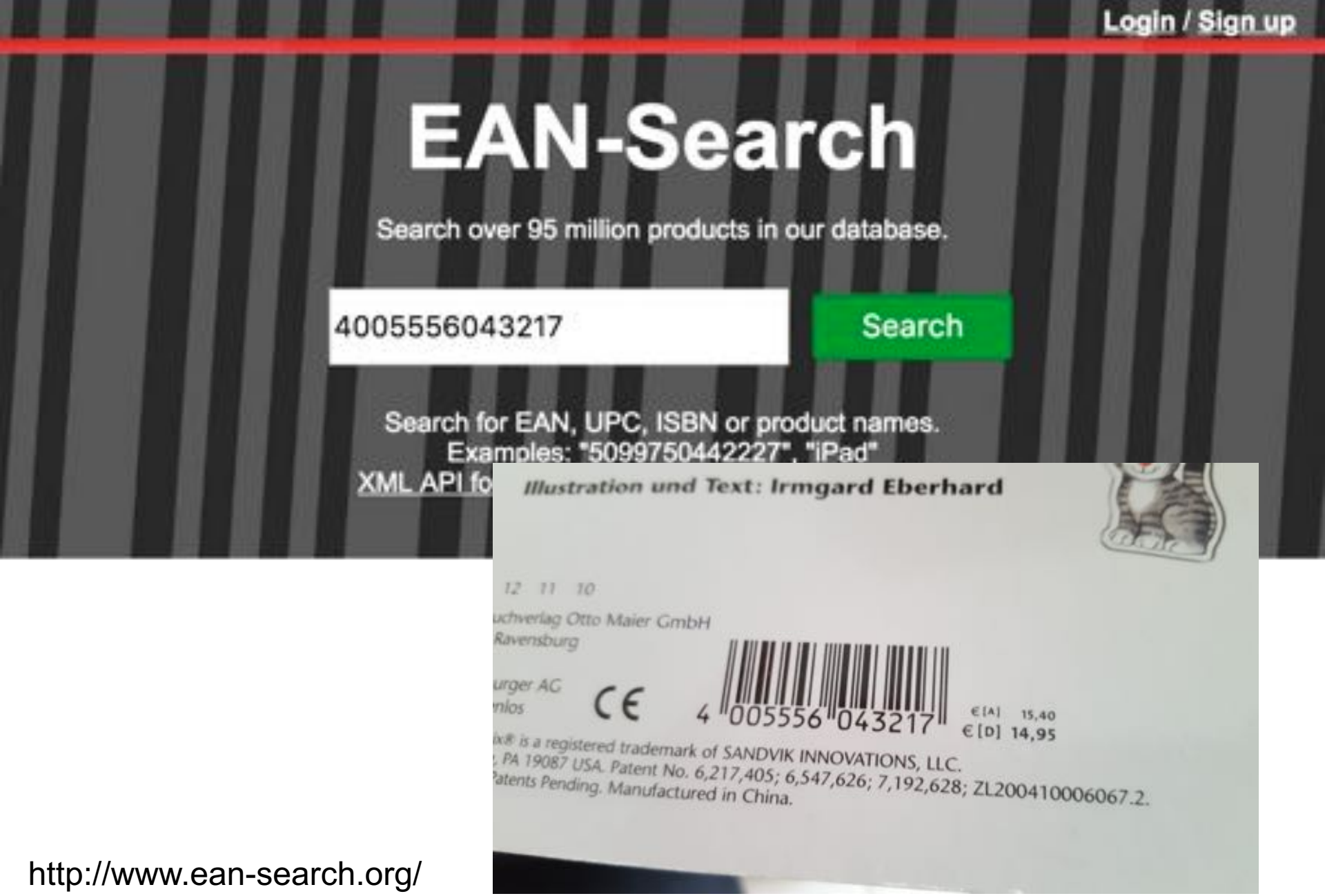
fragt an

Kasse

Datenbank

liefert zurück

EAN	Hersteller	Name	Bestand	Preis
4005556043217	Ravensburger	ministeps Magnetbuch Was gehört wohin	45	14,95
4005556043224	Ravensburger	Kleine Ente kommst du mit?	32	12,95
...	...	...	...	...



http://www.ean-search.org/

Login / Sign up

EAN-Search

Search over 95 million products in our database.

4005556043217

Search

Search for EAN, UPC, ISBN or product names.
Examples: "5099750442227", "iPad"
XML API for applications or mass queries available.

Product name for EAN 4005556043217:

Ravensburger Bücher ministeps Magnetbuch Was gehört wohin

4 005556 043217

http://www.ean-search.org/

Was hat das mit dem Zaubertrick zu tun?

Prinzip der Prüfsumme

Prinzip der Prüfsumme

Prinzip der Prüfsumme

Verallgemeinert spricht der Informatiker von „Redundanz“ (Zusatzinformation)

Hintergrund:

- Bei analogen Medien wie einer Schallplatte hört man bei einem Staubkorn, einem Haar oder einem Kratzer ein Knacken beim Abspielen. Solche Störungen und Fehler sind ganz normal und führten früher auch dazu, dass eine Kopie immer ein klein wenig schlechter wurde als die vorherige Version
- Computer arbeiten mit Informationen, indem sie alles mit 0 und 1 notieren wie die Kärtchen im Zaubertrick (An/Aus)
- Beim Lesen von Informationen von einer CD, einem USB-Stick oder einer Festplatte passieren ebenfalls Lesefehler durch Umwelteinflüsse (Schmutz, elektrische Störungen). Da die Information aber nicht mehr direkt ausgegeben wird wie bei der Schallplatte, sondern erst von einem Computer verarbeitet wird, können zusätzliche Informationen hinzugefügt werden, damit Lesefehler erkannt und teilweise sogar automatisch korrigiert werden können.

Verallgemeinert spricht der Informatiker von „Redundanz“ (Zusatzinformation)

Hintergrund:

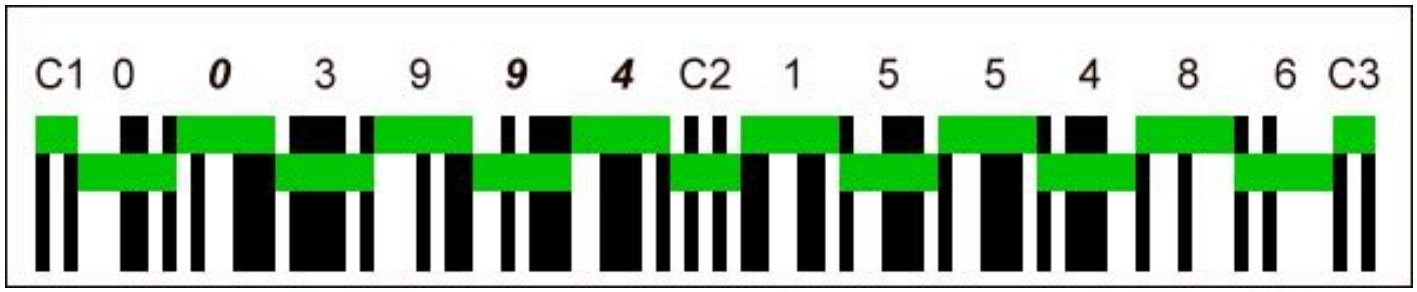
- Auf einer CD sind zum Beispiel 25% der gespeicherten Daten solche zusätzlichen Redundanten-Informationen. Ein Kratzer auf der CD ist nicht hörbar, weil er einfach „rausgerechnet“ werden kann, solange es nicht zu viele Kratzer gibt.
- Lese- und Übertragungsfehler sind normal und passieren immer. Fehlererkennende und Fehlerkorrigierende Codes erlauben eine perfekte fehlerfreie Kopie der gespeicherten Information. Nur so kann ein Computer über das Internet mit einem anderen kommunizieren oder wir heute mit dem Handy telefonieren oder surfen.
- Dieses Prinzip findet sich in praktischen allen digitalen Geräten wieder und unser Handynetz, Internet, E-Banking würde alles ohne diese Prinzip nicht funktionieren. Es bildet ein fundamentales Prinzip der Informatik, was seit über 50 Jahren Bestand hat.

Wie entstehen eigentlich solche Codes?

Was sind die Überlegungen dahinter?

Codierung hat unterschiedliche Ziel:
- möglichst wenig 0en und 1en benötigen
- Fehler erkennbar oder sogar korrigierbar

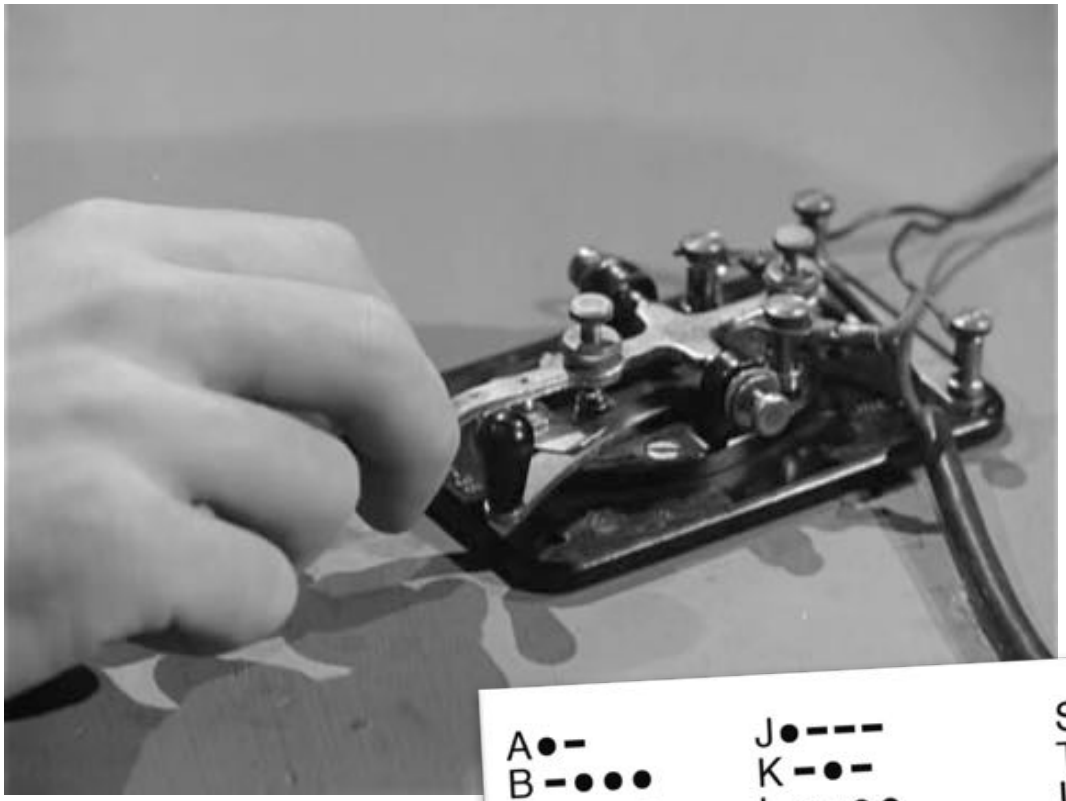
EAN 13 Codierung



Ziffer	Muster			Linienbreitenabfolge		Kodierung der 13. Ziffer
	links		rechts	links ung./ rechts	links ger.	
	ungerade Quersumme	gerade Quersumme	(gerade Quersumme)			
0	0001101	0100111	1110010	3211	1123	UUUUUU GGGGGG
1	0011001	0110011	1100110	2221	1222	UUGUGG GGGGGG
2	0010011	0011011	1101100	2122	2212	UUGGUG GGGGGG
3	0111101	0100001	1000010	1411	1141	UUGGGU GGGGGG
4	0100011	0011101	1011100			G
5	0110001	0111001	1001110			G
6	0101111	0000101	1010000			G
7	0111011	0010001	1000100			G
8	0110111	0001001	1001000			GGGGGG
9	0001011	0010111	1110100	3112	2113	UGGUGU GGGGGG

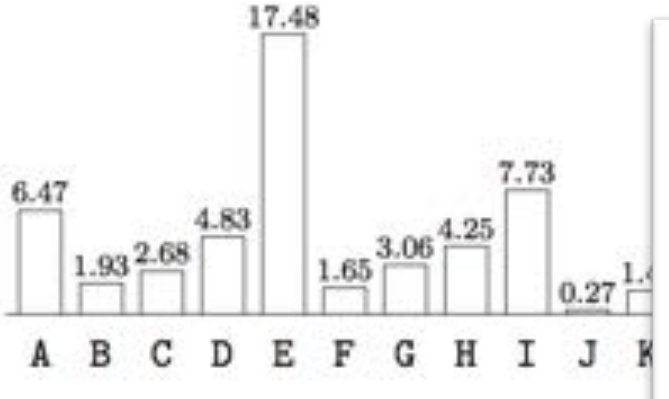
Ziel:
- Lesefehler vermeiden
- die S/W Folgen für jede Ziffer sollten sich möglichst stark unterscheiden
- Leserichtung egal

Morsecode als einfacheres Beispiel



A ●−	J ●−−−	S ●●●
B −●●●	K −●−	T −
C −●−●	L ●−●●	U ●●−
D −●●	M −−	V ●●●−
E ●	N −●	W ●−−
F ●●−●	O −−−	X −●●−
G −−●	P ●−−●	Y −●−−
H ●●●●	Q −−●−	Z −−●●
I ●●	R ●−●	

Kompression durch kürzere Codes für häufige Buchstaben (Entropiekodierung)



A ●−	J ●−−−	S ●●●
B −●●●	K −●−	T −
C −●−●	L ●−●●	U ●●−
D −●●	M −−	V ●●●−
E ●	N −●	W ●−−
F ●●−●	O −−−	X −●●−
G −−●	P ●−−●	Y −●−−
H ●●●●	Q −−●−	Z −−●●
I ●●	R ●−●	

Abbildung 1: Häufigkeitsverteilu

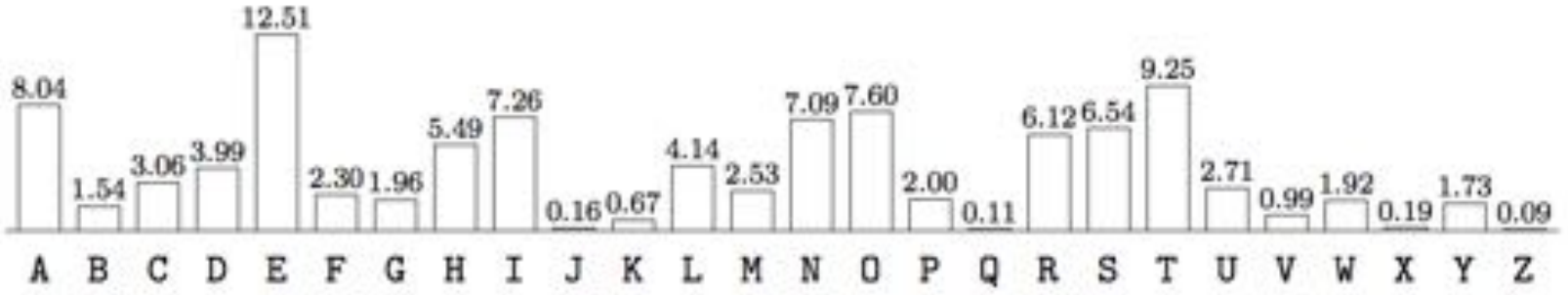


Abbildung 2: Häufigkeitsverteilung von Einzelbuchstaben im Englischen (in %).

Besonders clevere Codierung kann auch eine Verschlüsslung sein.

1827 3225XM C1626 W987
SEXTO

101

H6R 5RH DE C 1346 = 3TLE = 2TL 224 = HUW XNG =
DKRKI CUZAF MNSDC AWXVJ DVZNH DMOZN NWRJC KKJQO
ELWIK XDUUF ECEGN OUNNQ CIIZX FUTWF BTNWI GOECK
CMYUC KTTYB ZMDTU WCNWH OXOFX ERVQW JUCVY PQACQ
EBMXE NOQKF LWRWR LQKXZ BPYWR GQVYG WJDGA QXKVC
MQQJJ PVSLG WFZJZ HHWQG YFCQQ RMVRR QQIDQ QVVIW
LJLBH LHHDI OFWUY JJQGX BWPZ
CCT 2/3 RCWGN
1852 FLC

Enigma

Einfache Geheimschriften knacken

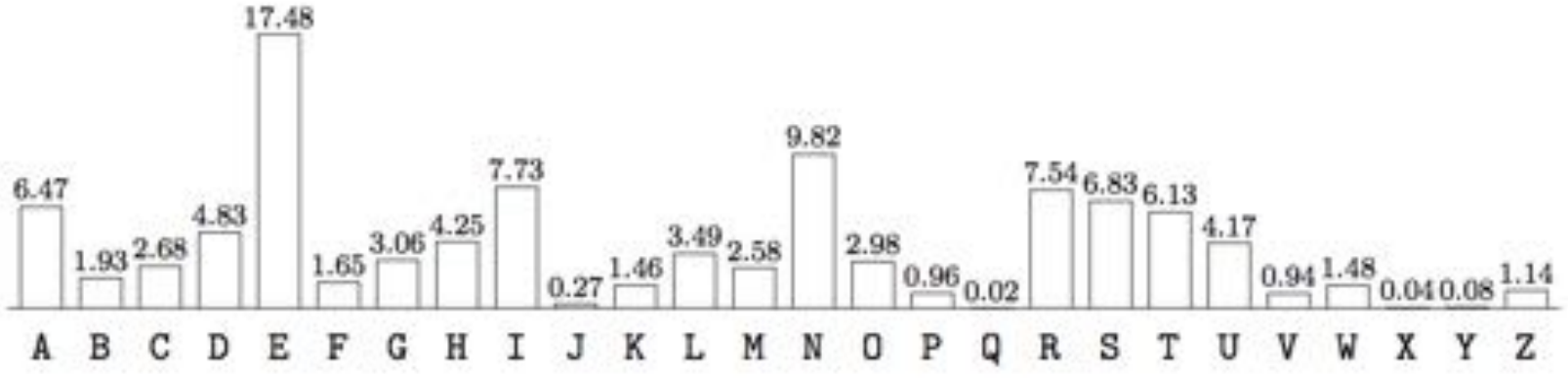
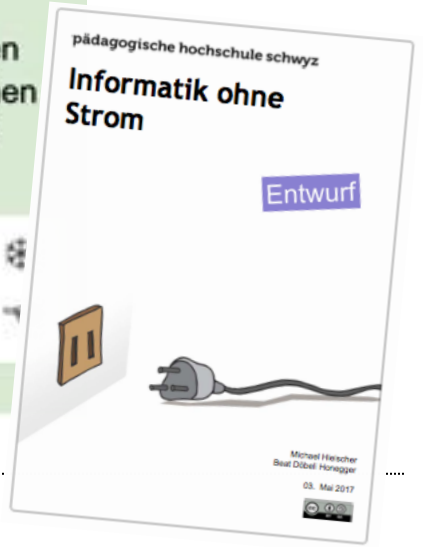


Abbildung 1: Häufigkeitsverteilung von Einzelbuchstaben im Deutschen (in %).

3. Codes knacken

Einfache Geheimschriften, bei denen ein Zeichen jeweils genau einem anderen Zeichen oder Code zugeordnet wird, lassen sich relativ leicht durch Raten entziffern. In deutschen Texten kommt der Buchstabe „E" am häufigsten vor. Ebenfalls häufig sind N, I, S, R, A und T. Schaffst du es, die folgende Botschaft zu entziffern?

Solltest du auch nach mehreren Versuchen nicht weiter kommen, findest du eine Lösungshilfe auf der Rückseite dieser Broschüre.



Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten.

Zeichen	M	P	I	S
ASCII Code	01001001	01010000	01001001	01010011


Informatik ohne Strom

Streng geheim – Text im Computer

phsz

2

Worum geht es?

Informationen wurden schon lange vor dem Computer „codiert“. Einem Symbol, einem Zeichen oder einem Laut wurde eine ganz bestimmte Bedeutung zugeordnet (z.B. Hieroglyphen, Musiknoten, Sprache).
Da der Computer nur genau zwei Symbole - 0 und 1 - kennt, müssen alle Informationen aus einleitugen Kombinationen von mehreren Den und 1en aufgeschrieben, also „codiert“ werden. „Codierung“ kann man auch für einfache Geheimschriften verwenden.
Nur wer die richtige Zuordnung von Zeichen zu Code kennt, kann sie entziffern.

TOP SECRET

Wie funktioniert es?

1. Die Geheimschrift des Computer

Hast du schon einmal eine geheime Botschaft verschickt? Vielleicht hast du schon einmal einen Brief in Spiegelschrift geschrieben oder einen, in dem alle Anfangsbuchstaben der Wörter einen gemeinen Satz gebildet haben. Es gibt ganz viele Möglichkeiten, etwas so aufzuschreiben, dass zwar alle deine Botschaft ansehen, aber nur gewisse Empfänger sie verstehen können. Der Computer verwendet auch eine Art Geheimschrift, die für uns unverständlich klingt: **01000001 01000010 01000011** diese drei Zahlenfolgen stehen für "ABC" in sogenannten ASCII-Code. In einer Tabelle wird jedem Zeichen (Buchstabe, Satzzeichen, Ziffer usw.) genau eine bestimmte Zahlenfolge aus 0 und 1 zugeordnet. Wir müssen also nur in die Tabelle schauen und ablesen, welches Zeichen sich hinter der Zahl versteckt. Kannst du den Code für X,Y und Z in der Tabelle rechts fortsetzen und ergänzen? Kannst du deinen Vornamen mit Hilfe der Tabelle als lange Folge von 0en und 1en aufschreiben? Der ASCII-Code ist schon über 50 Jahre alt und stammt aus den Anfangsjahren der Computer. Er wird aber auch heute noch an vielen Stellen verwendet. Welche Zahlenfolge damals welchem Zeichen zugeordnet wurde, ist eigentlich egal. Wichtig ist nur, dass die Zuordnung eindeutig ist (d.h. jedes Zeichen hat einen eigenen Code) und dass alle die gleiche Codetabelle verwenden.

ASCII-Code	Zeichen
00100000	Leerzeichen
01000001	A
01000010	B
01000011	C
01000100	D
01000101	E
01000110	F
01000111	G
01001000	H
01001001	I
01001010	J
01001011	K
01001100	L
01001101	M
01001110	N
01001111	O
01010000	P
01010001	Q
01010010	R
01010011	S
01010100	T
01010101	U
01010110	V
01010111	W
	X
	Y
	Z

2. Einfache Geheimschriften selbst erstellen

Kannst du das Prinzip des Codierens und Decodierens auch für Geheimschriften anwenden? Der Freimaurer-Code wurde im Mittelalter verwendet. Nur wer den Code kannte, konnte die geheime Botschaft lesen. Kannst du die folgende Botschaft entschlüsseln, wenn du die Code-Gitter darunter zur Hilfe nimmst? Kannst du eine eigene, ähnliche Geheimschrift erfinden?

Freimaurer-Code:

3. Codes knacken

Einfache Geheimschriften, bei denen ein Zeichen jeweils genau einem anderen Zeichen oder Code zugeordnet wird, lassen sich relativ leicht durch Raten entziffern. In deutschen Texten kommt der Buchstabe „E“ am häufigsten vor. Ebenfalls häufig sind N, I, S, R, A und T. Schaffst du es, die folgende Botschaft zu entziffern?

MISSISSIPPI benötigt damit 8 Bit * 11 = 88 Bit

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten.
Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4

Zunächst alle Knoten mit Häufigkeit in eine Liste legen

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort,
welches wir mit möglichst wenig 0en und 1en übertragen möchten.
Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4

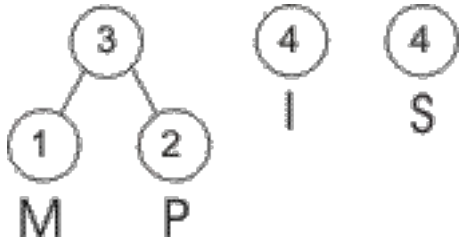
Die beiden Knoten, mit der geringsten Häufigkeit zusammenfassen unter einem neuen Elternknoten.

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten.
Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4



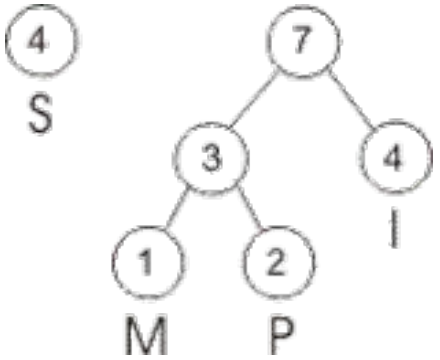
Die Summe der Häufigkeiten als neuen Elternknoten verwenden. Verfahren fortsetzen, bis alle Zeichen Teil vom Baum sind.

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten.
Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4



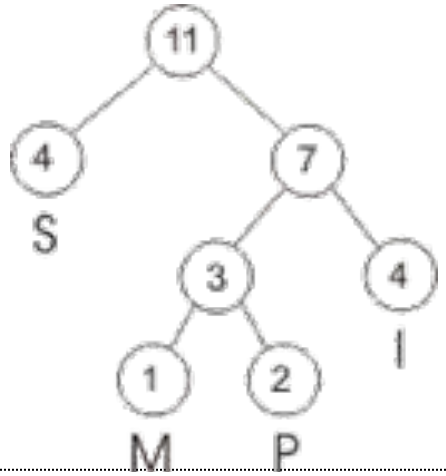
In diesem Schritt hätten wir sowohl mit I als auch mit S einen Elternknoten bilden können.

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten. Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4



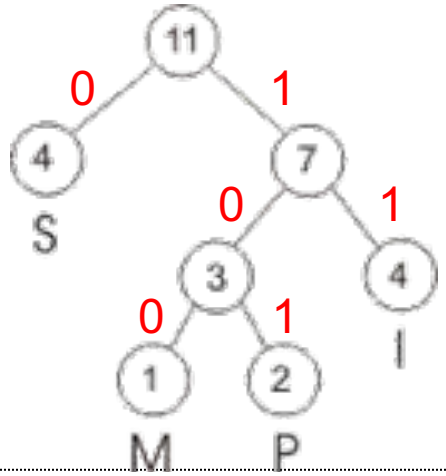
Fertig – Huffman-Baum erstellt. Jetzt einfach Binärzahlen hinzufügen

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten. Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4



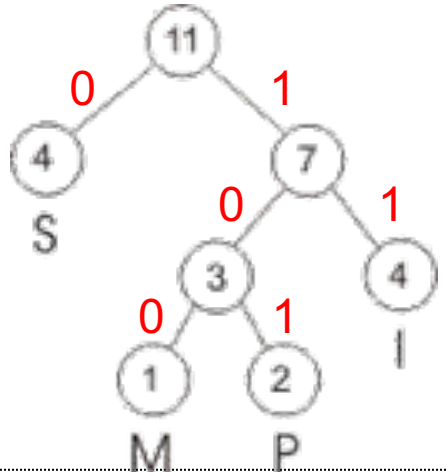
Fertig – Huffman-Baum erstellt. Jetzt einfach Binärzahlen hinzufügen

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten.
Wir können die Häufigkeit der Buchstaben zählen.

Zeichen	M	P	I	S
Häufigkeit	1	2	4	4



Codetabelle:

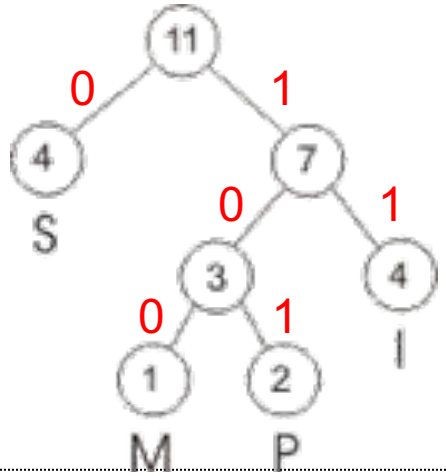
Zeichen	**Code**
S	0
I	11
P	101
M	100

Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten. Wir können die Häufigkeit der Buchstaben zählen.

M	I	S	S	I	S	S	I	P	P	I
100	11	0	0	11	0	0	11	101	101	11



Codetabelle:

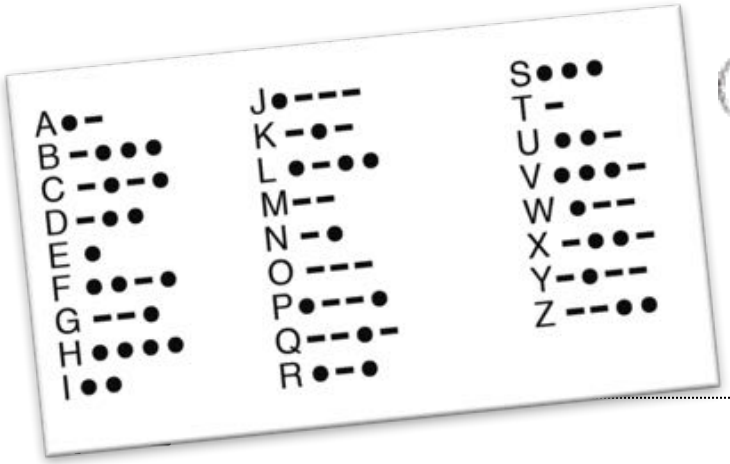
Zeichen	Code
S	0
I	11
P	101
M	100

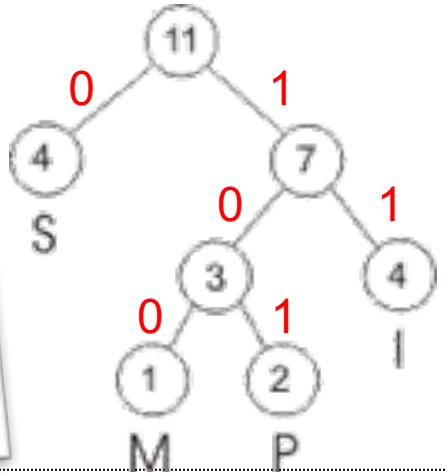
Huffman-Algorithmus

Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten. Wir können die Häufigkeit der Buchstaben zählen.

M	I	S	S	I	S	S	I	P	P	I
100	11	0	0	11	0	0	11	101	101	11





Codetabelle:

Zeichen	Code
S	0
I	11
P	101
M	100

Huffman-Algorithmus

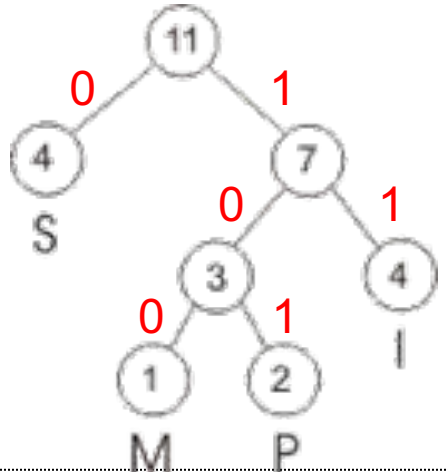
Mit dem Huffman-Algorithmus kann man eine (sub-)optimale binäre Codierung berechnen.

Beispiel:
Angenommen der Text "MISSISSIPPI“ wäre ein typisches Wort, welches wir mit möglichst wenig 0en und 1en übertragen möchten.
Wir können die Häufigkeit der Buchstaben zählen.

M	I	S	S	I	S	S	I	P	P	I
100	11	0	0	11	0	0	11	101	101	11

UTF-8 Zeichensatz:

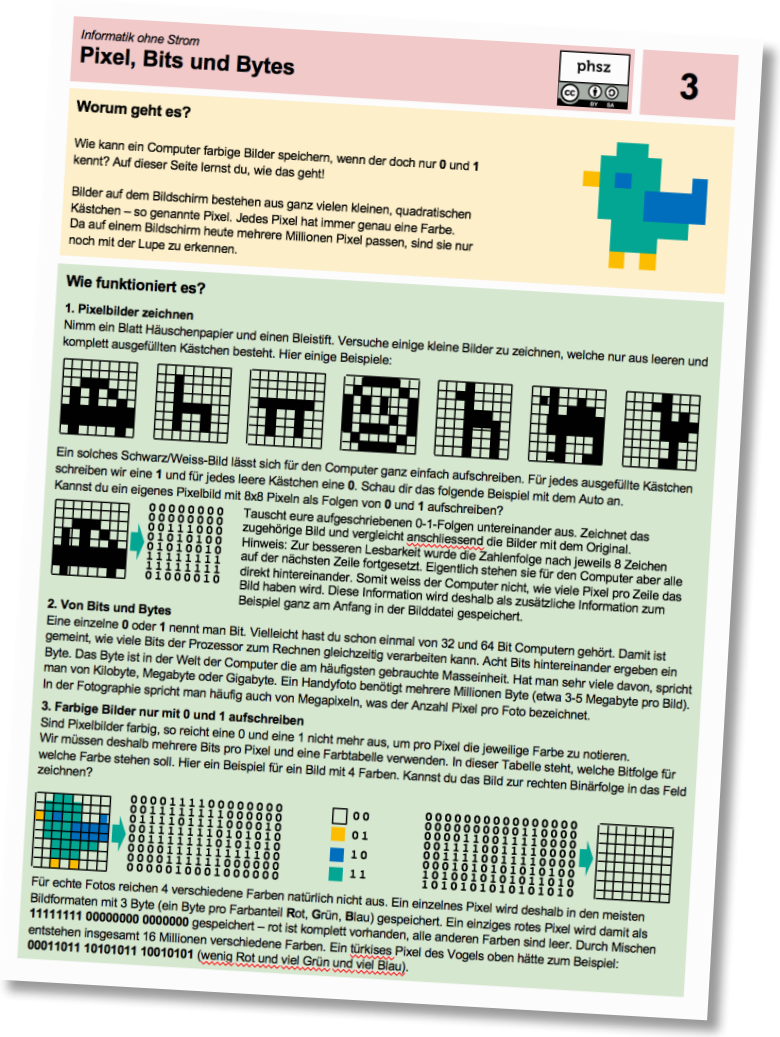
Zeichen	Anzahl Bits
a	8
ä,ö,ü	16
€	24
🚀	32



Codetabelle:

Zeichen	Code
S	0
I	11
P	101
M	100

Mit 0 und 1 geht nicht nur Text ...



Informatik ohne Strom

Pixel, Bits und Bytes

phsz

3

Worum geht es?

Wie kann ein Computer farbige Bilder speichern, wenn der doch nur **0** und **1** kennt? Auf dieser Seite lernst du, wie das geht!

Bilder auf dem Bildschirm bestehen aus ganz vielen kleinen, quadratischen Kästchen – so genannte Pixel. Jedes Pixel hat immer genau eine Farbe. Da auf einem Bildschirm heute mehrere Millionen Pixel passen, sind sie nur noch mit der Lupe zu erkennen.

Wie funktioniert es?

1. Pixelbilder zeichnen

Nimm ein Blatt Häuschenpapier und einen Bleistift. Versuche einige kleine Bilder zu zeichnen, welche nur aus leeren und komplett ausgefüllten Kästchen besteht. Hier einige Beispiele:

Ein solches Schwarz/Weiss-Bild lässt sich für den Computer ganz einfach aufschreiben. Für jedes ausgefüllte Kästchen schreiben wir eine **1** und für jedes leere Kästchen eine **0**. Schau dir das folgende Beispiel mit dem Auto an. Kannst du ein eigenes Pixelbild mit 8x8 Pixeln als Folgen von **0** und **1** aufschreiben?

Tauscht eure aufgeschriebenen 0-1-Folgen untereinander aus. Zeichnet das zugehörige Bild und vergleicht anschliessend die Bilder mit dem Original. Hinweis: Zur besseren Lesbarkeit wurde die Zahlenfolge nach jeweils 8 Zeichen auf der nächsten Zeile fortgesetzt. Eigentlich stehen sie für den Computer aber alle direkt hintereinander. Somit weiss der Computer nicht, wie viele Pixel pro Zeile das Bild haben wird. Diese Information wird deshalb als zusätzliche Information zum Beispiel ganz am Anfang in der Bilddatei gespeichert.

2. Von Bits und Bytes

Eine einzelne **0** oder **1** nennt man Bit. Vielleicht hast du schon einmal von 32 und 64 Bit Computern gehört. Damit ist gemeint, wie viele Bits der Prozessor zum Rechnen gleichzeitig verarbeiten kann. Acht Bits hintereinander ergeben ein Byte. Das Byte ist in der Welt der Computer die am häufigsten gebrauchte Masseinheit. Hat man sehr viele davon, spricht man von Kilobyte, Megabyte oder Gigabyte. Ein Handyfoto benötigt mehrere Millionen Byte (etwa 3-5 Megabyte pro Bild). In der Fotographie spricht man häufig auch von Megapixeln, was der Anzahl Pixel pro Foto bezeichnet.

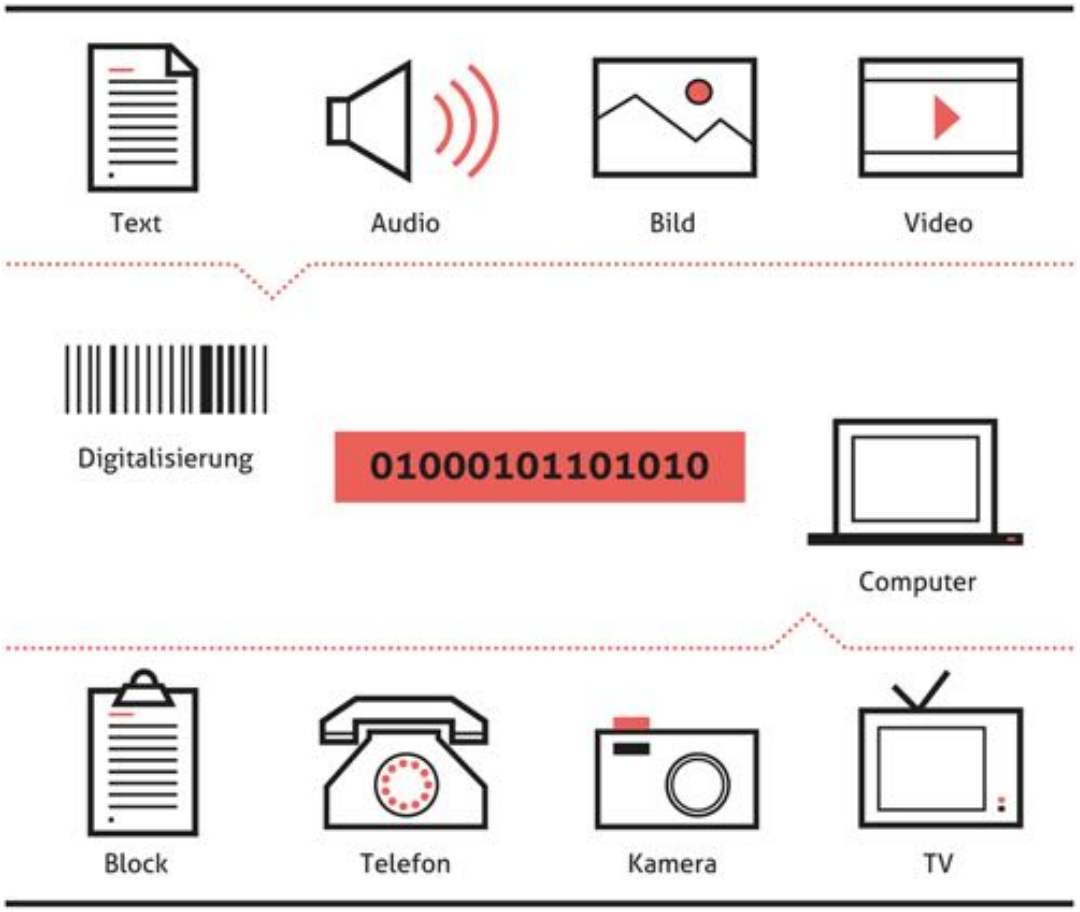
3. Farbige Bilder nur mit 0 und 1 aufschreiben

Sind Pixelbilder farbig, so reicht eine 0 und eine 1 nicht mehr aus, um pro Pixel die jeweilige Farbe zu notieren. Wir müssen deshalb mehrere Bits pro Pixel und eine Farbtabelle verwenden. In dieser Tabelle steht, welche Bitfolge für welche Farbe stehen soll. Hier ein Beispiel für ein Bild mit 4 Farben. Kannst du das Bild zur rechten Binärfolge in das Feld zeichnen?

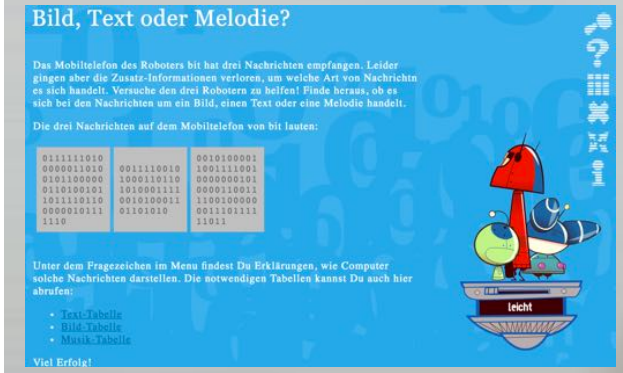
00 01 10 11

Für echte Fotos reichen 4 verschiedene Farben natürlich nicht aus. Ein einzelnes Pixel wird deshalb in den meisten Bildformaten mit 3 Byte (ein Byte pro Farbanteil **R**ot, **G**rün, **B**lau) gespeichert. Ein einziges rotes Pixel wird damit als **11111111 00000000 0000000** gespeichert – rot ist komplett vorhanden, alle anderen Farben sind leer. Durch Mischen entstehen insgesamt 16 Millionen verschiedene Farben. Ein türkises Pixel des Vogels oben hätte zum Beispiel: **00011011 10101011 10010101** (wenig Rot und viel Grün und viel Blau).

Wenn alles nur aus 0en und 1en besteht, woher weiss man dann eigentlich um was für Daten es sich handelt?

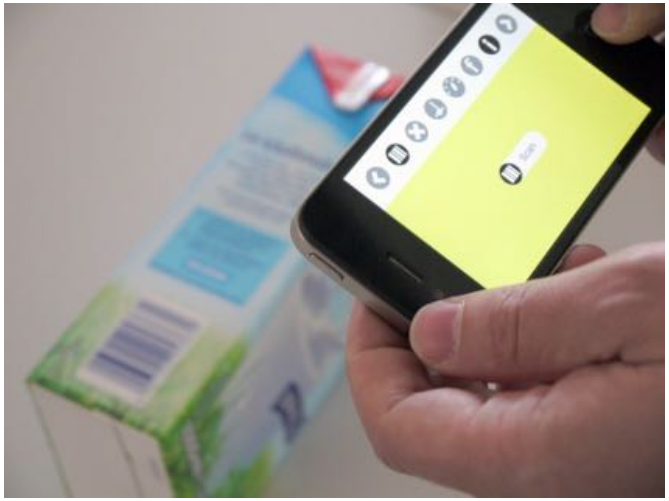


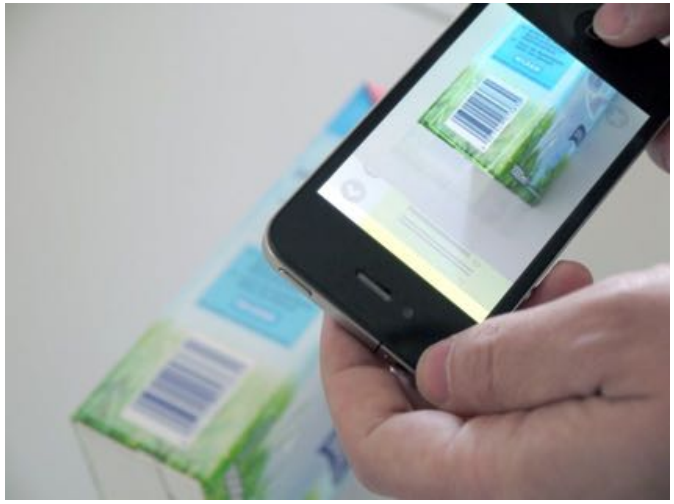
Es ist nicht immer ersichtlich ...

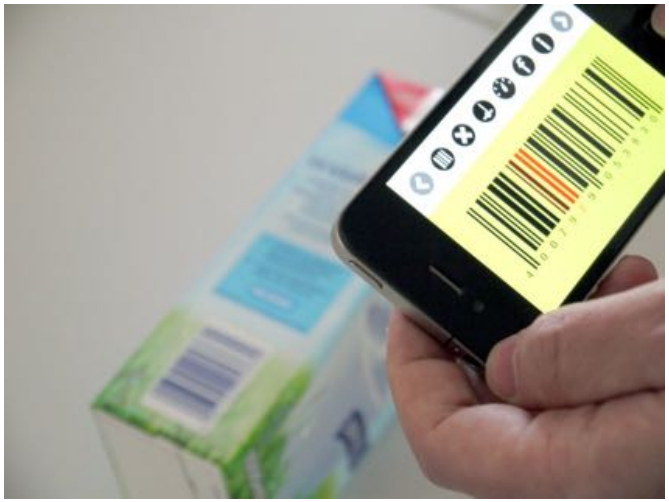


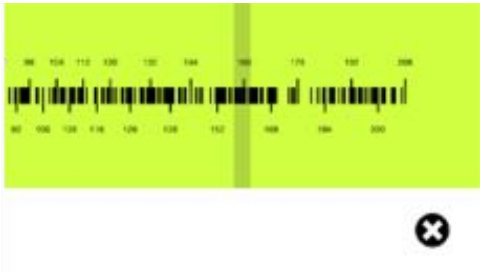
http://ilearnit.ch/de/1b/explore.html

Barcodas – barcode music generator – IOS-App









www.nr37.nl/?c=software