Grundbegriffe der Informatik

1. Begriff der „Informatik“

Definition seit den 60er Jahren: Wissenschaft von der maschinellen Informationsverarbeitung (engl. Computer Sciences)

a) **Theoretische Informatik**
mathematisch-logische Grundlagen aller Teilgebiete der Informatik, z.B. Compilerbau; Entwicklung der Software (= Definition einer Programmiersprache) orientiert sich an der Ökonomie mathematischer Theoriebildung: möglichst wenige Annahmen, möglichst keine Ausnahmen bzw. wenige Sonderfälle!

b) **Technische Informatik**
Konstruktion von Rechnern, Speicherchips und Prozessoren bzw. Peripherie (Elektrotechnik; Bereitstellung der Hardware)

c) **Praktische Informatik**
…schlägt die „Brücke“ zwischen Hardware und Anwendungssoftware, isb. Compilerbau. Der Compiler „übersetzt“ Programme, die in einer Programmiersprache (z.B. BASIC, Cobol, Fortran, Pascal, VB, C, C++, Java usw.) formuliert sind, in die Maschinensprache.

d) **d) Angewandte Informatik**
Einsatz von Rechnern in den verschiedenen Lebensbereichen: Textverarbeitung, Tabellenkalkulation, Datenbanken, Bildbearbeitung, Animationen, Roboter, Automaten, Maschinenbau, Navigationstechnik, CAD usw.

Beispiele zu den Teilgebieten der Informatik:

a) Die Theoretische Informatik befasst sich mit

- der abstrakten Beschreibung von Maschinen zur Informationsverarbeitung (die sog. *Automatentheorie*),
- der Beurteilung von Problemen hinsichtlich ihrer Berechenbarkeit (die sog. *Berechenbarkeitstheorie*),
- der Beurteilung von Algorithmen hinsichtlich ihrer Qualität (die sog. *Komplexitätstheorie*).

b) Die Technische Informatik befasst sich hauptsächlich mit der Hardware von Computern und Computernetzwerken. Dazu zählen:

- die Entwicklung von Computern mit allen Einzelteilen (die sog. *Mikroprozessortechnik*), einschließlich des Aufbaus von Prozessoren (die sog. *Rechnerarchitektur*),
- die Entwicklung von Netzwerktechnologien einschließlich der dafür benötigten Software (die sog. *Rechnerkommunikation*).

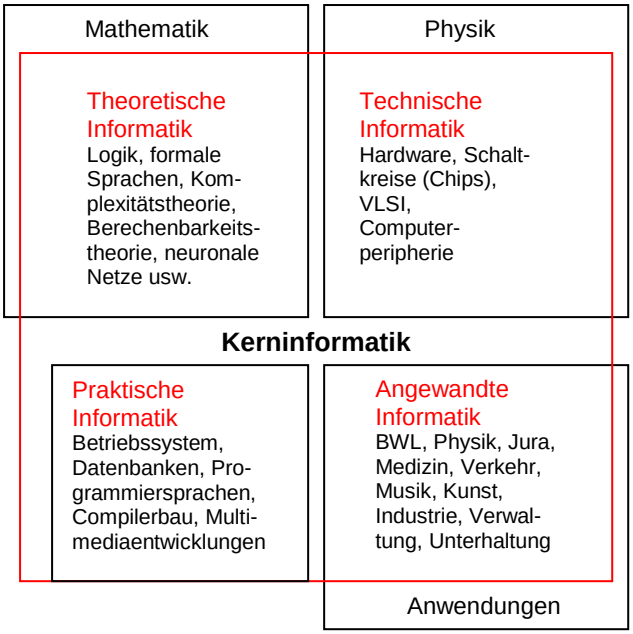
c) Die Praktische Informatik stellt Grundlagen bereit, die es überhaupt erst ermöglichen, mit Computern sinnvoll zu arbeiten:

- *Programmiersprachen* (Beispiel: C++ und entsprechende Compiler),
- *Algorithmen* (Beispiele: Methode zum Finden von Primzahlen),
- *Betriebssystemen* (Beispiele: Linux, Betriebssystem eines MP3-Players),
- *Datenbanken* (Beispiele: MySQL, Bilddatenbank von Google Earth).

© Dr. Walter Kirchgessner 2008

d) Die Angewandte Informatik befasst sich mit:

- *kaufmännischen Anwendungen* (Beispiele: Bürosoftware "Microsoft Office", Unternehmenssoftware "SAP R/3"),
- *wissenschaftlichen Anwendungen* (Beispiele: Computer-Algebra-System "Maple", Steuerung von Messgeräten mit "LabVIEW"),
- *technischen Anwendungen* (Beispiele: GPS-Empfänger, Steuereinheit einer Waschmaschine, CAD, DTP, Ampelanlagen, Autopilot usw.),
- Unterhaltungselektronik…



2. Allgemeine Grundlagen

a) Binärsystem (*Dualsystem*)

…arbeitet mit binären Informationseinheiten: „**Bit**“ (von „Binary Digit“)
→ 1 Bit: Möglichkeit zweier Antworten!

Mit elektr. Rechenanlagen fanden die Binärzahlen ihre Anwendung

Früher bei der Lochkarte: entweder Lochung oder keine.
Beim Rechner: entweder Strom oder keiner.

„Strom“ oder „kein Strom“ kann also eine Information über folgende Inhalte bedeuten:
Buchstaben, Sonderzeichen, Ziffern, Steuerzeichen, Messwerte, Bilder und Grafiken.

Beispiel: In einem Loch liegt **eine** Kugel - In einem Loch liegt **keine** Kugel.
Für diese zwei Informationen genügen die Ziffern 0 und 1.

Ein Bit genügt nur bei Fragen, die **zwei** Antwortmöglichkeiten zulassen:
ja/nein, wahr/falsch, schwarz/weiß, hell/dunkel, groß/klein, stark/schwach, links/rechts usw.
Bei einer Frage mit **vier** Antwortmöglichkeiten (z.B. Himmelsrichtungen) werden zwei Bits benötigt; bei den Zwischenrichtungen (z.B. Nordost) werden die nötigen Bit-Kombinationen aus drei Bits erzielt: 000, 001, 010, 011, 100, 101, 110, 111

Es gibt genau 2^N mögliche Bitfolgen, die Werte in den einzelnen Spalten sind also die Potenzen von zwei:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	**64**	**32**	**16**	**8**	**4**	**2**	**1**

© Dr. Walter Kirchgessner 2008

Die achtwertige Dezimalzahl **2 4 3 1 5 3 1 2** ergibt sich aus
$2*10^7 + 4*10^6 + 3*10^5 + 1*10^4 + 5*10^3 + 3*10^2 + 1*10^1 + 2*10^0$

Die achtwertige Binärzahl **1 1 0 1 1 0 0 1** bedeutet dem Dezimalsystem nach:
$\mathbf{1}*2^7 + \mathbf{1}*2^6 + \mathbf{0}*2^5 + \mathbf{1}*2^4 + \mathbf{1}*2^3 + \mathbf{0}*2^2 + \mathbf{0}*2^1 + \mathbf{1}*2^0 =$
128 64 0 16 8 0 0 1

0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010
sind demnach die Dezimalzahlen von Null bis zehn im Binärsystem.

$= 0*2^0 , 1*2^0 , 1*2^1 + 0*2^0 , 1*2^1 + 1*2^0$ usw.
$= 0 \quad 1 \quad 2 + 0 \quad 2 + 1$

Mit Bezug auf ein einzelnes Bit, das an sich nur zwei Zustände (0 oder 1, wahr oder falsch usw.) bedeuten kann, lassen sich mit zwei Bits also schon vier Zustände, Kombinationen oder Varianten darstellen, mit drei Bits acht, mit vier Bits 16 usw. usw.

1. Bit 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
2. Bit 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1
3. Bit 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1
4. Bit 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

Jeder dieser Bit-Kombinationen kann man dann z.B. einen Buchstaben zuordnen und erhält bereits sechzehn Zeichen unseres Alphabets.

Mit 8 Bits lassen sich nun 256 Kombinationen erzielen, mit denen sich z.B. das ganze Alphabet (inklusive Groß- und Kleinschreibung und Sonderzeichen) darstellen lässt.

Eine Gruppe aus 8 Bits nennt man „**Byte**“
(z.B. Arbeitsspeicher: in MB: 1 MB = 1.024 kb = 1.048.576 Bytes = 8.388.608 Bits)

Beispiel aus ASCII-Code: Beim Drücken der Taste A wird an den Rechner ein Maschinenbefehl mit dem Byte 0 1 0 0 0 0 0 1 gesandt. Für die Datenausgabe wird der Binärcode wieder zurückverwandelt.

Um die Bitmuster in der Maschinensprache aber noch einfacher darzustellen, wird häufig das folgende Zahlensystem verwendet:

b) Hexadezimalsystem *(Hexagesimalsystem, Sedezimalsystem)*

Bei der „Verständigung“ eines PCs z.B. mit seinem Drucker schickt der Rechner die Befehle in Form von Zahlen an den Drucker. Diese werden meist im „Sechzehner-System“ verschlüsselt.

Es hat 16 Ziffern (statt nur 10 wie beim Dezimalsystem):
1 bis 0, also zehn, dazu vom Alphabet die Zeichen A bis F.

Zeichenfolge	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Hex-Zeichen	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
num. Wert	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Zur Darstellung eines 8-Bit-Zeichens fasst man im HEX-Code jeweils 4 Bit zu einem Halbbyte (Nibble) zusammen.

Jedes Zeichen des HEX-Codes steht also für 4 Bits:

© Dr. Walter Kirchgessner 2008

Beispiel 1

F → 1111 → 8+4+2+1 = 15 [Dez]

Beispiel 2

65 → 110 | 0101 → 4+2+0 | 0+4+0+1 → **6** x 16^1 + **5** x 16^0 = 101 [Dez]

Jede mögliche Bit-Kombination eines Byte-Zeichens kann also mit 2 HEX-Zeichen dargestellt werden. Den Zahlenwert einer Folge von HEX-Ziffern erhält man, indem man jede Ziffer entsprechend ihrer Ziffernposition mit der dazugehörigen Potenz der Basiszahl 16 multipliziert und die Ergebnisse aufsummiert:

Wie die Dezimalzahl **327** für den Zahlenwert
3 x 10^2 + **2** x 10^1 + **7** x 10^0 steht,

repräsentiert die HEX-Zahl **1AF3**

1 x 16^3 + **A** x 16^2 + **F** x 16^1 + 3 x 16^0 =
1 x 4096 + 10 x 256 + 15 x 16 + 3 x 1 = **6899**

Die HEX-Darstellung wird oft von Assembler-Programmierern vorgezogen.

c) ASCII-Code (s. Anhang)

Dem ASCII-Code (s. Anlage) zufolge mit ursprünglich nur 7 Bits, also 128 mit darstellbaren Zeichen würde das Wort „Code“ so dargestellt werden:

C	**o**	**d**	**e**	= alphabetisch
100 0011	110 1111	110 0100	110 0101	= Binär-Code (7-Bit)
43	6F	64	65	= HEX-Code

Der 7-Bit-ASCII berücksichtigt allerdings weder die deutschen Umlaute noch das „ß“. Der acht Bit wird als Prüf- oder Steuerbit angehängt (**ANSI-Code** für Windows mit 256 Zeichen).

d) Unicode

Der Unicode (seit 1991) verwendet 16 Bits und kann alle Zeichen aller geschriebenen Weltsprachen darstellen.

e) Dateien

Eine Datei (Text, Zahl, Bild, Musik, Programme usw.) ist eine beliebig lange Folge von gespeicherten Bytes. Ihre Größe bezeichnet die Anzahl der in ihr enthaltenen Bytes (B):

k = 1024 = 2^{10} (k = **kilo**)
M = 1024 x 1024 = 2^{20} (M = **mega**)
G = 1024 x 1024 x 1024 = 2^{30} (G = **giga**)
T = 1024 x 1024 x 1024 x 1024 = 2^{40} (T = **tera**)

f) Speichergrößen

Beispiele:
eine Notiz: ~ 200 B, ein Brief: ~ 3 kB, ein DOS-Programm: ~ 300 kB
Diskettenkapazität: ~ 1,4 MB, ein Windows-Programm: ~ 5 MB
ein Musiktitel: ~ 40 MB im Wave-Format, 4 MB im mp3-Format
Hauptspeichergröße: ~ 512 MB
DVD: ~ 4,7 GB - 17 GB, Festplatte: ~ 160 GB

© Dr. Walter Kirchgessner 2008

Anhang: ASCII-Tabelle

Höherwertiger Teil → / Niederwertiger Teil ↓					0	0	0	0	1	1	1	1
					0	0	1	1	0	0	1	1
					0	1	0	1	0	1	0	1
					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	¤	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	´	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	A	LF	SUB	*	:	J	Z	j	z
1	0	1	1	B	VT	ESC	+	;	K	[	k	{
1	1	0	0	C	FF	FS	,	<	L	\	l	¦
1	1	0	1	D	CR	GS	–	=	M	]	m	}
1	1	1	0	E	SO	RS	.	>	N	^	n	~
1	1	1	1	F	SI	US	/	?	O	_	o	DEL

Dual-Code

HEX-Code

© Dr. Walter Kirchgessner 2008