



Claudio Franzetti

Essenz der Informatik

EBOOK INSIDE

 **Springer** Vieweg



Essenz der Informatik

Claudio Franzetti

Essenz der Informatik

Claudio Franzetti
Meilen, Schweiz

ISBN 978-3-662-58533-7 ISBN 978-3-662-58534-4 (eBook)
<https://doi.org/10.1007/978-3-662-58534-4>

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Springer Vieweg

© Springer-Verlag GmbH Deutschland, ein Teil von Springer Nature 2019

Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Der Verlag, die Autoren und die Herausgeber gehen davon aus, dass die Angaben und Informationen in diesem Werk zum Zeitpunkt der Veröffentlichung vollständig und korrekt sind. Weder der Verlag, noch die Autoren oder die Herausgeber übernehmen, ausdrücklich oder implizit, Gewähr für den Inhalt des Werkes, etwaige Fehler oder Äußerungen. Der Verlag bleibt im Hinblick auf geografische Zuordnungen und Gebietsbezeichnungen in veröffentlichten Karten und Institutionsadressen neutral.

Springer Vieweg ist ein Imprint der eingetragenen Gesellschaft Springer-Verlag GmbH, DE und ist ein Teil von Springer Nature

Die Anschrift der Gesellschaft ist: Heidelberger Platz 3, 14197 Berlin, Germany

Nichts ist vom Erfolg her zweifelhafter und von der Durchführung her gefährlicher als der Wille, sich zum Neuerer aufzuschwingen. Denn wer dies tut, hat die Nutznießer des alten Zustandes zu Feinden, während er in den möglichen Nutznießern des neuen Zustandes nur lasche Verteidiger findet.

Nicolò Machiavelli

Je planmäßiger die Menschen vorgehen, desto wirksamer vermag sie der Zufall zu treffen.

Friedrich Dürrenmatt

Nullus propheta in patria: Der Prophet gilt nichts im eigenen Land.

Spruchwort

Vorwort

Diese Buch richtet sich an Anwender und Benutzer, die ich einerseits als Schüler und Lehrer der Sekundarstufe und andererseits als typische Mitarbeiter von Nicht-IT-Unternehmen verstehe. Das Publikum ist also sehr breit.

Meine Kinder sind etwa in diesem Alter. Ich muss allerdings feststellen, dass sie zwar ein Fach „Informatik“ belegen aber dass sich die Themen eher am Maschinenschreiben orientieren. Bei Gruppenarbeiten (kollaboratives Arbeiten) oder beim Vokabellernen, z.B. am Smartphone, vermisse ich jede effiziente IT-Unterstützung.

In meiner aktuellen Firma höre ich ständig vonseiten der Informatiker, die Mitarbeiter verstünden sie nicht, man würde aneinander vorbeireden. Bei einer Funktionsstörung will der Benutzer diese so schnell wie möglich behoben sehen, die Erklärungen werden erduldet und nicht genutzt, sich weiterzubilden.

Das wahrgenommene Desinteresse wird als Inkompetenz interpretiert. Dadurch verringert sich die Anstrengung, systematisch zu informieren. Dadurch erhöhen sich die widersprüchlichen Mitteilungen und verstärken die unklaren Äußerungen der Benutzer. Dadurch erhöht sich die wahrgenommene Inkompetenz der Benutzer durch die Techniker ... und weiter so.

Es scheint mir äußerst wichtig, den Mitarbeitenden die Möglichkeit zu eröffnen, sich stressfrei ein wenig Informatikwissen anzueignen und damit Anhaltspunkte zu gewinnen, um mit den Technikern zu reden.

Besonders in kleineren Firmen ist die Wahrscheinlichkeit groß, als Fachperson mit IT-Entwicklern kommunizieren und in Projekten zusammenarbeiten zu müssen. Deshalb ist es wichtig, gewisse Konzepte, Prinzipien und gewisse Zusammenhänge verstehen zu können.

Somit habe ich die Gelegenheit wahrgenommen, diese Broschüre herauszugeben. Wie in der IT üblich, ist dem Vorhaben ein „Make-or-Buy“-Entscheidung vorangegangen. Ich habe das Angebot studiert, wobei auch und vor allem Publikationen für den Schulgebrauch im Brennpunkt standen. Keines hat mich davon überzeugt, kein eigenes Buch schreiben zu müssen. Deshalb habe ich die Herausforderung – *façon de parler* – angenommen.

Die Qualifikation für den Inhalt dieses Buches habe ich mir in langjähriger Beschäftigung mit Computern redlich erworben: Anfang der Achtzigerjahre im Ingenieurstudium an der ETH mit Einführung in Pascal und Rechnen auf der CDC7600 mit dem exotischen Betriebssystem „Venus“, dann als Forschungsingenieur mit Strömungssimulationen in Fortran auf der IBM 7600VF, Vax 3000, Silicon Graphics IRIS und dem damaligen Superrechner Cray-1 in Lausanne. Als Finanzfachmann entwickelte ich Datenbanken und Derivatepreise für Banken. In höheren Führungspositionen blieb IT mehr als ein Hobby. Wieder bei einer Kleinfirma als Geschäftsführer entwickelten wir Software für die Versicherungsindustrie. Und schließlich war ich für die IT-Entwicklung in der jetzigen Firma verantwortlich.

Immer war es mein Ziel, produktive Systeme einzuführen, beispielsweise das Wiki „Confluence“ oder das Issue-Tracking namens „Jira“.

Es ist mein lebhafter Wunsch, dass diese Schrift zu Diskussionen Anlass geben wird. Denn dies ist ein Zeichen, dass es gebraucht wird.

Im Vorspann sind mehrere Zitate aufgeführt. Die einen sollen Skepsis bezüglich der Planbarkeit von IT-Entwicklungen ausdrücken, denn meiner Meinung nach sind diese ingenieurmäßigen Vorgehensweisen nicht immer problemadäquat. Dies wird in einem Kapitel thematisiert. Das Machiavelli-Zitat soll auf die Schwierigkeit hinweisen, die IT-Verantwortliche begegnen, wenn sie ihrer natürlichen Rolle als Neuerer nachkommen. IT-Entwicklung ist ein riskantes Geschäft.

Der Text ist gespickt mit Beispielen und Illustrationen. Hier versuche ich, möglichst keine proprietären Systeme zu zeigen sondern auf frei verfügbare Ressourcen zu setzen. Damit kann die oder der Lesende selber Methoden nachvollziehen, damit spielen oder erweitern.

Der Text ist nicht primär darauf ausgelegt, das Programmieren zu vermitteln. Es möchte aber grundlegende Sachverhalte anhand einer Blocksprache wie Scratch aufzeigen, die dem Publikum den Einstieg in die Vertiefung ermöglichen. Die Absicht geht also viel mehr in die Breite als in die Tiefe.

Einige Kapitel sind mit einem * versehen. Diese sind im Vergleich zum Haupttext möglicherweise etwas anspruchsvoller oder setzen gewisse Kenntnisse voraus, die für die IT aber nicht ausschlaggebend sind. Abschnitte mit † sind für Mitarbeiter gedacht und weniger für Schüler, die den betrieblichen Alltag noch nicht kennen.

Zuletzt möchte ich allen danken, die an dieser Publikation mitgewirkt haben. Mein spezieller Dank geht an meinen Kollegen Dipl.-Ing. Claus Frauenheim, der sowohl gute inhaltliche Anregungen gegeben als auch das Manuskript genau gelesen hat. Fehler sind ausschließlich dem Autor anzulasten.

Inhaltsverzeichnis

Abbildungsverzeichnis	xiii
Tabellenverzeichnis	xvii
1 Einführung	1
Literatur	2
2 Ganz kurze Geschichte der IT	3
2.1 Rechner	3
2.2 Software	7
Literatur	9
3 Alles ist Zahl	11
3.1 Zweier- oder Binärsystem	11
3.2 Sechzehner-System	13
3.3 Gleitkomma-Zahlen*	14
3.4 Rohe Daten, Alphabete	14
3.5 Komprimierung und Verifizierung	15
3.6 Einfache Formallogik*	19
3.7 Rechenschaltungen*	20
Literatur	24
4 Computer als Hardware	25
4.1 Zentrale Einheiten	26
4.2 Periphere Geräte, Eingabe und Ausgabe	31
4.3 Smartphones	32
4.4 Sensoren, IoT	34
Literatur	37
5 Betriebssystem und Benutzeroberfläche	39
5.1 Operating Systems	40
5.2 Oberfläche – Benutzerkommunikation	41

5.3	Benutzer-Interaktion	41
5.4	Dateiverwaltung – Filesystem	42
5.5	Geräteverwaltung	44
5.6	Virtuelle Maschinen	45
5.7	Eingebettete Systeme	46
6	Information und Medien	49
6.1	Kommunikation	49
6.2	Desinformation, Propaganda, Fake-News	54
6.3	Soziale Medien	56
6.4	Medienkompetenz	57
	Literatur	63
7	Datenorganisation	65
7.1	Datentypen und -strukture	66
7.2	Versionierung	67
7.3	Relationale Datenbanken	68
7.4	NoSQL-Datenbanken	73
7.5	Markup-Sprachen	74
7.6	Datensicherung, Backup	84
	Literatur	85
8	Netzwerke	87
8.1	Typen	88
8.2	Client-Server	89
8.3	Schichtenmodell und Protokolle	92
8.4	Internet, World-wide-Web	93
8.5	Webservices, APIs	95
8.6	Deepweb und Darknet	96
	Literatur	99
9	Programmierung	101
9.1	Voraussetzungen	103
9.2	Philosophie	105
9.3	Compiler und Interpreter	106
9.4	Einfache Blocksprache – Einführung Programmieren	107
9.5	Hoch-Sprachen	118
9.6	Skript-Sprachen	128
9.7	Und Tabellenkalkulatoren?	131
9.8	Umgebung und Tools	132
9.9	Architekturmuster	136
	Literatur	138

10 Applikationen	139
10.1 Systematik	139
10.2 Office: Schreibmaschine 4.0	140
10.3 Generische Anwendungen	144
10.4 Geschäfts-Anwendungen [†]	154
10.5 Hobby-Anwendungen	159
10.6 Gaming	161
10.7 Simulation	165
Literatur	169
11 Sicherheit	171
11.1 Verschlüsselung	171
11.2 Identität und Echtheit	172
11.3 Zugang	176
11.4 Hacking und Malware	179
Literatur	182
12 IT-Organisation[†]	183
12.1 Strategie der IT	183
12.2 Management der Informationssysteme	185
12.3 IT-Service-Management	187
Literatur	190
13 Cloud-Computing	191
13.1 Infrastruktur	193
13.2 Plattform	194
13.3 Services	194
13.4 Blockchain	195
14 IT-Entwicklung	203
14.1 Herausforderungen	203
14.2 Geschäftsprozesse und IT-Entwicklung	205
14.3 Problemadäquate Methoden	207
14.4 Monolithisch versus Service	208
14.5 Planmethoden, Wasserfall	209
14.6 Agile Methoden	212
14.7 Diskussion	215
Literatur	220
15 Digitalisierung, digitale Transformation	223
15.1 Ökonomische Transformation [†]	224
15.2 Einfluss auf verschiedene Bereiche	229
Literatur	240

16 Künstliche Intelligenz	241
16.1 Roboter	242
16.2 Virtual und Augmented Reality	243
16.3 Arten von Künstlicher Intelligenz	246
16.4 Algorithmen und Machine-Learning	250
16.5 Super-Intelligenz	261
Literatur	262
17 IT und Gesellschaft	263
17.1 Soziale Aspekte	263
17.2 Ökonomische Aspekte	272
17.3 Ökologische Aspekte	275
17.4 Technologische Aspekte	277
Literatur	284
A Installation Scratch	287

Abbildungsverzeichnis

1.1	Übersicht Geschäfts-IT	1
2.1	Curta-Rechner	4
2.2	UNIVAC I Röhrenrechner	5
2.3	Personal Computer von IBM	7
3.1	Datei-Erkennung	13
3.2	Lydische Schrift	15
3.3	PNG-Datei im Editor	15
3.4	Kontaktschaltungen	21
3.5	Addierer-Schaltungen	22
3.6	XOR-Schaltung	22
3.7	4-Bit-Volladdierer	23
3.8	IC-Gehäuse	23
4.1	Modell Zentraleinheit	26
4.2	Diode und Transistor	27
4.3	AND/NAND-Schaltung	28
4.4	RAM-Speicher	30
4.5	CPU-RAM-Verbindung	30
4.6	Modell Zentraleinheit	30
4.7	Offenes Smartphone	33
5.1	Schichtenmodell	39
5.2	Betriebssysteme	40
5.3	Anteile der Betriebssysteme	41
5.4	Eingabeaufforderung	42
5.5	Desktop-Bildschirm	43
5.6	Die drei typischen Eingabe-Akteure.	44
5.7	Die zwei Typen von virtuellen Maschinen.	46

6.1	Selbstvertrauen und Wissen	52
6.2	Das Kommunikationsmodell von Shannon, erweitert.	52
6.3	Benutzer-Rennliste der Sozialen Medien	57
7.1	Ablagestruktur des Betriebssystem Windows	66
7.2	Vierstufiger Baum	67
7.3	Textverarbeitung in der Cloud	68
7.4	Wiki und Versionierung	68
7.5	Nicht normalisierte Verknüpfung.	71
7.6	Auflösung einer N:M-Beziehung	71
7.7	Zuordnungstabelle bei N:M-Beziehung.	72
7.8	ER-Diagramm zum Bibliothekenverbund	72
7.9	Selbstreferentielle Tabelle.	73
7.10	Quellcode einer japanischen HTML-Seite	77
7.11	Ausschnitt aus ShareLatex	82
7.12	Schema einer betrieblichen Datensicherung.	85
8.1	Das ARPANET ganz zu Beginn im Jahr 1969.	87
8.2	ARPANET im Jahr 1977	88
8.3	Schematische Darstellung von drei Clients mit einem Server. .	89
8.4	Client-Server-Architekturen	90
8.5	Zeitverlauf zur Client-Server-Architektur.	91
8.6	OSI-7-Schichtenmodell	92
8.7	TCP/IP-Schichten	93
8.8	Verpackungsmodell	94
8.9	Domain-Namen	95
8.10	Struktur des Webs.	98
9.1	Schritte der Problemlösung	102
9.2	Abstraktion als Brücke.	104
9.3	Compiler und Linker	107
9.4	Flussdiagramm zum Programm rtbis.	121
9.5	Darstellung der Simulation mit Scratch.	128
9.6	VBA-Programmierungsumgebung.	132
9.7	Beispiel Atom-IDE	134
9.8	Beispiel Anwendungsfalldiagramm	135
9.9	Beispiel Sequenzdiagramm	135
9.10	Beispiel Zustandsdiagramm	136
9.11	Zustandsdiagramm: Theologie.	136
9.12	Übersicht MVC	137
10.1	Synopsis Software	139
10.2	Typisches Aussehen eines Tabellenkalkulations-Programms .	141
10.3	Typisches Aussehen eines Textprogramms	141
10.4	Spracherkennung von Google Drive	142
10.5	Spracherkennung auch in Fremdsprachen.	142

10.6	Formatvorlagen für Word in der Entwurfsansicht.....	143
10.7	Vergleich von Raster- und Vektordarstellung	144
10.8	Screenshot Business-App.	145
10.9	Die Architektur eines Browsers.	146
10.10	Einstellungen des Opera-Browsers.....	147
10.11	Desktop-Publishing mit der freien Software „Scribus“.....	155
10.12	Editieren einer Pdf-Datei mit Okular.....	156
10.13	Zustandsdiagramm	157
10.14	Manipulierte Fotografie.....	160
10.15	Spielereien an einem Foto mit GIMP	161
10.16	Benutzer-Interface von Audacity, einem Audio-Editor.	162
10.17	Ausschnitt aus dem User Interface von LMMS.....	162
10.18	Beschleunigtes Rechnen dank der Grafikkarte.....	163
10.19	Raumgleiter Hermes.	165
10.20	Die CRAY-1 von Lausanne.....	166
10.21	Monte Carlo Simulation	168
11.1	Asymmetrische Verschlüsselung.....	173
11.2	Daten und Identität	175
11.3	Signieren von Dokumenten.....	176
11.4	Firewall	178
12.1	Überblick FitSM	188
13.1	Cloud-Computing	191
13.2	Cloud Konfiguration.	192
13.3	Hashfunktion	196
13.4	Ein Merkle-Baum oder Hash-Tree	197
13.5	Blockchain	198
13.6	Mining-Algorithmus als Flowchart.	199
14.1	Wesentliche Funktionen für die IT-Entwicklung.	204
14.2	Ein BPMN-Prozess	205
14.3	Ein BPMN-Prozess	208
14.4	Verfahrensmethoden	208
14.5	Phasenkonzept der NASA	210
14.6	Projekt-Abweichungsgrafik	212
14.7	Phasenmodell von Siemens	212
14.8	Übersicht Projektmanagement	213
14.9	Backlog	214
14.10	Backlog im Issue-Tracker	215
14.11	Beispiel Backlog.	216
14.12	Zielerreichung nach Methode.....	217
14.13	Unterschiedliche Ziele der zwei Verfahrensmethoden	219
15.1	Technologieverlauf.....	224

15.2 Digitale Transformation von Unternehmen.....	225
15.3 Verteilte Buchhaltung im Vergleich	227
15.4 Transaktion mit Bitcoin.	228
15.5 Alte Aktenvorschrift.	231
15.6 Geoinformationssystem mit Lärmbelastung.....	231
15.7 Digitalisierte Bücher, Beispiel.	235
15.8 Ein möglicher Patientenpfad.	237
16.1 Kleiner Lernroboter Ozobot.	243
16.2 Realität als Kontinuum.	244
16.3 Video-Anprobe einer Brille, Beispiel (Foto: C. Franzetti).	245
16.4 AR-Tourismus-Applikation	246
16.5 Frühe VR-Brille.	246
16.6 Die Grenze zwischen Mensch und Maschine.	248
16.7 Maschinelles Lernen im Kontext.....	250
16.8 Lernen in zwei Kontexten	251
16.9 Koenigsberger Brücken.....	253
16.10 Lernen in zwei Kontexten	254
16.11 Künstliches Chromosom.....	255
16.12 Crossing-Over.....	255
16.13 Generelle Fragestellung als Black-Box.	256
16.14 Künstliches Neuronales Netz.	257
16.15 Aktivierungsfunktionen	258
16.16 Input-Probieren.	258
16.17 Körpererkennung	259
16.18 Neuronales Netz zum Probieren.....	260
17.1 MOOC-Angebote.....	265
17.2 Profiling	268
17.3 Spuren von Fitness-Geräten	271
17.4 Moore'sches Gesetz.	278
17.5 Big-Data schematisch	279
17.6 Auto als Datenquelle	280
17.7 Quantenrechner-Architektur	281
A.1 Anmeldung Scratch.	287
A.2 Entwicklungsumgebung von Scratch.....	288

Tabellenverzeichnis

3.1	Bitmuster und Interpretation	12
5.1	Treiberliste	44
7.1	Einfache Datentypen.....	67
8.1	Die wichtigsten Arten von Servern.....	90
9.1	Drei Fischobjekte von Scratch	127
10.1	Die schnellsten Computer der Welt	167
13.1	Die 10 größten Cloudanbieter.....	191
14.1	Projektsituationen.....	210
14.2	Erfolgsquote nach Verfahrensmodell	216
14.3	Das Bimodalmodell von Gartner	219
17.1	Zukunft der Berufe	283

Kapitel 1

Einführung

Zu Beginn eine Begriffsklärung (oder das pure Gegenteil): Im Titel dieser Broschüre steht „Informatik“. Viele denken aber automatisch an „IT“, das für Informationstechnologie steht aber auch „Informationstechnik“ abkürzt. In letzter Zeit hat sich noch der Begriff „IKT“ für Informations- und Kommunikationstechnologie eingeschlichen.

Wir finden, dass die Kommunikation, das Zur-Verfügung-Stellen und Übermitteln von Informationen, eigentlich im Überbegriff Information schon enthalten ist. Eine Definition von Kommunikation lautet: Austausch von Information über Netzwerke.

Der Unterschied zwischen Technik und Technologie ist der Unterschied zwischen Anwendung (Technik) und Forschung, Entwicklung und Wissenschaft (Technologie). Letztere kann man wiederum mit Informatik, auf englisch „Computer Science“ gleichsetzen.

Da man auf dem Papier eher Theorie denn Praxis üben kann, sollte der Begriff Informatik eigentlich recht gut passen. Denn es soll der Zweck dieser Broschüre sein, ein paar grundlegende Tatsachen in Erinnerung zu rufen, die für den täglichen Gebrauch der Informationstechnik hilfreich sein können. Zudem soll sie helfen, das Verständnis zwischen IT-Spezialisten und IT-Anwendern zu verbessern.

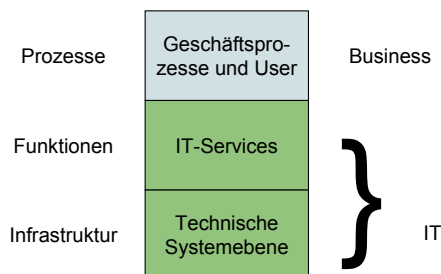


Abb. 1.1 Übersicht Geschäfts-IT. Die IT kann unterstützend oder gestaltend sein. Zweites ist besonders in disruptiven Zeiten der Fall, in denen der Fortschritt das Business bestimmt.

Die Abbildung 1.1 zeigt schematisch den Zusammenhang zwischen Geschäftstätigkeit und IT, wobei letztere in Infrastruktur und Funktionen oder Dienste aufgeteilt ist. Die Schnittstelle zwischen Business und IT ist eine Problemzone. Denn bei der Weiterentwicklung der Dienste sind es ja Menschen, die miteinander eine Lösung erarbeiten müssen, auch wenn sie in ganz unterschiedlichen Modellen und Vorstellungen geschult worden sind.

Dieses Hindernis kann man durch Kennenlernen der jeweils anderen „Welt“ senken. Für die Geschäftsseite heißt dies, sich über IT einen gewissen Überblick zu verschaffen.

Im Marketing gilt der Spruch: „People don’t know what they want until they know what they can get.“¹ Dies gilt häufig auch für die Geschäftsseite einer Unternehmung. Meistens bleibt man beim Bekannten haften. Vom Autopionier Henry Ford ist der Spruch überliefert: „Wenn ich die Menschen gefragt hätte, was sie wollen, hätten sie gesagt: schnellere Pferde.“ Neue Situationen, neue Technologien, neue Geschäftsmodelle sind problematisch. Der Physiker Max Planck sagte über die Physik (Planck 1955, 22):

Eine neue wissenschaftliche Wahrheit pflegt sich nicht in der Weise durchzusetzen, daß ihre Gegner überzeugt werden und sich als belehrt erklären, sondern vielmehr dadurch, daß ihre Gegner allmählich aussterben und daß die heranwachsende Generation von vornherein mit der Wahrheit vertraut gemacht ist.

Wer sich mit Innovation im Geschäftsumfeld befasst, muss die Persönlichkeit besitzen, gegen Widerstände anzukämpfen. Schon Niccolò Machiavelli, ein Philosoph und Dichter, hat im fünfzehnten Jahrhundert die Problematik der Innovatoren beschrieben, wie es im Zitat von Seite v angedeutet ist.

Literatur

Planck, M. (1955). *Wissenschaftliche Selbstbiographie mit der von Max von Laue gehaltenen Traueransprache*. Lebensdarstellungen deutscher Naturforscher. J.A. Barth, Berlin.

¹ Die Leute wissen erst was sie wollen, wenn sie erkennen, was sie haben können.



Kapitel 2

Ganz kurze Geschichte der IT

Die Geschichte der IT ist aus einer technologischen Sicht vor allem die Geschichte der Computer (Rechner, Hardware) und der Programme (Software). Notwendige Vorarbeiten in Mathematik und Logik waren etwa die Zahlendarstellung im Zweiersystem und die entsprechende Algebra der Logik von George Boole. Diese ist ebenfalls zweiwertig, denn Aussagen sind entweder wahr oder falsch. In den Dreißigerjahren stellte Alan Turing ein vollständiges (theoretisches) Modell eines Universalrechners vor.

An der Zweiteilung in Hard- und Software halten wir fest.

2.1 Rechner

Das Rechnen, bestehend aus den vier Grundrechenarten Addition, Subtraktion, Multiplikation und Division, wurde schon seit Tausenden von Jahren im Osten und im Westen durch den Zählrahmen oder Abakus unterstützt. Mit der Einführung der arabischen Zahlen, die allerdings aus Indien stammen, und der Erfindung der Logarithmen im 17. Jahrhundert kam der Rechenschieber auf. Die nächste Etappe bildeten die mechanischen Rechenmaschinen, deren erste von Wilhelm Schickard 1632 entwickelt wurde, gefolgt von der Pascaline von Blaise Pascal von 1642. 1676 legte Gottfried Wilhelm Leibniz seinen Rechner mit der Staffelwalze vor. Vor allem Uhrmacher wie Jean-Baptiste Schwilgué vom Straßburger Dom besaßen die feinmechanischen Fertigkeiten, um komplizierte Rechner (und Astrolabien) zu bauen. Um 1850 wurde der erste mechanische Rechner von Charles Xavier Thomas industriell gefertigt. Der letzte mechanische Rechner ist die Curta von Curt Herzstark, der bis 1945 gebaut wurde und ein 15-stelliges Resultatwerk aufwies. Er sah wie eine kleine Kaffeemühle aus (siehe Abbildung 2.1).

Programmierbare Rechenmaschinen brauchen einen Speicher, der als gestanzte Metallplatte vom Weber und Erfinder Joseph-Marie Jacquard eingeführt wurde, um komplizierte Muster zu weben. Charles Babbage stellte 1833



Abb. 2.1 Curta-Rechner (Wikipedia 2018), gemeinfrei

das funktionsfähige Konzept vor, das er in der „Analytical Engine“ zu realisieren versuchte, allerdings an Geld und Zeit scheiterte. Diese Maschine sollte die Anweisungen aus den gestanzten Platten einlesen. Ada Lovelace, Tochter des Dichters Lord Byron, beschäftigt sich schon früh mit Mathematik. Sie begeisterte sich für die technischen Erfindungen und lernte mit 16 Jahren Babbage kennen. Für seine Maschine, Vorläufer des modernen Computers, entwickelt sie ein komplexes Programm und gilt so als erste Programmiererin der Welt. Sie gab der Programmiersprache **ADA** den Namen.

Herman Hollerith verwendete als Statistiker der US-Volkszählung 1890 Lochkarten und entsprechende Auswertemaschinen, um dem wachsenden Datenanfall Herr zu werden. Aus seiner Firma ging später IBM hervor, kurz für International Business Machines.

2.1.1 Erste Generation

Die Rechner der Vierzigerjahre fußten auf elektromechanischen Relais oder Elektronen-Röhren. Ihre Anzahl konnte man an einer Hand ablesen. Sie waren riesig, schwer, störanfällig und teuer, so dass die Einsatzmöglichkeiten sehr eingeschränkt blieben. Ein Repräsentant ist ENIAC (Electronic Numerical Integrator And Computer), der 1945/46 in den USA fertiggestellt wurde. Den ersten elektronischen Rechner zu nennen, fällt schwer. Im Jahr 1941 allerdings brachte Konrad Zuse die Z3 zum laufen und auch in den USA nahmen John Atanasoff und sein Doktorand einen Computer in Betrieb.

Dieser brauchte ca. 18.000 Röhren, benötigte eine Fläche von 150 qm und wog 30 Tonnen. Die Steuerung der Maschine (Programmierung) wurde durch das Stellen von Hunderten von Schaltern und das Stecken von Kabelverbin-



Abb. 2.2 UNIVAC I Röhrenrechner. Foto: Matthias Kirschner, gemeinfrei, (Wikipedia 2016).

dungen erreicht. Zur selben Zeit entwickelte John von Neumann (1903-1957) das Grundkonzept moderner Rechner.

Der erste kommerzielle Computer der USA wurde 1951 in Betrieb genommen, die sogenannte UNIVAC I (UNIVersal Automatic Computer I) von Abbildung 2.2.

Ein wesentliches Prinzip ist, dass das Programm genauso wie die Daten im Speicher abgelegt wird. Damit ist nicht mehr nur ein starrer Ablauf möglich, sondern das Programm kann geändert werden und im Extremfall sogar sich selbst verändern.

2.1.2 Zweite Computergeneration

Elektronenröhren schalteten zwar viel schneller als elektromechanische Relais, doch sie waren teuer, produzierten einen Haufen Wärme, nahmen viel Platz weg und hatten eine geringe Lebensdauer. Im Jahr 1948 erfand William Shockley den Transistor und wurde später mit dem Nobelpreis für Physik geehrt.

Etwa von 1955 bis 1965 wurden Elektronenröhren zunehmend durch Transistoren ersetzt, einem Schaltelement aus Silizium, Gallium und Arsen. Als Hauptspeicher dienten Magnetkernspeicher, erste Magnetbänder und Magnetplatten. In den Jahren 1958/59 entwickelt die Firma Texas Instruments

den ersten integrierten Schaltkreis. Und 1960 wird die Programmiersprache ALGOL-60 vorgestellt.

2.1.3 Dritte Computergeneration

In den frühen Siebzigerjahren wurden mehrere Transistoren und Bauelemente zu Schaltgruppen zusammengefasst. Die Rechner werden durch *Betriebssysteme* gesteuert. Es entstanden erste Rechenzentren.

Computer-Benutzer gaben ihren Auftrag (Job) auf und mussten auf die Aushändigung der Resultate warten. Typisch war die Eingabe eines Programms auf Lochkarten. Später wurden Terminals angeschlossen, an denen Benutzer Befehle direkt eingeben konnten. Da dann mehrere Programme gleichzeitig abgearbeitet werden, müssen die Betriebssysteme das *time sharing* beherrschen.

2.1.4 Vierte Computergeneration

Riesige Fortschritte in der Halbleitertechnik, besonders der Miniaturisierung, ermöglichen eine immer höhere Integrationsdichte, d.h. Schalter pro Fläche. Dadurch werden sogenannte „Mikrocomputern“ möglich und für Private langsam erschwinglich. 1977 kam der sehr preiswerte Apple II auf den Markt, der von einer einzigen Person, Steve Wozniak, entworfen wurde. 1981 konterte IBM mit dem als Personal Computer PC bezeichneten Modell (Abbildung 2.3), das für einige Jahr als der „PC“ galt. Viele kleinere Firmen bauten PC-kompatible Modelle. Es verwendete DOS 1.0 von Microsoft als Betriebssystem. Das ursprüngliche Betriebssystem kaufte Microsoft für schätzbare 50,000 USD und machte Millionen wenn nicht Milliarden damit.

Nebeneinander existieren heute:

- Grossrechner (Main-Frames),
- Workstations und
- Personal Computers (PC) sowie
- Personal Devices (Smart-Phone, Spielkonsolen, Tablets etc.).

Durch die nun mögliche Vernetzung wurden Aufgabenteilungen wie im Client-Server-Konzept möglich, die neben die Rechenzentren treten. Letztere haben dank der „Cloud“-Entwicklung wieder Morgenluft bekommen.



Abb. 2.3 Personal Computer von IBM (Wikipedia contributors 2018), ©Ruben de Rijcke, CC-BY-SA 3.0

2.2 Software

Es scheint nicht klar zu sein, wann der Begriff „Software“ Eingang in den Sprachgebrauch gefunden hat; vermutlich Ende der Fünfzigerjahre und nachdem der Begriff „Hardware“ gebräuchlich wurde. Das Paar Hard- und Software wurde von Howard Aiken, der die Mark I entwickelt hatte und Grace M. Hopper, die als Mathematikerin die Anweisungen auf Lochkarten brachte, verkörpert. Grace merkte schnell, dass wiederkehrende Nebenrechnungen häufig auftraten, die aber erst mit einem Speicher wiederverwertet werden konnten. Diese Nebenrechnungen nannte man dann „Subroutines“. Hopper stellte ein Programm zusammen, das in der Lage war, diese gespeicherten Subroutines wiederholt einzulesen. Sie nannte dieses Programm „Compiler“. Verbesserte Versionen des Compilers wurden dann den Käufern mit der Hardware ausgeliefert.

1957 wurde **FORTRAN** von IBM vorgestellt, eine höhere Programmiersprache. Der Name leitet sich von Formula-Translator ab, womit auch schon etwas über die Verwendung als eher mathematischer Problemlöser gesagt ist. **LISP**, heute noch verwendet, und **ALGOL** sind weitere, bei Akademikern damals beliebte Sprachen. Auf Bestreben des U.S. Department of Defense wurde **COBOL** 1959 entwickelt, das dem Aufbau der englischen Sprache glich.

Mit der wachsenden Komplexität und Leistungsfähigkeit der Rechner wurde es für den Benutzer zusehends schwieriger, alle Verwaltungs- und Überwachungsaufgaben im Betrieb zu erledigen. Hardwareproduzenten und Benutzergruppen stellten kleinere und größere Programme zusammen, um den Betrieb zu vereinfachen. Solche „Systemsoftware“ verwandelte sich dann in die sogenannten Betriebssysteme oder englisch „Operating Systems“. Ende der Sechzigerjahre entstand das Betriebssystem **UNIX**, zuerst für den internen Gebrauch der Bell Labs, dann mit der zusammen entwickelten Sprache **C**

neu implementiert. Durch die Nähe von UNIX mit C eignet sich diese Sprache sowohl für systemnahe als auch für problemnahe Programme.

UNIX ist modular und damit nicht sehr einheitlich aufgebaut. Dieses Betriebssystem erlaubte es mit dem Time-Sharing, der Zuteilung von Rechenleistung auf mehrere, an Terminalen sitzende Benutzer, beinahe moderne Verhältnisse zu erlangen. Jeder hatte den Eindruck, an einem eigenen Gerät zu arbeiten. C ist eine der am weitesten verbreiteten Programmiersprachen. UNIX blieb eine lizenzierte Software von AT&T während an der Universität von Berkley eine freie Distribution unter dem Kürzel BSD für „Berkley Systems Distribution“ entstand. Wie man sich leicht vorstellen kann, kamen sich die kommerziellen Interessen in die Quere, was zu gerichtlichen Verfahren führte. Um die Geschichte kurz zu halten: es entstanden mehrere proprietäre Systeme (AIX, Solaris etc.). Im Jahr 1991 entwickelte der Finne Linus Torvalds eine Programm, das sich als Betriebssystem herausstellte. Im Namen ist ein Verweis auf UNIX impliziert, da es ähnlich aufgebaut ist. Um den Kern des Programms sind verschiedene Distributionen entstanden.

Zurück in der Zeit: 1968 kündigte IBM an, künftig Software unabhängig von der Hardware zu verkaufen. Damit startet der Software-Markt und die Nachfrage nach Portabilität.

Die frühen Sprachen sind typischerweise *Top-Down-Ansätze*. Ein Problem wird in kleinere Portionen mit Wiederverwertbarkeit oder Bibliotheken zerlegt. Das liegt auch der prozeduralen, ablaufgesteuerten Programmierung zugrunde.

Im Laufe der Achtzigerjahre gewannen *Bottom-up-Ansätze* für die Programmiersprachen an Bedeutung. Dies war ebenfalls der gestiegenen Komplexität geschuldet. Ausgangspunkt war damals das Objekt, das die Eigenschaften einer sogenannten Klasse aufweist. Das Objekt zeichnet sich durch Strukturelemente wie *Daten* und funktionale Elemente wie *Methoden* aus. Ein Fahrzeug beispielsweise besitzt einen Fahrer, eine Farbe und Leistungsmerkmale und kann fahren, stoppen, zuladen etc. Die Objekte sind der Ausgangspunkt für die Programmierung.

Die grundlegenden Ideen der Objektorientierung stammen von den Sprachen SIMULA und deren Nachfolgerin SMALLTALK und flossen um 1983 in einer Vermengung von C in die verbreitete Sprache C++ und **Objective C** ein. Die Referenz auf C ist eindeutig. 1995 entstand dann daraus die Sprache JAVA. Etwa gleichzeitig entstand Python, die eine recht breite Anwendergemeinschaft gefunden hat. Die Applikation in Android werden mit JAVA realisiert, Apple verwendet für das iPhone **Objective C**.

In den letzten Jahren haben sich auch wieder neue Sprachen entwickelt. Zu nennen sind *Go* oder Golang von Google, *Swift* von Apple und *Hack* von Facebook. Go ist nicht objektorientiert, aber sie kann gut mit gleichzeitig ausführbaren Elementen (Concurrency) umgehen.

Weitere Details findet man bei Petzold (1999).

Quiz zu Kapitel 2

Quiz. Wie steht es mit einem Selbsttest?

1. Wann wurde der erste elektronische Computer in Betrieb genommen?
2. Wie wurden die Schaltungen realisiert?
3. Welche Bedeutung kommt den Transistoren und der Miniaturisierung zu?
4. Wie entstand die Unterscheidung in Hard- und Software?
5. Wann kam der PC auf den Markt?
6. Wofür steht **FORTRAN**?
7. Welches sind aktuell die gebräuchlichsten Programmiersprachen?
8. Was ist Objektorientierung?
9. Was unterscheidet **C++** von **C**?
10. Welche Sprachen kommen in Android und im iPhone zur Anwendung?

Literatur

Petzold, C. (1999). *Code : the hidden language of computer hardware and software*. Microsoft Press, Redmond, Wash.

Wikipedia (2016). UNIVAC I — Wikipedia, Die freie Enzyklopädie. https://de.wikipedia.org/w/index.php?title=UNIVAC_I&oldid=154306265. [Online; Stand 25. August 2018].

Wikipedia (2018). Curta — Wikipedia, Die freie Enzyklopädie. <https://de.wikipedia.org/w/index.php?title=Curta&oldid=178429295>. [Online; Stand 1. Oktober 2018].

Wikipedia contributors (2018). IBM Personal Computer — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=IBM_Personal_Computer&oldid=861836470. [Online; accessed 1-October-2018].



Kapitel 3

Alles ist Zahl

... aber nicht unbedingt alle Ziffern werden benötigt. Die Basis der Computer sind physikalische Eigenschaften der Materie. Ein Teilchen ist geladen oder nicht, Strom fließt oder nicht, ein Sitz ist besetzt oder nicht, ein Gatter ist offen oder zu usw. Viele Eigenschaften sind zweiwertig. Und dies ist die Basis der Rechner. Zweiwertige Zahlensysteme, man nennt sie binär, bestehen nur aus den Ziffern 1 und 0. Die Basis der Zahlen ist 2. Für die „normalen“ Zahlen ist die Basis 10. Man kann die Basis beliebig wählen, z.B. 8 oder auch 16.

3.1 Zweier- oder Binärsystem

Das Zweiersystem wurde von Gottfried Wilhelm Leibniz am Anfang des 18. Jahrhunderts, genauer 1705, in einem Artikel veröffentlicht und darin vollständig dokumentiert. Man kann jede Zahl als Summe von Potenzen von 2 darstellen. Man kann mit Binärzahlen rechnen wie mit „normalen“ Zahlen, also im Zehnersystem.

Beispiel 3.1.1 (Zahlenstruktur). Die Zahl 123 besteht aus 3 Einsern, 2 Zehnern und einem Hunderter, also $123_{10} = 3 \cdot 1 + 2 \cdot 10 + 1 \cdot 100$. Andererseits gilt auch: $123_{10} = 1 \cdot 64 + 1 \cdot 32 + 1 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 1 \cdot 1$ oder 1111011₂.
△

Beispiel 3.1.2 (Morse-Code). Das erst vor Kurzem offiziell abgeschaffte Morse-System stellt Buchstaben mit zweiwertigen Zeichen dar. „Kurz“ oder „lang“ als Ton, Licht, als ein oder zwei Flaggen usw. Die Übersetzung von Zeichen in Ziffern nennt man *Kodieren*. Im folgenden ein kodierter Text:

```
..-.-./...//.-./../-.../
//...-./.-./-./-.../---/...//...-.-./...//.-
//.-.-./..//...//...-.-./-...//...-.-.-
//--./.-./..//...-.-./-./-//
.-/----/-.-/-...//...-.-/-.-/-
```

Ich liebe franzoesischen Kaese speziell gereiften Roquefort

△

Da sämtliche Daten und Programm-Befehle binärcodiert werden, also als 1 oder 0 dargestellt werden, kann eine bestimmte Bitfolge unterschiedliche Dinge bedeuten. Die Interpretation ist abhängig vom Ort der Verarbeitung, z.B.

Bitmuster	01100001 01100010 01100011 01100100
ASCII-Zeichen	abcd
ganze Zahl	1 633 837 924
Gleitpunktzahl	0.7608150709 ... $\cdot 2^{-30}$
Befehl	LOAD 6447972

Tabelle 3.1 Das Bitmuster ist interpretationsbedürftig. Je nach Ort oder Kontext des Computers ist erst ersichtlich, was das Bitmuster bedeutet.

im Steuerwerk, im Rechenwerk für ganze Zahlen, im Rechenwerk für Gleitpunktzahlen usw. Ähnliches gilt für Dateien, die ja auch nur Bits bestehen. Deshalb erkennt man von „aussen“ an der sogenannten Dateierweiterung, was der Inhalt ist. Die ersten Zeichen der Datei lassen wiederum das Format erkennen (siehe Abbildung 3.1 und 3.3).

Definition 1 (Bit, Byte). Das Byte ist eine Maßeinheit der Informatik, welches (meist) für eine Folge aus 8 Bit steht. Ein Bit ist eine zweiwertige (binäre) Ziffer, die den Wert 1 oder 0 annimmt. In Tabelle 3.1 ist ein Beispiel wiedergegeben.

Beispiel 3.1.3 (Addition). Das Rechnen mit Binärzahlen ist gleich wie das bekannte Rechnen mit Dezimalzahlen. Allerdings kommt es zu einem Übertrag, sobald $1+1$ vorkommt. Wir addieren zwei Zahlen, d.h. 1011_2 und 1101_2 .

Summand 1	1 0 1 1
Summand 2	1 1 0 1
Übertrag	1 1 1 1
Summe	1 1 0 0 0

$1+0=1$, $1+1$ gibt 0 mit Übertrag 1, $1+1+1$ gibt 1 mit Übertrag 1. Dieses Rechenbeispiel werden wir weiter unten wieder verwenden, und zwar bei den Rechenschaltungen. Der erste Rechenschritt hat noch keinen Übertrag. Jeder Schritt gibt die Summe und einen Übertrag aus.

△

Beispiel 3.1.4 (Subtraktion). Und nun die Subtraktion der beide Zahlen, also $1101_2 - 1011_2$:

Minuend	1 1 0 1
Subtrahend	1 0 1 1
Borger	0 0 1 0
Differenz	0 0 1 0

Bei der Subtraktion kann es sein, dass das Resultat negativ ist. Der letzte Borger ist dann 1. △

Beispiel 3.1.5 (Subtraktion ohne Borgen). Die Subtraktion kann man auch ohne Borgen ausführen. Dazu berechnet man das Komplement des Subtrahenden. Dieses ist die Zahl, die sich aus dem Vertauschen von 1 und 0 des Subtrahenden ergibt. Im vorhergehenden Beispiel also von 1011 zu 0100¹. Dann addiert man noch eine 1 zu 0101. Diese Zahl addiert man zum Minuenden: 1101 + 0101 = 10010. Sodann streicht man die erste Zahl und es bleibt 0010 übrig, dasselbe Resultat wie in Beispiel 3.1.4. △

3.2 Sechzehner-System

Zahlen kann man als Potenzen von beliebigen Basen, also 10, 2, 8 oder 16, darstellen. Die Darstellung im Binärsystem hat den großen Nachteil, dass die Zahlen viele Stellen aufweisen. Die Zahl 15₁₀ ist beispielsweise 1111₂. Damit verdoppelt sich die Länge. Andererseits könnte man die Basis 16 wählen, wobei dann der Zahlenvorrat mit Buchstaben angereichert werden müsste, also {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. Die Zahl 15₁₀ wäre dann F₁₆. Entsprechend ist 101011111111 dann 15FF. Vier Ziffern der binären Darstellung werden durch eine Zahl repräsentiert, eine Kompression vom Faktor 4. Das Sechzehner- oder Hexadezimalsystem ist eine logische Erweiterung des

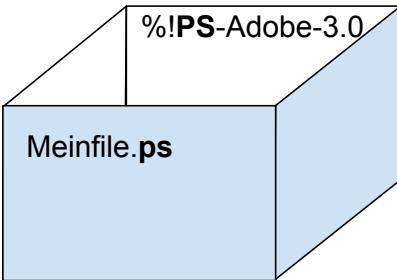


Abb. 3.1 Symbolisch soll die Darstellung zeigen, dass eine Datei von außen eine Erweiterung („extension“) trägt, die die Art von Datei definiert. Hier wird mit ps ein Postscript-File bezeichnet. Im Innern, in der ersten Zeile kann das Programm den Datentyp lesen.

¹ Das Komplement ist auch die Zahl mit nur Einsen minus den Subtrahenden, 1111-1011. Weiters ist 1111=10000-1. Alles zusammen: 1101+(1111-1011)+1-10000.

Binärsystems, denn $2^4 = 16$.

Die größtmögliche Binärzahl, die in einem Rechenschritt verarbeitet werden kann, nennt man „Datenwort“. In den Siebzigerjahren war die Wortlänge der Rechner 8 Bit, die man dann *Byte* nannte. Mit der Leistungssteigerung der Prozessoren konnten diese ständig größere Brocken verarbeiten, nämlich 16, 32 und heute meist 64 Bits.

3.3 Gleitkomma-Zahlen*

Bis hierhin haben wir nur mit ganzen Zahlen gerechnet, weil wir von der Binärdarstellung ausgegangen sind. Neben den ganzen Zahlen 1, 2, ... gibt es aber die gebrochenen Zahlen, z.B. $1/3$, $5/16$ und 0.1234 sowie die irrationalen Zahlen wie $\sqrt{2}$. Nicht-ganze Zahlen, sogenannte reellen Zahlen, können unendlich viele Stellen nach dem Komma aufweisen. Die Informatik muss also das Problem lösen, reelle Zahlen mit beschränktem Platz bestens anzunähern.

Hierfür hat man sich auf das Gleitkomma-Format festgelegt, bei dem das Komma hinter die erste signifikante Ziffer gebracht wird und der Ort des Kommas über den Exponenten der Zehnerpotenz festgehalten wird. Beispielsweise ist die Lichtgeschwindigkeit $c = 299792458[m/s]$. Sie wird als $2,99792458 \cdot 10^8$ dargestellt. Daraus folgt $c^2 = 8.98755178E+16$. Man beachte, dass die Ausgabe den amerikanischen Punkt anstatt das Komma aufweist und dass „E“ für die Zehn verwendet wird.

Gleitkommazahlen können leicht ungenau sein. Wenn ich $\sqrt{2}^2$ rechnen lasse, was im Skript als $(2**0.5)**2$ daherkommt, dann bekomme ich nicht exakt 2 sondern 2.0000000000000004. Oder $((2**0.5)**0.5)**4$ ergibt 1.9999999999999998.

3.4 Rohe Daten, Alphabete

Zeichenkette	F r a n z e t t i
HEX	46 72 61 6E 7A 65 74 74 69

Beispiel 3.4.1 (Spezialbuchstaben). Du sitzt am Computer und betrachtest deine Tastatur. Die einzelnen Tasten tragen eine Beschriftung, z.B. **s**. Die schweizerische Tastatur kennt das „ß“ nicht. Nun kann man diesen Buchstaben über seine Nummer produzieren. Sie lautet ALT+0223. Welche anderen Ziffern funktionieren? △

³ in <http://www.unicode.org/copyright.html>, abgerufen am 1. November 2018.

	1092	1093		
0	Α	Υ	10920	10930
1	Β	8	10921	10931
2	ϸ	+	10922	10932
3	λ	Ƨ	10923	10933
4	↘	⌚	10924	10934
5	λ	Μ	10925	10935
6	Ι	Υ	10926	10936

Letters
10920 Α LYDIAN LETTER A
10921 Β LYDIAN LETTER B
10922 ϸ LYDIAN LETTER G
10923 λ LYDIAN LETTER D
10924 ↘ LYDIAN LETTER E
10925 λ LYDIAN LETTER V
10926 Ι LYDIAN LETTER I
10927 Υ LYDIAN LETTER Y
10928 ϸ LYDIAN LETTER K
10929 λ LYDIAN LETTER L
1092A ϸ LYDIAN LETTER M
1092B ϸ LYDIAN LETTER N
1092C ο LYDIAN LETTER O
1092D ϸ LYDIAN LETTER R
1092E ϸ LYDIAN LETTER SS
1092F T LYDIAN LETTER T
10930 Υ LYDIAN LETTER U
10931 8 LYDIAN LETTER F
10932 + LYDIAN LETTER Q
10933 Ƨ LYDIAN LETTER S
10934 ⌚ LYDIAN LETTER TT
10935 Μ LYDIAN LETTER AN
10936 Υ LYDIAN LETTER EN
10937 Υ LYDIAN LETTER LY
10938 ϸ LYDIAN LETTER NN
10939 ↑ LYDIAN LETTER C

Abb. 3.2 Lydische Schrift mit Unicode-Nummern (Copyright ©1991-2018 Unicode, Inc. and others, all rights reserved, distributed under the Terms of Use³).

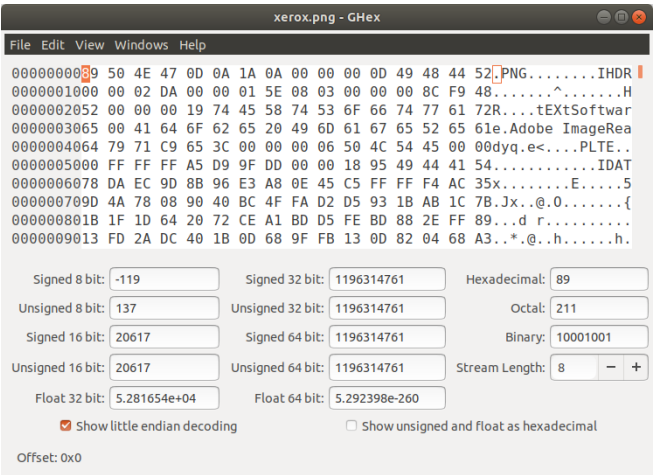


Abb. 3.3 Eine Bild-Datei vom Typ .PNG als Zahlenfolge im Sechzehner-Darstellung. Man verwendet einen Spezialeditor, um solche Darstellungen zu machen.

3.5 Komprimierung und Verifizierung

Daten können häufig komprimiert werden; dies ist eine Erfahrungstatsache, denn wir kennen Programme wie Zip, Gzip, RAR etc. Es ist aber ebenso eine