

# Inhalt

Vorwort .....	23
Vorwort des Gutachters .....	25
<b>1 Einstieg in C .....</b>	<b>27</b>
1.1 Übersicht zu C .....	27
1.2 Der ANSI-C-Standard .....	28
1.2.1 Welcher C-Standard wird in diesem Buch verwendet? ....	30
1.2.2 Der Vorteil des ANSI-C-Standards .....	31
1.3 Der POSIX-Standard .....	31
1.4 Vor- und Nachteile der Programmiersprache C .....	32
1.5 C in diesem Buch .....	33
1.6 Was benötige ich für C? .....	33
1.6.1 Texteditor .....	33
1.6.2 Compiler .....	34
1.6.3 All-in-one – die Entwicklungsumgebung .....	34
1.7 Welcher Compiler und welches Betriebssystem? .....	35
<b>2 Das erste Programm .....</b>	<b>37</b>
2.1 Der beste Lernerfolg .....	37
2.2 »Hallo Welt« in C .....	37
2.3 Analyse des Programms .....	39
<b>3 Grundlagen .....</b>	<b>43</b>
3.1 Zeichensätze .....	43
3.1.1 Basic-Zeichensatz .....	43
3.1.2 Ausführungszeichensatz (Steuerzeichen) .....	44
3.1.3 Trigraph-Zeichen .....	46
3.2 Symbole von C .....	48
3.2.1 Bezeichner .....	48
3.2.2 Schlüsselwörter .....	48
3.2.3 Literale .....	48
3.2.4 Einfache Begrenzer .....	50
3.2.5 Kommentare .....	51

<b>4 Formatierte Ein-/Ausgabe mit »scanf()« und »printf()« .....</b>	<b>55</b>
4.1 Formatierte Eingabe mit »scanf()« .....	55
4.1.1 Der Adressoperator »&« .....	57
4.1.2 Probleme und deren Behandlung mit »scanf()« .....	58
4.1.3 Überprüfen auf das richtige Format .....	61
4.1.4 Zusammenfassung zu »scanf()« .....	63
4.2 Formatierte Ausgabe mit »printf()« .....	63
<b>5 Basisdatentypen .....</b>	<b>67</b>
5.1 Deklaration und Definition .....	67
5.2 Der Datentyp »int« (Integer) .....	69
5.3 Variablen verwenden .....	70
5.4 Der Datentyp »long« .....	73
5.5 Der Datentyp »long long« .....	74
5.6 Der Datentyp »short« .....	74
5.7 Ganzzahlige Typen mit vorgegebener Breite – <stdint.h> .....	75
5.7.1 <inttypes.h> (C99) .....	77
5.8 Die Gleitpunkttypen »float« und »double« .....	78
5.8.1 Gleitpunkttypen im Detail .....	80
5.8.2 »float« im Detail .....	80
5.8.3 »double« im Detail .....	81
5.8.4 long double .....	81
5.8.5 Einiges zu n-stelliger Genauigkeit .....	82
5.9 Numerische Gleitpunktprobleme .....	83
5.10 Komplexe Gleitpunkttypen – <complex.h> .....	86
5.11 Der Datentyp »char« .....	87
5.12 Nationale contra internationale Zeichensätze .....	92
5.13 Der Breitzeichen-Typ »wchar_t« .....	94
5.14 Multibyte-Zeichen .....	95
5.15 Boolescher Wert – <stdbool.h> .....	96
5.16 Vorzeichenlos und vorzeichenbehaftet .....	97
5.17 Limits für Ganzzahl- und Gleitpunkt datentypen .....	99
5.18 Der Typ »void« .....	102
5.19 Konstanten .....	103
5.19.1 Ganzzahlkonstanten .....	103
5.19.2 Gleitpunktkonstanten .....	103
5.19.3 Zeichenkonstanten .....	104
5.19.4 Stringliterale (Stringkonstante) .....	104
5.20 Umwandlungsvorgaben für formatierte Ein-/Ausgabe .....	104

**6 Operatoren ..... 113**

6.1	Exkurs zu Operatoren .....	113
6.2	Arithmetische Operatoren .....	114
6.2.1	Dividieren von Ganzzahlen .....	115
6.3	Erweiterte Darstellung arithmetischer Operatoren .....	117
6.4	Inkrement- und Dekrement-Operatoren .....	118
6.5	Bit-Operatoren .....	119
6.5.1	Bitweises UND .....	120
6.5.2	Bitweises ODER .....	122
6.5.3	Bitweises XOR .....	122
6.5.4	Bitweises Komplement .....	123
6.5.5	Linksverschiebung .....	124
6.5.6	Rechtsverschiebung .....	125
6.5.7	Rezept für Fortgeschrittene .....	125
6.6	Makros für logische Operatoren und Bit-Operatoren – <iso646.h> .....	126
6.7	Der »sizeof«-Operator .....	127
6.7.1	C versus C++ .....	129

**7 Typumwandlung ..... 131**

7.1	Implizite Datentypumwandlung .....	131
7.1.1	Implizites »char« nach »int« .....	131
7.1.2	Implizites »float« nach »double« .....	132
7.1.3	Implizite Umwandlung in einen komplexen Gleitpunkttyp .....	132
7.1.4	Übliche arithmetische Datentypumwandlung .....	132
7.2	Explizite Datentypumwandlung mit dem »cast«-Operator .....	135

**8 Kontrollstrukturen ..... 137**

8.1	Verzweigungen mit der »if«-Bedingung .....	137
8.1.1	Anweisungsblock .....	138
8.2	Die Verzweigung mit »else if« .....	141
8.3	Die Verzweigung mit »else« .....	143
8.4	Der !-Operator (logischer Operator) .....	147
8.5	Logisches UND (&&) – logisches ODER (  ) .....	149
8.6	Der Bedingungsoperator »?:« .....	151
8.7	Fallunterscheidung: die »switch«-Verzweigung .....	153
8.7.1	default .....	156
8.8	Die »while«-Schleife .....	157
8.8.1	Endlosschleife (»while«) .....	159
8.8.2	Fehlervermeidung bei »while«-Schleifen .....	160

8.9	Die »do while«-Schleife .....	161
8.10	Die »for«-Schleife .....	165
8.10.1	Einsatzmöglichkeiten der »for«-Schleife .....	168
8.11	Kontrollierte Sprünge .....	172
8.11.1	<i>continue</i> .....	172
8.11.2	<i>break</i> .....	173
8.12	Direkte Sprünge mit »goto« .....	174
8.13	Notationsstil .....	175
8.13.1	K&R-Stil .....	175
8.13.2	Whitesmith-Stil .....	175
8.13.3	Allman-Stil .....	175
8.13.4	GNU EMACS-Stil .....	176
8.13.5	Der Stil des Autors ;) (K&R-like) .....	176
<b>9</b>	<b>Funktionen .....</b>	<b>177</b>
9.1	Was sind Funktionen? .....	177
9.2	Wozu dienen Funktionen? .....	177
9.3	Definition von Funktionen .....	177
9.4	Funktionsaufruf .....	178
9.5	Funktionsdeklaration .....	180
9.6	Lokale Variablen .....	181
9.7	Globale Variablen .....	184
9.8	Statische Variablen .....	185
9.9	Schlüsselwörter für Variablen – Speicherklassen .....	186
9.9.1	<i>auto</i> .....	187
9.9.2	<i>extern</i> .....	187
9.9.3	<i>register</i> .....	187
9.9.4	<i>static</i> .....	187
9.10	Typ-Qualifizierer .....	188
9.10.1	<i>volatile</i> .....	188
9.10.2	<i>const</i> .....	188
9.11	Geltungsbereich von Variablen .....	188
9.12	Speicherklassen-Spezifizierer für Funktionen .....	190
9.12.1	<i>extern</i> .....	190
9.12.2	<i>static</i> .....	190
9.12.3	<i>volatile</i> .....	190
9.13	Datenaustausch zwischen Funktionen .....	191
9.14	Wertübergabe an Funktionen (call-by-value) .....	192
9.15	Der Rückgabewert von Funktionen .....	195
9.16	Die Hauptfunktion »main()« .....	197

9.17	Rückgabewert beim Beenden eines Programms .....	199
9.17.1	Programmende auswerten .....	200
9.18	Funktionen der Laufzeitbibliothek .....	202
9.19	Getrenntes Kompilieren von Quelldateien .....	203
9.20	Rekursive Funktionen (Rekursion) .....	206
9.20.1	Exkurs: Stack .....	206
9.20.2	Rekursionen und der Stack .....	206
9.20.3	Fakultät .....	211
9.20.4	Fibonacci-Zahlen .....	212
9.20.5	Größter gemeinsamer Teiler (GGT) .....	213
9.21	»inline«-Funktionen .....	217
<b>10</b>	<b>Präprozessor-Direktiven .....</b>	<b>221</b>
10.1	Einkopieren von Dateien mittels »#include« .....	222
10.2	Makros und Konstanten – »#define« .....	224
10.2.1	Symbolische Konstanten mit »#define« .....	225
10.2.2	Makros mit »#define« .....	229
10.3	Bedingte Kompilierung .....	233
10.4	Vordefinierte Präprozessor-Direktiven (ANSI C) .....	238
10.5	Ersetzung eines Makroparameters durch einen String .....	240
10.6	»#undef« – Makronamen wieder aufheben .....	241
10.7	Ausgeben von Fehlermeldungen – »#error« .....	242
10.8	»#pragma« .....	243
<b>11</b>	<b>Arrays .....</b>	<b>245</b>
11.1	Arrays deklarieren .....	245
11.2	Initialisierung und Zugriff auf Arrays .....	246
11.2.1	Gültigkeitsbereich von Arrays .....	251
11.3	Arrays vergleichen .....	253
11.4	Anzahl der Elemente eines Arrays ermitteln .....	255
11.5	Übergabe von Arrays an Funktionen .....	256
11.6	Arrays aus Funktionen zurückgeben .....	258
11.7	Programmbeispiel zu den Arrays .....	259
11.8	Einlesen von Array-Werten .....	263
11.9	Mehrdimensionale Arrays .....	263
11.9.1	Mehrdimensionale Arrays initialisieren .....	264
11.9.2	Übergabe von zwei- bzw. mehrdimensionalen Arrays an Funktionen .....	276
11.10	Arrays in Tabellenkalkulation einlesen (*.CSV-Dateien) .....	278

11.11 Strings/Zeichenketten (»char«-Array) .....	280
11.11.1 Vom String zur Binärzahl .....	283
11.12 Einlesen von Strings .....	286
11.13 Die Standard-Bibliothek <string.h> .....	288
11.13.1 »strcat()« – Strings aneinanderhängen .....	288
11.13.2 »strchr()« – ein Zeichen im String suchen .....	289
11.13.3 »strcmp()« – Strings vergleichen .....	290
11.13.4 »strcpy()« – einen String kopieren .....	291
11.13.5 »strcspn()« – einen Teilstring ermitteln .....	292
11.13.6 »strlen()« – Länge eines Strings ermitteln .....	292
11.13.7 »strncat()« – String mit n Zeichen aneinanderhängen ...	293
11.13.8 »strcmp()« – n Zeichen von zwei Strings miteinander vergleichen .....	294
11.13.9 »strncpy()« – String mit n Zeichen kopieren .....	294
11.13.10 »strpbrk()« – nach dem Auftreten bestimmter Zeichen suchen .....	295
11.13.11 »strrchr()« – das letzte Auftreten eines bestimmten Zeichens im String suchen .....	296
11.13.12 »strspn()« – das erste Auftreten eines Zeichens, das nicht vorkommt .....	296
11.13.13 »strstr()« – einen String nach dem Auftreten eines Teilstrings durchsuchen .....	297
11.13.14 »strtok()« – einen String anhand bestimmter Zeichen zerlegen .....	297
<b>12 Zeiger (Pointer) .....</b>	<b>299</b>
12.1 Zeiger deklarieren .....	300
12.2 Zeiger initialisieren .....	301
12.2.1 Speichergröße von Zeigern .....	312
12.3 Zeigerarithmetik .....	313
12.4 Zeiger, die auf andere Zeiger verweisen .....	314
12.4.1 Subtraktion zweier Zeiger .....	315
12.5 Typensicherung bei der Dereferenzierung .....	316
12.6 Zeiger als Funktionsparameter (call-by-reference) .....	317
12.6.1 Zeiger als Rückgabewert .....	320
12.7 Array und Zeiger .....	323
12.8 Zeiger auf Strings .....	330
12.8.1 Zeiger auf konstante Objekte (Read-only-Zeiger) .....	330
12.9 Zeiger auf Zeiger und Stringtabellen .....	331
12.9.1 Stringtabellen .....	333

12.10 Zeiger auf Funktionen .....	340
12.11 void-Zeiger .....	346
12.12 Äquivalenz zwischen Zeigern und Arrays .....	349
12.13 Der »restrict«-Zeiger .....	351

## 13 Kommandozeilenargumente ..... 355

13.1 Argumente an die Hauptfunktion übergeben .....	355
13.2 Optionen (Schalter) aus der Kommandozeile auswerten .....	361

## 14 Dynamische Speicherverwaltung ..... 365

14.1 Das Speicherkonzept .....	366
14.2 Speicherallokation mit »malloc()« .....	367
14.3 Das NULL-Mysterium .....	370
14.3.1 NULL für Fortgeschrittene .....	370
14.3.2 Was jetzt – NULL, 0 oder \0 ... ? .....	372
14.3.3 Zusammengefasst .....	373
14.4 Speicherreservierung und ihre Probleme .....	373
14.5 »free()« – Speicher wieder freigeben .....	374
14.6 Die Freispeicherverwaltung .....	377
14.6.1 Prozessinterne Freispeicherverwaltung .....	379
14.7 Dynamische Arrays .....	381
14.8 Speicher dynamisch reservieren mit »realloc()« und »calloc()« .....	385
14.9 Speicher vom Stack anfordern mit »alloca()« (nicht ANSI C) .....	389
14.10 »free()« – Speicher wieder freigeben .....	389
14.11 Zweidimensionale dynamische Arrays .....	390
14.12 Wenn die Speicherallokation fehlschlägt .....	393
14.12.1 Speicheranforderung reduzieren .....	394
14.12.2 Speicheranforderungen aufteilen .....	395
14.12.3 Einen Puffer konstanter Größe verwenden .....	396
14.12.4 Zwischenspeichern auf Festplatte vor der Allokation .....	397
14.12.5 Nur so viel Speicher anfordern wie nötig .....	397

## 15 Strukturen ..... 399

15.1 Struktur deklarieren .....	399
15.2 Initialisierung und Zugriff auf Strukturen .....	401
15.3 Strukturen als Wertübergabe an eine Funktion .....	408
15.4 Strukturen als Rückgabewert einer Funktion .....	410
15.5 Strukturen vergleichen .....	412
15.6 Arrays von Strukturen .....	413

15.7	Strukturen in Strukturen (Nested Structures) .....	420
15.8	Kurze Zusammenfassung zu den Strukturen .....	430
15.9	Unions .....	431
15.10	Der Aufzählungstyp »enum« .....	436
15.11	Typendefinition mit »typedef« .....	439
15.12	Attribute von Strukturen verändern (nicht ANSI C) .....	443
15.13	Bitfelder .....	446
15.14	Das »offsetof«-Makro .....	453
<b>16</b>	<b>Ein-/Ausgabe-Funktionen .....</b>	<b>455</b>
16.1	Was ist eine Datei? .....	455
16.2	Formatierte und unformatierte Ein-/Ausgabe .....	455
16.3	Standard-Streams .....	456
16.4	Höhere Ein-/Ausgabe-Funktionen .....	456
16.5	Datei (Stream) öffnen – »fopen« .....	457
16.5.1	Modus für »fopen()« .....	460
16.5.2	Maximale Anzahl geöffneter Dateien – »FOPEN_MAX« .....	463
16.6	Zeichenweise lesen und schreiben – »getchar()« und »putchar()« .....	464
16.6.1	Ein etwas portableres »getch()« .....	466
16.7	Zeichenweise lesen und schreiben – »putc()«/»fputc()« und »getc()«/»fgetc()« .....	468
16.8	Datei (Stream) schließen – »fclose()« .....	474
16.9	Formatiertes Einlesen/Ausgeben von Streams mit »fprintf()« und »fscanf()« .....	477
16.10	Standard-Streams in C .....	482
16.10.1	Standard-Streams umleiten .....	482
16.11	Fehlerbehandlung von Streams – »feof()«, »ferror()« und »clearerr()« .....	485
16.12	Gelesenes Zeichen in die Eingabe zurückziehen – »ungetc()« .....	487
16.13	(Tastatur-)Puffer leeren – »fflush()« .....	489
16.14	Stream positionieren – »fseek()«, »rewind()« und »ftell()« .....	490
16.15	Stream positionieren – »fsetpos()«, »fgetpos()« .....	494
16.16	Zeilenweise Ein-/Ausgabe von Streams .....	496
16.16.1	Zeilenweise lesen mit »gets()«/»fgets()« .....	496
16.16.2	Zeilenweise schreiben mit »puts()«/»fputs()« .....	499
16.16.3	Zeilenweise vom Stream einlesen mit »getline()« (nicht ANSI C) .....	500
16.16.4	Rezepte für zeilenweises Einlesen und Ausgeben .....	502
16.17	Blockweise lesen und schreiben – »fread()« und »fwrite()« .....	509

16.17.1	Blockweise lesen – »fread()« .....	510
16.17.2	Blockweise schreiben – »fwrite()« .....	512
16.17.3	Big Endian und Little Endian .....	517
16.18	Datei (Stream) erneut öffnen – »freopen()« .....	519
16.19	Datei löschen oder umbenennen – »remove()« und »rename()« ...	521
16.19.1	remove() .....	521
16.19.2	rename() .....	522
16.20	Pufferung einstellen – »setbuf()« und »setvbuf()« .....	523
16.20.1	Die Funktion »setbuf()« .....	524
16.20.2	Die Funktion »setvbuf()« .....	528
16.21	Temporäre Dateien erzeugen – »tmpfile()« und »tmpnam()« .....	530
16.21.1	»mkstemp()« – sichere Alternative für Linux/UNIX (nicht ANSI C) .....	534
16.22	Fehlerbehandlung .....	535
16.22.1	Fehlerausgabe mit »perror()« .....	536
16.22.2	Fehlerausgabe mit »strerror()« .....	537
16.23	Formatiert in einen String schreiben und formatiert aus einem String lesen – »sscanf()« und »sprintf()« .....	539
16.24	Byte- und wide-orientierter Stream .....	542
16.25	Ein fortgeschrittenes Thema .....	544
16.26	Low-Level-Datei-I/O-Funktionen (nicht ANSI C) .....	552
16.26.1	Datei öffnen – »open()« .....	553
16.26.2	Datei schließen – »close()« .....	559
16.26.3	Datei erzeugen – »creat()« .....	560
16.26.4	Schreiben und Lesen – »write()« und »read()« .....	561
16.26.5	File-Deskriptor positionieren – »lseek()« .....	571
16.26.6	File-Deskriptor von einem Stream – »fileno()« .....	572
16.26.7	Stream von File-Deskriptor – »fdopen()« .....	574

## 17 Attribute von Dateien und das Arbeiten mit Verzeichnissen (nicht ANSI C) ..... 577

17.1	Attribute einer Datei ermitteln – »stat()« .....	577
17.1.1	»stat()« – »st_mode« .....	578
17.1.2	»stat()« – »st_size« .....	583
17.1.3	»stat()« – »st_atime«, »st_mtime« und »st_ctime« .....	585
17.1.4	»stat()« – »st_gid« und »st_uid« .....	589
17.1.5	»stat()« – »st_nlink«, »st_ino« .....	590
17.1.6	»stat()« – »st_dev«, »st_rdev« .....	590
17.2	Prüfen des Zugriffsrechts – »access()« .....	593

17.3 Verzeichnispunktfunktionen .....	595
17.3.1 Verzeichnis erstellen, löschen und wechseln – »mkdir()«, »rmdir« und »chdir« .....	596
17.3.2 In das Arbeitsverzeichnis wechseln – »getcwd()« .....	601
17.3.3 Verzeichnisse öffnen, lesen und schließen – »opendir()«, »readdir()« und »closedir()« .....	603
<b>18 Arbeiten mit variabel langen Argumentlisten – &lt;stdarg.h&gt; ...</b>	<b>609</b>
18.1 Makros in <stdarg.h> – »va_list«, »va_arg«, »va_start« und »va_end« .....	609
18.2 Die Argumentliste am Anfang oder Ende kennzeichnen .....	610
18.3 »vprintf()«, »vsprintf()«, »vfsprintf()« und »vnsprintf()« .....	615
18.4 Variadic Makros – __VA_ARGS__ .....	619
<b>19 Zeitroutinen .....</b>	<b>623</b>
19.1 Die Headerdatei <time.h> .....	623
19.1.1 Konstanten in der Headerdatei <time.h> .....	624
19.1.2 Datums- und Zeitfunktionen in <time.h> .....	624
19.2 Laufzeitmessung (Profiling) .....	634
<b>20 Weitere Headerdateien und ihre Funktionen (ANSI C) .....</b>	<b>637</b>
20.1 <assert.h> – Testmöglichkeiten und Fehlersuche .....	637
20.2 <cctype.h> – Zeichenklassifizierung und Umwandlung .....	639
20.3 Mathematische Funktionen – <math.h>, <tgmath.h> und <complex.h> .....	643
20.3.1 Funktionen für reelle und komplexe Gleitpunkttypen ....	644
20.3.2 Funktionen nur für reelle Gleitpunkttypen .....	646
20.3.3 Funktionen nur für komplexe Gleitpunkttypen .....	647
20.3.4 Typgenerische Makros – <tgmath.h> .....	649
20.3.5 Gleitpunktwerte klassifizieren .....	650
20.3.6 Makro zum Vergleichen von reellen Zahlen .....	651
20.3.7 Zugriff auf die Gleitpunkt-Umgebung – <fenv.h> .....	652
20.4 <stdlib.h> .....	655
20.4.1 Programmbeendigung – »exit()«, »_exit()«, »atexit()« und »abort()« .....	655
20.4.2 Strings in numerische Werte konvertieren .....	658
20.4.3 Bessere Alternative – Strings in numerische Werte konvertieren .....	661
20.4.4 Zufallszahlen .....	666

20.4.5 Absolutwerte, der Quotient und der Rest von Divisionen .....	667
20.4.6 Suchen und Sortieren – »qsort()« und »bsearch()« .....	669
20.4.7 system() .....	671
20.5 <locale.h> – länderspezifische Eigenheiten .....	673
20.6 Nicht-lokale Sprünge – <setjmp.h> .....	676
20.7 <signal.h> .....	680
20.8 <string.h> – die »mem...«-Funktionen zur Speichermanipulation .....	685
20.8.1 »memchr()« – Suche nach einzelnen Zeichen .....	686
20.8.2 »memcmp()« – bestimmte Anzahl von Bytes vergleichen .....	686
20.8.3 »memcpy()« – bestimmte Anzahl von Bytes kopieren .....	687
20.8.4 »memmove()« – bestimmte Anzahl von Bytes kopieren .....	687
20.8.5 »memset()« – Speicherbereich mit bestimmten Zeichen auffüllen .....	688

## 21 Dynamische Datenstrukturen ..... 691

21.1 Lineare Listen (einfach verkettete Listen) .....	691
21.1.1 Erstes Element der Liste löschen .....	698
21.1.2 Ein beliebiges Element in der Liste löschen .....	700
21.1.3 Elemente der Liste ausgeben .....	702
21.1.4 Eine vollständige Liste auf einmal löschen .....	708
21.1.5 Element in die Liste einfügen .....	710
21.2 Doppelt verkettete Listen .....	717
21.3 Stacks nach dem LIFO-(Last-in-First-out-)Prinzip .....	734
21.4 Queues nach dem FIFO-Prinzip .....	754
21.5 Dynamisches Array mit flexiblen Elementen .....	762

## 22 Algorithmen ..... 765

22.1 Was sind Algorithmen? .....	765
22.2 Wie setze ich Algorithmen ein? .....	766
22.3 Sorteralgorithmen .....	766
22.3.1 »Selection Sort« – sortieren durch Auswählen .....	767
22.3.2 Insertion Sort .....	769
22.3.3 Bubble Sort .....	771
22.3.4 Shellsort .....	772
22.3.5 Quicksort .....	776
22.3.6 qsort() .....	782
22.3.7 Zusammenfassung der Sorteralgorithmen .....	784
22.4 Suchalgorithmen – Grundlage zur Suche .....	791
22.4.1 Lineare Suche .....	792
22.4.2 Binäre Suche .....	794

22.4.3	Binäre (Such-)Bäume .....	796
22.4.4	Elemente im binären Baum einordnen .....	799
22.4.5	Binäre Bäume traversieren .....	804
22.4.6	Löschen eines Elements im binären Baum .....	805
22.4.7	Ein binärer Suchbaum in der Praxis .....	808
22.4.8	Binäre Suchbäume mit Eltern-Zeiger und Threads .....	817
22.4.9	Ausgeglichene Binärbäume .....	818
22.4.10	Algorithmen für ausgeglichene Bäume – eine Übersicht	818
22.5	Hashing (Zerhacken) .....	819
22.5.1	Wann wird Hashing verwendet? .....	820
22.5.2	Was ist für das Hashing erforderlich? .....	820
22.5.3	Hash-Funktion .....	824
22.5.4	Hashing mit direkter Adressierung .....	829
22.5.5	Vergleich von Hashing mit binären Bäumen .....	829
22.6	String-Matching .....	830
22.6.1	Brute-Force-Algorithmus .....	831
22.6.2	Der Algorithmus von Knuth/Morris/Pratt (KMP) .....	833
22.6.3	Weitere String-Matching-Algorithmen .....	840
22.7	Pattern Matching (reguläre Ausdrücke) .....	841
22.8	Backtracking .....	847
22.8.1	Der Weg durch den Irrgarten .....	847
22.8.2	Das 8-Dame-Problem .....	860

## 23 CGI mit C ..... 869

23.1	Was ist CGI? .....	869
23.2	Vorteile von CGIs in C .....	869
23.3	Andere Techniken der Webprogrammierung .....	870
23.4	Das dreistufige Webanwendungsdesign .....	871
23.4.1	Darstellungsschicht .....	871
23.4.2	Verarbeitungsschicht .....	872
23.4.3	Speicherschicht .....	872
23.5	Clientseitige Programmierung .....	873
23.5.1	JavaScript .....	873
23.5.2	Java-Applets .....	873
23.6	Serverseitige Programmierung .....	873
23.7	Der Webserver .....	874
23.7.1	Das Client/Server-Modell des Internets .....	874
23.7.2	Serverimplementierung .....	875
23.7.3	Hosting-Services .....	876
23.7.4	Schlüsselfertige Lösung .....	876

23.7.5	Weitere Möglichkeiten .....	877
23.7.6	Apache .....	877
23.8	Das HTTP-Protokoll .....	888
23.8.1	Web-Protokolle .....	888
23.8.2	Wozu dienen Protokolle? .....	888
23.8.3	Was ist ein Protokoll? .....	889
23.8.4	Normen für die Netzwerktechnik .....	889
23.8.5	Das OSI-Schichtenmodell .....	889
23.8.6	Die Elemente einer URL .....	890
23.8.7	Client-Anfrage – HTTP-Request (Browser-Request) ....	892
23.8.8	Serverantwort (Server-Response) .....	895
23.8.9	Zusammenfassung .....	898
23.9	Das Common Gateway Interface (CGI) .....	898
23.9.1	Filehandles .....	898
23.9.2	CGI-Umgebungsvariablen .....	899
23.9.3	CGI-Ausgabe .....	904
23.10	HTML-Formulare .....	907
23.10.1	Die Tags und ihre Bedeutung .....	907
23.11	CGI-Eingabe .....	913
23.11.1	Die Anfrage des Clients an den Server .....	913
23.11.2	Eingabe parsen .....	917
23.12	Ein Gästebuch .....	922
23.12.1	Das HTML-Formular (»guestbook.html«) .....	922
23.12.2	Das CGI-Programm (»auswert.cgi«) .....	924
23.12.3	Das HTML-Gästebuch (»gaeste.html«) .....	932
23.13	Ausblick .....	933
<b>24</b>	<b>MySQL und C .....</b>	<b>935</b>
24.1	Aufbau eines Datenbanksystems .....	935
24.1.1	Warum wurde ein Datenbanksystem (DBS) entwickelt? 935	935
24.1.2	Das Datenbank-Management-System (DBMS) .....	936
24.1.3	Relationale Datenbank .....	939
24.1.4	Eigene Clients mit C für SQL mithilfe der ODBC-API entwickeln .....	939
24.2	MySQL installieren .....	940
24.2.1	Linux .....	940
24.2.2	Den Client »mysql« starten .....	941
24.3	Crashkurs (My)SQL .....	943
24.3.1	Was ist SQL? .....	944
24.3.2	Die Datentypen von (My)SQL .....	944

24.3.3	Eine Datenbank erzeugen .....	946
24.3.4	Eine Datenbank löschen .....	947
24.3.5	Datenbank wechseln .....	948
24.3.6	Eine Tabelle erstellen .....	948
24.3.7	Die Tabelle anzeigen .....	949
24.3.8	Tabellendefinition überprüfen .....	949
24.3.9	Tabelle löschen .....	950
24.3.10	Struktur einer Tabelle ändern .....	950
24.3.11	Datensätze eingeben .....	951
24.3.12	Datensätze auswählen .....	951
24.3.13	Ein fortgeschrittenes Szenario .....	952
24.3.14	Datensatz löschen .....	954
24.3.15	Datensatz ändern .....	954
24.3.16	Zugriffsrechte in MySQL .....	954
24.3.17	Übersicht über einige SQL-Kommandos .....	955
24.4	Die MySQL-C-API .....	957
24.4.1	Grundlagen zur Programmierung eines MySQL-Clients .....	957
24.4.2	Client-Programm mit dem gcc unter Linux und dem Cygwin-gcc-Compiler unter Windows .....	958
24.4.3	MySQL Client-Programme mit dem VC++ Compiler und dem Borland Freeware Compiler .....	959
24.4.4	Troubleshooting .....	961
24.4.5	Das erste Client-Programm – Verbindung mit dem MySQL-Server herstellen .....	961
24.4.6	MySQL-Kommandozeilen-Optionen .....	966
24.4.7	Anfrage an den Server .....	969
24.5	MySQL und C mit CGI .....	987
24.5.1	HTML-Eingabeformular .....	987
24.5.2	Die CGI-Anwendung »add_db.cgi« .....	988
24.5.3	Die CGI-Anwendung »search_db.cgi« .....	996
24.6	Funktionsübersicht .....	1004
24.7	Datentypenübersicht der C-API .....	1007

## 25 Netzwerkprogrammierung und Cross-Plattform-Entwicklung 1009

25.1	Begriffe zur Netzwerktechnik .....	1010
25.1.1	IP-Nummern .....	1010
25.1.2	Portnummer .....	1011
25.1.3	Host- und Domainname .....	1012
25.1.4	Nameserver .....	1013

25.1.5 Das IP-Protokoll .....	1013
25.1.6 TCP und UDP .....	1014
25.1.7 Was sind Sockets? .....	1014
25.2 Headerdateien zur Socketprogrammierung .....	1015
25.2.1 Linux/UNIX .....	1015
25.2.2 Windows .....	1016
25.3 Client/Server-Prinzip .....	1018
25.3.1 Loopback-Interface .....	1019
25.4 Erstellen einer Client-Anwendung .....	1019
25.4.1 »socket()« – Erzeugen eines Kommunikations- endpunktes .....	1020
25.4.2 »connect()« – ein Client stellt eine Verbindung zum Server her .....	1022
25.4.3 Senden und Empfangen von Daten .....	1027
25.4.4 »close()« und »closesocket()« .....	1029
25.5 Erstellen einer Server-Anwendung .....	1030
25.5.1 »bind()« – Festlegen einer Adresse aus dem Namensraum .....	1030
25.5.2 »listen()« – Warteschlange für eingehende Verbindungen einrichten .....	1032
25.5.3 »accept()« und die Serverhauptschleife .....	1033
25.6 (Cross-Plattform-)TCP-Echo-Server .....	1036
25.6.1 Der Client .....	1036
25.6.2 Der Server .....	1039
25.7 Cross-Plattform-Development .....	1043
25.7.1 Abstraction Layer .....	1043
25.7.2 Headerdatei für Linux/UNIX .....	1043
25.7.3 Linux/UNIX-Quellcodedatei .....	1044
25.7.4 Headerdatei für MS-Windows .....	1048
25.7.5 Windows-Quellcodedatei .....	1048
25.7.6 All together – die »main«-Funktionen .....	1052
25.7.7 Ein UDP-Beispiel .....	1056
25.7.8 Mehrere Clients gleichzeitig behandeln .....	1058
25.8 Weitere Anmerkungen zur Netzwerkprogrammierung .....	1066
25.8.1 Das Datenformat .....	1066
25.8.2 Der Puffer .....	1067
25.8.3 Portabilität .....	1068
25.8.4 Von IPv4 nach IPv6 .....	1068
25.8.5 RFC-Dokumente (Request for Comments) .....	1070
25.8.6 Sicherheit .....	1070

**26 Paralleles Rechnen ..... 1071**

26.1	Parallelität .....	1071
26.1.1	Single-Prozessorsysteme .....	1072
26.1.2	Hyperthreading .....	1072
26.2	Programmiertechniken der Parallelisierung .....	1073
26.2.1	Automatische Parallelisierung .....	1073
26.2.2	Halbautomatische Parallelisierung .....	1074
26.2.3	Echte Parallelisierung .....	1074
26.3	Vom Prozess zum Thread .....	1075
26.4	Mit den POSIX-Threads programmieren .....	1078
26.4.1	Ein serielles Beispiel .....	1078
26.4.2	Das Grundgerüst für ein Programm mit mehreren Threads .....	1080
26.4.3	Zusammenfassung .....	1086

**27 Sicheres Programmieren ..... 1087**

27.1	Buffer-Overflow (Speicherüberlauf) .....	1088
27.1.1	Speicherverwaltung von Programmen .....	1090
27.1.2	Der Stack-Frame .....	1091
27.1.3	Rücksprungadresse manipulieren .....	1092
27.1.4	Gegenmaßnahmen zum Buffer-Overflow während der Programmerstellung .....	1098
27.1.5	Gegenmaßnahmen zum Buffer-Overflow, wenn das Programm fertig ist .....	1101
27.1.6	Programme und Tools zum Buffer-Overflow .....	1104
27.1.7	Ausblick .....	1105
27.2	Memory Leaks (Speicherlecks) .....	1105
27.2.1	Bibliotheken und Tools zu Memory Leaks .....	1109
27.3	Tipps zu Sicherheitsproblemen .....	1110

**28 Wie geht's jetzt weiter? ..... 1113**

28.1	GUI-Programmierung – grafische Oberflächen .....	1114
28.1.1	Low-Level-Grafikprogrammierung .....	1114
28.1.2	High-Level-Grafikprogrammierung .....	1115
28.1.3	Multimedia-Grafikprogrammierung .....	1116

<b>Anhang .....</b>	<b>1117</b>
A Operatoren .....	1119
A.1 Rangfolge der Operatoren .....	1119
A.2 ASCII-Code-Tabelle .....	1121
A.3 Reservierte Schlüsselwörter in C .....	1122
A.4 Standard-Headerdateien der ANSI-C-Bibliothek .....	1122
A.5 Weiterführende Links .....	1122
B Die C-Standard-Bibliothek .....	1123
B.1 <assert.h> .....	1123
B.2 <complex.h> (C99) .....	1123
B.3 <ctype.h> .....	1127
B.4 <errno.h> .....	1128
B.5 <fenv.h> (C99) .....	1129
B.6 <float.h> .....	1131
B.7 <inttypes.h> (C99) .....	1133
B.8 <iso646.h> (NA1) .....	1135
B.9 <limits.h> .....	1135
B.10 <locale.h> .....	1136
B.11 <math.h> .....	1139
B.12 <setjmp.h> .....	1145
B.13 <signal.h> .....	1146
B.14 <stdarg.h> .....	1147
B.15 <stdbool.h> (C99) .....	1148
B.16 <stddef.h> .....	1148
B.17 <stdint.h> (C99) .....	1149
B.18 <stdio.h> .....	1150
B.19 <stdlib.h> .....	1156
B.20 <string.h> .....	1160
B.21 <tgmath.h> (C99) .....	1162
B.22 <time.h> .....	1163
B.23 <wchar.h> (NA1) .....	1165
B.24 <wctype.h> (NA1) .....	1172
<b>Index .....</b>	<b>1175</b>