

Auf einen Blick

Über den Autor	11
Einleitung	29
Teil I: Das erste Programm	35
Kapitel 1: In zehn Schritten zum ersten Compiler	37
Kapitel 2: Der (zumeist harmlose) Einstieg	51
Kapitel 3: Einfache C-Programme basteln	81
Kapitel 4: Variablen und Mathe	115
Teil II: Grundlegendes Sprachverständnis	165
Kapitel 5: Wir stehen vor einer großen Entscheidung	167
Kapitel 6: Ihre persönlichen Funktionen	207
Kapitel 7: Feinschliff für die C-Künste	253
Kapitel 8: Schleifen knüpfen	291
Kapitel 9: Zwischenstand und Reste essen	321
Teil III: Einfache Datenstrukturen und Zeiger	349
Kapitel 10: Arrays und Strings	351
Kapitel 11: Strings und so Zeugs	377
Kapitel 12: Die finstersten Aspekte von C: Zeiger	411
Kapitel 13: Erste Anwendungen der Zeiger	451
Kapitel 14: Alles über Strukturen	497
Teil IV: Daten speichern und verwalten	543
Kapitel 15: Die Festplatte als Diener	545
Kapitel 16: Dynamische Datenstrukturen	575
Kapitel 17: Die Geburt unserer Datenbank	591
Teil V: Der Top-Ten-Teil	607
Kapitel 18: Cehn Gründe für C im Jahr 2020, 2021, 2022	609
Kapitel 19: Zehn Empfehlungen zum Schreiben unlesbarer Programme	613
Kapitel 20: Zehn nützliche Internetadressen zu C	619
Stichwortverzeichnis	625

Inhaltsverzeichnis

Über den Autor	11
Einleitung	29
Was bringt es, C zu lernen?	29
Über dieses Buch.	30
Programme in diesem Buch.	30
Törichte Annahmen über den Leser.	31
Wie dieses Buch aufgebaut ist.	32
Teil I: Das erste Programm	32
Teil II: Grundlegendes Sprachverständnis	32
Teil III: Einfache Datenstrukturen und Zeiger	32
Teil IV: Daten speichern und verwalten	32
Teil V: Der Top-Ten-Teil	32
Symbole, die in diesem Buch verwendet werden.	33
Schlussgedanken	33
TEIL I	
DAS ERSTE PROGRAMM	35
Kapitel 1	
In zehn Schritten zum ersten Compiler	37
Die Installation der IDE	37
Code::Blocks installieren	38
Die Compilereinstellungen	40
Ihr erstes Programm	43
Welches C hätten Sie gern?	43
An die Tastatur!	44
Den C-Code eingeben	46
Ihr Programm erstellen	48
Ihr Programm ausführen	48
Ein Wort zu Code::Blocks und Spenden	49
Kapitel 2	
Der (zumeist harmlose) Einstieg	51
Die kurze und banale Geschichte von C	51
Wie aus einer süßen kleinen Textdatei ein Programm wird	52
Der Entwicklungsprozess in C	53
Der Quelltext (oder auch Sourcecode)	54
Der Compiler.	54
Der Linker	56
Das erste C-Programm – eine alte Tradition	56
Speichern! Kompilieren! Linken! Starten!	57

16 Inhaltsverzeichnis

Die notwendigen Editier- und Kompilierkünste	58
Die Quelltextdatei ändern.	59
Neukompilierung (oder: Spiel's nochmal in der Sprache C).	60
Raus mit den alten Sachen, lasst uns was Neues machen.	60
Mit Fehlern auf Du und Du.	61
Mist! Ein Fehler. Aber bevor Sie aus dem Fenster springen	61
Junge, was 'n Fehler!	63
Das fehlerhafte Programm reparieren.	64
Die fürchterlichen Linkerfehler	65
Linkerfehler beheben	66
Wie die Sprache C aussieht	67
Der große Zusammenhang.	67
Stückliste	69
Die Sprache C mit ihren Schlüsselwörtern.	70
Andere Sprachelemente von C.	71
Eingaben und Ausgaben (die hier mal was Gutes sind).	72
Stellen Sie sich bei Herrn Computer vor.	72
Die Belohnung!	73
Erst das Chaos, dann die Ordnung	75
Der C-Compiler räumt sein Zimmer nicht auf	75
Was aufheben, was wegwerfen?	75
Organisation ist alles!	76
Wichtige C-Regeln, an die Sie nie denken werden	77
Das hilfreiche Regelprogramm	77
Zeit für eine Bonusrunde	78

Kapitel 3

Einfache C-Programme basteln.	81
printf (wer bei print an Drucken denkt, liegt falsch).	81
Ein anderes »printf macht was Lustiges«-Programm	82
Noch mehr lustige Texte.	83
Einige Escape-Sequenzen	85
Das f steht für formatiert	85
Richt' Euch!	86
scanf – lassen Sie Ihren Scanner trotzdem aus!	88
scanf richtig einsetzen.	89
scanf im scharfen Einsatz	91
Kleine Änderungen an Whoru.c	92
Das Wunder von %s.	93
Zeit für Experimente	93
Bemerkungen, Kommentare und Vorschläge	95
Darf ich um einen kurzen Kommentar bitten?	95
Kommentarstile	96
Wieso sind Kommentare notwendig?	96
Bizarre Kommentare	97
Kommentare als Schalter	98
Vermeidung »verschachtelter« Kommentare	99

fgets und puts	100
Und tschüss scanf, willkommen fgets	101
Ein unfreundliches Programmbeispiel	101
Das finstere Geheimnis der Sprache C	101
Sichere Dateneingaben	102
put – put – putputput – puts	103
Noch eine doofe Spielerei	104
puts und Variablen	105
Licht! Kamera! Action! puts und fgets die Zweite	105
Mehr Spaß mit printf	107
Das alte »Text mit printf ausgeben«	108
Die Escape-Sequenzen von printf	108
Das Deluxe-Testprogramm für Escape-Sequenzen	109
Weitere Tests mit Printfun	110
Wieder mal die Komplexität des printf-Formats	111
Die Konvertierungszeichen von printf	112

Kapitel 4

Variablen und Mathe 115

Die sich ständig ändernde Variable	115
Strings ändern sich	116
Willkommen in der Welt der numerischen Variablen	117
Benutzung der ersten Integervariablen	118
Zuweisung von Werten an numerische Variablen	119
Die Eingabe von Zahlen über die Tastatur	120
Wie alt war Methusalem?	122
Mehr numerische Variablen und ein bisschen Mathe	124
Wieder auf der Jagd	124
Eine Portion Mathe	125
Wie lange brauchen Sie, um Methusalems Rekord zu brechen?	126
Methusalem: Bonusrunde! Jedes Los ein Gewinn!	127
Das unmittelbare Ergebnis	128
Diskurse, Diskussionen und die Definition von Variablen	129
»Warum muss ich eine Variable definieren?«	130
Verbotene und erlaubte Variablennamen	131
Vordefinieren von Variablen	131
Ein x-beliebiges Beispiel für Variablen	133
Mehrfachdeklarationen	134
Konstanten und Variablen	135
Träume von Freiheit und mehr Konstanz	135
Eine praktische Abkürzung	136
Das Schlüsselwort const	137
Der dritte Weg	138
Malen mit Zahlen	140
Zahlen in C	140
Wieso nimmt man int?	142

Mit oder ohne Vorzeichen – das ist hier die Frage	142
Wie bringt man eine Zahl eigentlich zum Fließen?	143
»Hey EIX32A, lass uns mal ein Fließkommaprogramm schreiben!«	144
Die Sache mit der E-Notation	145
Doppelt so groß wie float? Das ist double!	147
Bringen Sie Ihren Zahlen Format bei!	148
Die etwas andere Variable: char	149
Variablen mit einzelnen Zeichen	149
Variablen mit Zeichen abfüllen	151
Kitzelt eine Tasteneingabe den Computer?	151
Zeichenvariablen als Werte	152
Die erste wirkliche Mathestunde	154
Ein sehr knapper Überblick über die Operatoren	154
Das alte »Wie groß bist du?«-Programm	155
Heimtückische Änderungen an unserer Größenberechnung	156
Die hohe Kunst des Inkrementierens	157
Weniger lustig: Gewichtszunahme	157
Bonusprogramm! (Irgendwann nützt es Ihnen was)	158
Wer hat den Vortritt?	160
Ein Beispiel aus einer Examensklausur	160
Punktrechnung vor Strichrechnung	161
Klammern haben den Vortritt	162

TEIL II GRUNDLEGENDES SPRACHVERSTÄNDNIS..... 165

Kapitel 5 Wir stehen vor einer großen Entscheidung..... 167

Die mächtige if-Anweisung	168
Der Computer als Genie	168
Das Schlüsselwort if ganz aus der Nähe	170
Eine Anmerkung zur Schreibweise der if-Anweisung	174
Die endgültige Lösung für die Steuerproblematik	174
Gut, wenn's nicht wahr ist, was ist es dann?	176
Mit einem else alle Möglichkeiten abdecken	176
Eine Anmerkung zur Schreibweise	179
Der Sonderfall else-if und viel mehr Entscheidungen	180
if mit Zeichen und Strings	181
Die zahlenlose Welt von if	181
Was ist größer: S oder T, \$ oder -?	182
if und der Vergleich von Strings	183
Wie schreibt man richtig in C? Lektion 1	183
Immer von oben nach unten	184
Formatierungen des Quellcodes	184
Einrückungen, Teil 2	185
Vergessen wir kurz die Formatierung	186
Andere Möglichkeiten	186

for knüpft Schleifen	187
Wiederholte Wiederholungen	189
for formt die Schleifen	190
Wie in der Grundschule: Zählen bis 100	192
Schleifen fabrizieren	194
Endlich – das nützliche ASCII-Programm	194
Vorsicht, Endlosschleifen!	195
Schleifen gewaltsam abbrechen	197
Das Schlüsselwort break	198
Abkürzungen und die Kunst des Inkrementierens	199
Kryptische C-Operatoren, Teil 1: ++	199
Bahn frei, jetzt komm ich!	200
Rückwärtszählen: nelhäz sträwkcÜR	201
Rückwärtszählen passt sehr gut zum for	202
Kryptische C-Operatoren, Teil 2: --	202
Eine abschließende Verbesserung von Ollyolly.c	203
Je mehr Inkrement, desto mehr Wahnsinn	204
Schleifen mit Sprüngen	204
Kryptische C-Operatoren, Teil 3: Jetzt wird's verrückt	205

Kapitel 6

Ihre persönlichen Funktionen **207**

Die allererste Funktion schreiben	207
Dieses Programm braucht eine Funktion	209
Die nützliche Funktion idiot()	210
Das Tao der Funktionen	211
Zur Namensgebung von Funktionen	212
Ein Wort zum Prototyping	213
Variablen innerhalb von Funktionen	215
Bomben frei mit dem Bomber-Programm	215
Bringt die doppelte Variable Bomber.c zur Explosion?	216
Ein weiterführendes Beispiel	217
Brüderliches Teilen von Werten mithilfe globaler Variablen	219
Eine globale Variable erzeugen	219
Ein Beispiel für eine globale Variable	220
Funktionen ein Päckchen mit auf den Weg geben	222
Funktionen mit einer echten Funktion	224
Wie man einer Funktion einen Wert schickt	225
Unbedingt lesen: Variablenwirrwarr vermeiden	226
Mehr als einen Wert an eine Funktion übergeben	226
Wie man Strings an Funktionen übergibt	228
Funktionen mit Ergebnissen	230
Endlich sagt der Computer mal was Gutes über Sie	231
Fehlerbeseitigung in lq.c mit dem beliebten Typecasting-Trick	232
Mit return zurück an den Absender	234
Gib dem Mann mal einen Bonus!	235

Das Zwiebelschalenprinzip	237
Funktionen arbeiten von innen nach außen	238
Veränderungen am Programm lq.c.	239
Ein gängiges Beispiel für if	240
Das alte Spiel mit den Zufallszahlen	241
Benutzung der rand-Funktion	242
Zufallszahlen initialisieren	244
Mehr Zufall im Zufallsprogramm.	245
Verbesserungen am Zufallszahlenprogramm	246
Der teuflische Dr. Modulo.	248
Die Würfel sind gefallen	250

Kapitel 7

Feinschliff für die C-Künste	253
Dieses Zeug drängelt sich immer vor (der Rest folgt dem #)	253
Bitte vergiss mein nicht!	254
Ihre eigene h-Datei erstellen	257
Wofür #define gut ist.	259
Makros erwähnen wir besser erst gar nicht.	260
Wie man Eingaben vom Benutzer bekommt.	262
Die Tastatur abhören.	262
Das fürchterliche Ratespiel.	263
Die Shell beklaugen (wie man die Kommandozeile abhört)	265
Da gehen einem die Argumente aus – main passiert das nie	266
Also, irgendwelche Argumente dafür?	267
Anzeige der Argumente.	268
Wiederholungen sind gut	270
Ihr Gewicht auf dem Mond.	272
Noch mehr ifs.	274
Auswertung von mehreren Bedingungen in if-Abfragen.	275
Wer ist schlauer? Melvin oder Poindexter?	277
Gutes p, schlechtes P.	279
Weitere Weisheiten	280
Nun zu den schlechten Nachrichten	280
Mehr über Variablen	281
Typecasting und andere Probleme	281
Grundlagen der Typumwandlung	282
Die lange und die kurze Form	282
Die Konfusion weicht allmählich	283
signed und unsigned	284
unsigned Variablen haben eine negative Einstellung.	285
Faire und unfaire Variablen	286
Ein anderes Kapitel über Zahlen (die Sache mit der Hexerei).	286
Zahlensysteme und ihre Basis	287
Hexereien	288

Kapitel 8	
Schleifen knüpfen	291
Fakten zu while-Schleifen	291
Die erste Schleife binden	292
Das Schlüsselwort while	293
Was ist besser geeignet: while oder for?	295
Tausche unschönes for(;;) gegen elegantes while	296
Den Spieß umgedreht: do-while-Schleifen	297
while macht Kopfstände – do-while	298
De-tail von do-while	300
Trau niemals dem User – ein Fehler in Countdown.c	301
Die garantiert richtige Eingabe	302
Der bizarre Fall while TRUE (und sonstige Kuriositäten)	303
Fragestunde	303
Feintuning für Yorn.c	305
»Kochen, bis es gar ist«	306
Yorn.c als Funktion	307
Verschachtelte Schleifen und anderer Unsinn für Angeber	309
Verschachtelte Anwendungen	310
Schleifen und das heimtückische switch-case	311
Die Lösung mit switch-case	313
Der alte Trick mit switch-case	315
Eine besondere Beziehung zwischen while und switch-case	318
Kapitel 9	
Zwischenstand und Reste essen	321
Selbsttest via Hallo, Welt!	321
Das Hallo-Welt-Programm	322
Weiter geht's – ein närrisches Hallo-Welt-Programm	322
Halt die Welt an, ich möchte aussteigen!	323
Die Behandlung der Überbleibsel	326
Die Lösung mittels while-Schleife	328
Mehr über die math.h-Library	329
Die Probleme in einer höheren Potenz	330
Autsch! Wurzeln ziehen	333
Lästige Mathematik? – Sie packen das	334
Etwas wahrhaft Merkwürdiges zum Schluss	335
Die Gefahren der Benutzung von a++	336
Ja, und das Gleiche gilt für --	337
Wiederholung des ++a-Phänomens	338
goto-Anweisung – nein danke	338
Was ist nun goto?	339
Ein Beispiel, wie man goto nicht verwenden sollte	340
Alternativen zu goto	342

Und tschüss ... – das Programm verlassen	344
exit – der Notausgang	344
Zuletzt das berüchtigte Ask-Programm	345
Eine sinnlose Batchdatei als Beispiel	348

TEIL III
EINFACHE DATENSTRUKTUREN UND ZEIGER **349**

Kapitel 10
Arrays und Strings **351**

Wozu Arrays?	351
Wie man Arrays benutzt	352
Ein einfaches Programm, bevor es zu langweilig wird	353
Arrays in C erstellen	354
Auf die Elemente eines Arrays zugreifen	356
Werte an ein Array zuweisen	357
Alt, aber brauchbar	359
Strings und Arrays	362
Das Ende eines Strings (das Nullbyte)	363
Köder auslegen	364
Die Falle	366
Gegen den Buffer-Overflow	367
Arrays jeder Sorte	369
Sortier mich, aber schnell	369
Sortieren	371
Eine unverschämt große Menge Zahlen sortieren	373

Kapitel 11
Strings und so Zeugs **377**

Strings und Zeichen	377
Das Programm »Mach mich GROSS«	378
Eine bessere Möglichkeit, einen String zu durchlaufen	380
Ein total verdrehter String	382
Texte verflechten	384
Strings in C kopieren	386
Strings zerlegen	387
Sesam öffne dich	390
Stringmanipulationen	392
Strings verbinden	392
Die Stringfunktionen zusammengefasst	395
Arrays jenseits der ersten Dimension	395
Vereinbarung eines zweidimensionalen Arrays	396
Vereinbarung eines initialisierten zweidimensionalen Arrays	397
Zugriff auf die Elemente eines zweidimensionalen Arrays	399

Ein Array aus Strings	400
Arrays in der Grauzone	404
Dreidimensionale String-Arrays	406
Wieder hinter den sieben Bergen bei den sieben Zwergen	407
Kapitel 12	
Die finstersten Aspekte von C: Zeiger	411
Ein unhandliches und starres Speicherkonzept	411
Land kaufen, aber kein Haus bauen	412
Mehr oder weniger gefräßige Variablen	414
Die Größe eines Arrays berechnen	416
Lange Rede, kurzer Sinn	417
Adresse, Adresse, Adresse	418
Ein etwas simples Programm als brauchbares Beispiel	418
Hallo, Herr Zeiger!	420
Das zweite Metric-Programm	421
Die Vereinbarung der Zeigervariablen	422
Zeiger zum Speichern von Adressen benutzen	423
Was weiß nun das Programm?	423
Oh Schreck – eine kleine Übung zur Selbstkontrolle	425
Nun die Adresse der letzten Variablen	425
Mehr Zeiger, mehr Speicher, Wahnsinn ohne Ende	427
Römische Zahlen (und wieder eine Ausnahme)	430
Noch mehr Römer	432
Das total verdorbene Römer-Programm	433
Und nun das Sternchen, bitte schön	434
Rückblick	434
Zeiger, die zeigen und schauen	435
Der Zeiger als neugieriger Postbote	435
Noch mehr Durcheinander mit den *-Zeigern	437
Ein Zeiger und mehrere Variablen	438
Zeiger und Arrays	439
Ein Array mit einem Zeiger durchlaufen	440
Nun eine wahrhaft schwer zu knackende Nuss	442
Zeiger, Klammern, Arithmetik – puh	443
Zeiger sollen Arrays nicht ersetzen	444
Zeit, einen Schnitt zu machen	445
Die Beziehung zwischen Zeigern und Array-Klammern	446
Beseitigung aller Spuren der Array-Notation	447
Ungläubige können ja nachprüfen	447
Weitere Prüfung	448
Zum guten Schluss: Der Hauptunterschied zwischen Zeigern und Arrays	448

Kapitel 13	
Erste Anwendungen der Zeiger	451
Zeiger und Strings	451
Eine unmögliche Art, einen String auszugeben	451
Eine bessere Möglichkeit, einen String anzuzeigen	453
Mehr Tricks	454
Noch mehr Tricks	454
Ein allerletzter Trick	455
Die Tiefen der Bibliotheksdefinitionen	456
Oh! String! String! String!	457
Noch so eine irreführende Sache	458
Vereinbarung eines Strings mit einem char-Zeiger	460
Übergabe von Zeigern an Funktionen	461
Rückblick zu Zeiger-Problemen	461
Botschaften aus dem Jenseits	463
Übergabe von Arrays an und von Funktionen	468
Übergabe eines char-Arrays an eine Funktion	469
Die verbotenen Vokale	470
Übergabe beliebiger Arrays an Funktionen	473
Analyse der <code>caffeine</code> -Funktion	476
Funktionen, die ihre Variablen töten	477
Arrays aus Zeigern	478
Zeiger und Arrays	478
Ein String-Zeiger-Array erzeugen	479
Ein Beispielprogramm	481
Das Zeiger-Array der Kommandozeile	482
Was zum Kuckuck ist das nun?	483
Zeiger (auch in Arrays) sind Variablen	485
Sortieren von Strings mit Zeigern	486
Ein einfaches und falsches Sortierprogramm	486
Was war falsch?	488
Die entsetzlichen Indirekt-Zeiger	488
Sortieren von Strings nach allen Zeichen	491
Kapitel 14	
Alles über Strukturen	497
Mehrfachvariablen	497
Das Leben ohne Strukturen	498
Mit Arrays geht es nicht viel besser	500
Wir brauchen eine Karteikarte	501
Fakten zu Strukturen	503
Noch einmal das Passwort-Programm	506
Arrays aus Strukturen	508
Alles in einem Array	510
Der Rest der Mannschaft von Oz	511
Strukturkomponenten kopieren	514

Strukturen und Funktionen	518
Ein spezieller Rückblick: Vereinbarung einer globalen Variablen	519
Hey, Funktion, hier kommt eine Struktur!	519
Verschachtelte Strukturen	522
Rückgabe einer Struktur aus einer Funktion	524
Großer Dank gilt der Funktion malloc	530
Ein Beispiel ohne zusätzlichen Speicherplatz	531
Oh malloc, gib mir mehr Platz	532
Borgen und nicht wiedergeben ist wie gestohlen	533
Nun die richtige Speicherbereitstellung für Howdy.c	535
Strukturen und (schon wieder!) Zeiger	537
Was Sie brauchen, um eine Struktur im Speicher anzulegen	537
Das erste Programm mit Zeiger und Struktur	538
Eine leere Zeigerhülle	541

TEIL IV DATEN SPEICHERN UND VERWALTEN 543

Kapitel 15 Die Festplatte als Diener 545

Hello Disk!	545
Ein kleines Textstück auf die Platte schreiben	546
Wie die Arbeit mit Dateien funktioniert	547
Etwas aus einer Datei lesen	550
Versehentliches Überschreiben verhindern	552
Binär oder Text – das ist hier die Frage	555
Ein eigenes type-Programm	555
Ein Dump erstellen	558
Formatierte Ein-/Ausgabe	560
Die formatierte Ausgabe	561
Formatierte Eingabe aus Dateien	562
Ein Array in eine Datei schreiben	563
Ein Array aus der Datei lesen	564
Daten lesen und schreiben	566
Das Grundgerüst für ein Programm zum Speichern von Strukturen	566
Eine Struktur in eine Datei schreiben	568
Die Funktion write_info	569
Die read_info-Funktion	571

Kapitel 16 Dynamische Datenstrukturen 575

Dynamische Arrays	575
Arrays mit variabler Länge	576
I' werd' narrisch: variable Stringlängen	578

Kurzer Rückblick zur Datenspeicherung	580
Wie verkettete Listen arbeiten	582
Die erste Struktur	583
Eine Struktur mehr und der Link darauf	584
Und nun der Rest	586
Die NULL kennzeichnet das Ende	588

Kapitel 17

Die Geburt unserer Datenbank	591
Das unvermeidliche Bankkontenprogramm	591
Das BANK-Programm	591
Was bereits funktioniert	595
Datensätze aus Listen löschen	597
Überflüssige Datensätze entfernen	598
Die Funktion deleteAccount()	599
Die Arbeitsweise der Funktion deleteAccount	601
Einmal vom Speicher zur Disk und zurück	602
Die Liste auf Platte speichern	602
Eine verkettete Liste von Platte laden	604

TEIL V

DER TOP-TEN-TEIL	607
-------------------------------	------------

Kapitel 18

Cehn Gründe für C im Jahr 2020, 2021, 2022	609
C++, C#, Java, PHP und die anderen {}	609
C ist schlank	609
C ist Fahren ohne ABS und ESP	610
C ist die Sprache der APIs und Kernel	611
C ist 31337	611
C ist die Sprache der Aufzüge und Kühlschränke	611
C ist allgegenwärtig	612
C ist die Sprache der Schnittstellen	612
C treibt die Dinge voran	612
C ist wichtig	612

Kapitel 19

Zehn Empfehlungen zum Schreiben unlesbarer Programme	613
Lügen Sie in den Kommentaren	613
Verwenden Sie möglichst kurze Variablennamen	614
Nutzen Sie Copy&Paste ausgiebig	614
Stöbern Sie im Thesaurus	614
Legen Sie sich niemals fest	615
Wenn's mit dem Englisch hapert	615
Ziehen Sie Schreibfehler konsequent durch	615

Seien Sie modern und geben Sie Ihren Quellcode frei	616
Finden Sie Workarounds für eigene Fehler	616
Verzichten Sie auf lesbare Codeformatierungen	617

Kapitel 20

Zehn nützliche Internetadressen zu C 619

Stackoverflow	619
A Programmer's Heaven	619
Codeguru.com	619
C for Dummies	620
Nachschlagewerk zur Bibliothek	620
Noch ein Nachschlagewerk	620
Der C-Standard	620
Der Compiler – Code::Blocks	620
Visual Studio Express Editions	621
c++.net	621

Stichwortverzeichnis 625