



by Dan Gookin



WILEY

Wiley Publishing, Inc.

C

ALL-IN-ONE DESK REFERENCE

FOR

DUMMIES®



by Dan Gookin



Wiley Publishing, Inc.

C All-in-One Desk Reference For Dummies®

Published by

Wiley Publishing, Inc.

111 River Street

Hoboken, NJ 07030-5774

Copyright © 2004 by Wiley Publishing, Inc., Indianapolis, Indiana

Published by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Legal Department, Wiley Publishing, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, (317) 572-3447, fax (317) 572-4355, e-mail: brandreview@wiley.com.

Trademarks: Wiley, the Wiley Publishing logo, For Dummies, the Dummies Man logo, A Reference for the Rest of Us!, The Dummies Way, Dummies Daily, The Fun and Easy Way, Dummies.com, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at 800-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Library of Congress Control Number: 2004102339

ISBN: 0-7645-7069-2

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

10/RX/QY/QU/IN



About the Author

Dan Gookin has been writing about technology for 20 years. He has contributed articles to numerous high-tech magazines and written more than 90 books about personal computing technology, many of them accurate.

He combines his love of writing with his interest in technology to create books that are informative and entertaining, but not boring. Having sold more than 14 million titles translated into more than 30 languages, Dan can attest that his method of crafting computer tomes does seem to work.

Perhaps Dan's most famous title is the original *DOS For Dummies*, published in 1991. It became the world's fastest-selling computer book, at one time moving more copies per week than the *New York Times* number-one best seller (although, because it's a reference book, it could not be listed on the *NYT* best seller list). That book spawned the entire line of *For Dummies* books, which remains a publishing phenomenon to this day.

Dan's most recent titles include *PCs For Dummies*, 9th Edition; *Buying a Computer For Dummies*, 2005 Edition; *Troubleshooting Your PC For Dummies*; *Dan Gookin's Naked Windows XP*; and *Dan Gookin's Naked Office*. He publishes a free weekly computer newsletter, "Weekly Wambooli Salad," and also maintains the vast and helpful Web site www.wambooli.com.

Dan holds a degree in communications and visual arts from the University of California, San Diego. He lives in the Pacific Northwest, where he enjoys spending time with his four boys in the gentle woods of Idaho.

Publisher's Acknowledgments

We're proud of this book; please send us your comments through our online registration form located at www.dummies.com/register/.

Some of the people who helped bring this book to market include the following:

Acquisitions, Editorial, and Media Development

Project Editor: Rebecca Whitney

Acquisitions Editor: Greg Croy

Technical Editor: Greg Guntle

Editorial Manager: Carol Sheehan

Editorial Assistant: Amanda M. Foxworth

Cartoons: Rich Tennant, www.the5thwave.com

Composition

Project Coordinator: Maridee Ennis

Layout and Graphics: Karl Brandt,
Denny Hager, Joyce Haughey,
Stephanie D. Jumper, Michael Kruzil,
Melanee Prendergast, Jacque Roth,
Julie Trippetti, Mary Gillot Virgin

Proofreaders: Arielle Carole Mennelle,
Dwight Ramsey, Brian H. Walls

Indexer: Infodex Indexing Services Inc.

Publishing and Editorial for Technology Dummies

Richard Swadley, Vice President and Executive Group Publisher

Andy Cummings, Vice President and Publisher

Mary Bednarek, Executive Editorial Director

Mary C. Corder, Editorial Director

Publishing for Consumer Dummies

Diane Graves Steele, Vice President and Publisher

Joyce Pepple, Acquisitions Director

Composition Services

Gerry Fahey, Vice President of Production Services

Debbie Stailey, Director of Composition Services

Contents at a Glance



<i>Introduction</i>	<i>1</i>
<i>Book I: Hello, C.....</i>	<i>7</i>
Chapter 1: Your Basic C Program	9
Chapter 2: How It All Works	19
Chapter 3: More Basics, Comments, and Errors	33
Chapter 4: Introducing Numbers and Variables	47
Chapter 5: More Variables and Basic I/O	61
Chapter 6: Decision Time	79
Chapter 7: Looping	89
Chapter 8: Using Constants	101
Chapter 9: Mysterious Math	109
Chapter 10: It's Only Logical	123
<i>Book II: Middle C</i>	<i>133</i>
Chapter 1: Variables from Beyond Infinity	135
Chapter 2: The Madness of Printf()	149
Chapter 3: Maniacal Math Functions	165
Chapter 4: Not Truly Random	181
Chapter 5: While Going Loopy	193
Chapter 6: More Decision Making	205
Chapter 7: The Goto Chapter	217
<i>Book III: Above C Level.....</i>	<i>221</i>
Chapter 1: Asking for Arrays	223
Chapter 2: I Sing of Strings	243
Chapter 3: Messing with Characters	265
Chapter 4: Stinkin' Structures	271
Chapter 5: Creating Your Own Functions	293
Chapter 6: Quitting Before You're Done	319
Chapter 7: More Variable Nonsense	329

<i>Book IV: Advanced C</i>	347
Chapter 1: Introduction to Evil Pointers	349
Chapter 2: Getting to the *Point	373
Chapter 3: Binary Bits	381
Chapter 4: The Myth of the Array	403
Chapter 5: Pointers and Strings	417
Chapter 6: Crazy Arrays of Pointers	429
Chapter 7: Functions and Pointers	451
Chapter 8: Structures, Pointers, and the Malloc Deity	471
Chapter 9: Does Anyone Have the Time?	485
Chapter 10: Building Big Programs	497
Chapter 11: Help!	511
<i>Book V: Disk Drive C</i>	521
Chapter 1: Just Your Standard I/O	523
Chapter 2: Interacting with the Command Line	537
Chapter 3: Hello, Disk!	547
Chapter 4: More Formal File Writing and Reading	561
Chapter 5: Random Access Files	575
Chapter 6: Folder Folderol	585
Chapter 7: Under New File Management	607
<i>Book VI: The Joy of Linked Lists</i>	615
Chapter 1: Why Linked Lists?	617
Chapter 2: Dawn of the Database	629
Chapter 3: Storing a Linked List on Disk	643
Chapter 4: The Nightmare of the Double-Linked List	655
<i>Book VII: Appendixes</i>	667
Appendix A: The Stuff You Need to Know before Reading	
Everything Else in This Book	669
Appendix B: ASCII Table	681
Appendix C: Answers to Exercises	685
Appendix D: C Language Keywords and Operators	763
Appendix E: C Language Variable Types	767
Appendix F: Escape Sequences	769
Appendix G: Conversion Characters	771
<i>Index</i>	773

Table of Contents

<i>Introduction</i>	1
Why Bother with C When C++ Is Obviously Blah-Blah-Blah?	1
About This Here Dummies Approach	2
How This Book Works	3
Icons Used in This Book	5
Final Thots	5
 <i>Book 1: Hello, C</i>	 7
 Chapter 1: Your Basic C Program	 9
The Section Where the Author Cannot Resist	
Describing the History of C	9
Time to Program!	11
The basic, simplest C program	11
The <code>main()</code> function	12
Inside the <code>main()</code> function	13
“Am I done?”	13
Declaring <code>main()</code> as an <code>int</code>	14
A little sprucing up	15
Finally, making <code>main()</code> do something	16
One more thing!	17
The C Skeleton	18
 Chapter 2: How It All Works	 19
Your Computer Programming Bag of Tools	20
The C Programming Language	20
Keywords	20
Functions	21
Operators	23
Variables and values	23
Other C language goodies	24
Putting It Together in the Editor	24
Making a Program	25
The compiler	25
The object code file	28
The linker	29

Chapter 3: More Basics, Comments, and Errors 33

Simple “Hello” Programs	33
The STOP program	34
Reediting your source code	34
Printing text with <code>printf()</code>	35
Introducing the newline, <code>\n</code>	35
Adding Comments, Remarks, and Suggestions	36
/* C language comments */	36
Using comments to disable code	38
Watch out for nested comments!	39
Fixing a double-quote problem	41
Debugging	42
Prepare to fail	43
Dealing with linker errors	44
Run-time errors and bugs	44
Error messages subtle and gross	45

Chapter 4: Introducing Numbers and Variables 47

Going Numb with Numbers	47
The joys of integers	48
Flying high with floating-point numbers	49
Introduction to Variables	52
Creating variables	52
Assigning values to variables	53
Using an int variable	54
Changing a variable’s contents	55
Using a float variable	56
Using multiple variables	56
Immediately declaring a variable’s value	58
The Official Introduction to Basic Math Operators	59

Chapter 5: More Variables and Basic I/O 61

The Good Ol’ <code>char</code> Variable	61
Presenting the single character	62
Single quotes are holy too	63
Here a <code>char</code> , there a <code>char</code> , everywhere a <code>char char</code>	64
Displaying characters one at a time with <code>putchar()</code>	64
Using <code>char</code> as a tiny integer value	65
Getting Input from the Keyboard	68
Reading the keyboard one key at a time	68
The problem with <code>getchar()</code>	69
Clearing the input stream	71
Reading a chunk of text with <code>gets()</code>	72
How to input numeric values	74
Directly reading values with <code>scanf()</code>	75
Summary of Basic Text I/O Functions	77

Chapter 6: Decision Time	79
Making Decisions with <code>if</code>	79
Are they equal?	80
The old less-than, greater-than puzzle	81
Even more comparisons!	82
<code>else</code> , the Anti- <code>if</code> Statement	83
Or Else!	83
Making Multiple Decisions	84
<code>else if</code> to the rescue!	84
Get the order right!	85
Chapter 7: Looping	89
Presenting the <code>for</code> Loop	89
Dissecting the <code>for</code> loop	90
Counting to 10	91
Counting by twos	93
Endless Loops	93
Your first endless loop	94
A forever loop on purpose	95
Breaking out with <code>break</code>	96
Nesting Loops	97
The 17,576 Names of God	98
Multiple <code>for</code> Conditions	99
Chapter 8: Using Constants	101
Are Constants Necessary?	101
A program example to drive home the point	101
But then, something changes!	102
Chickening out and using variables instead	103
Constants: The Anti-Variable!	103
Declaring a constant	104
Using constants in your code	105
Other Things You Can <code>#define</code>	106
Chapter 9: Mysterious Math	109
Math Review	109
The Sacred Order of Precedence	110
My Dear Aunt Sally	111
What about multiple multiplications?	112
Fooling old Sally with parentheses	113
Say It Out Loud: Unary Operators!	114
Going negative	114
Getting silly	115
Incrementing and Decrementing and Loving It	115
Just add one to it	116
Just take one away from it	117
Pre-incrementing and post-incrementing	117

Other Cryptic Math Shortcuts	119
Incrementing by more than one	119
Multiplying a variable by itself	120

Chapter 10: It's Only Logical123

Comparisons from Hell	123
YORN.C, attempt number 1	123
YORN.C, attempt number 2	124
YORN.C, attempt number 3	125
Here Are Your Logical Operators, Mr. Spock!	127
Introducing Mr. Logical OR	128
Say hello to Mr. Logical AND	129
YORN.C, attempt number 4	131
Multiple Madness with Logical Operators	132

Book 11: Middle C 133

Chapter 1: Variables from Beyond Infinity135

Review of C Language Variable Types	135
The long and the short of it	137
Not long enough	138
Float me to the moon	139
Double or nothing!	140
Signed, Unsigned, Soap, No Soap, Radio	141
Greetings, O Unsigned One	142
Unsigned variables seem to lack that negative-attitude problem	144
One last twisty turn	145
Fair and Unfair Variables	146
Typecasting and Other Acting Problems	146
Pretending that a variable is something it's not	146
Cast away!	147
C Language Variable Reference	148

Chapter 2: The Madness of Printf()149

Going Numb with Numbering Systems	149
Counting bases has nothing to do with baseball	150
Base 2, or binary	151
Base 16, or hexadecimal	151
Displaying hex values with printf()	153
Escape in hex	155
Base 8, or octal	156
It's not a counting base — it's scientific!	157
Scientific notation is for floats only	159
Using immediate scientific notation values	160

Putting <code>printf()</code> to the Test	160
Formatting floating-point numbers	160
Setting the output width for values	161
Padding integers	162
A handy hex trick	163
Justifying text with <code>printf()</code>	163
Other formatting options	164
Chapter 3: Maniacal Math Functions	165
The Symbols That C Forgot	166
The root of the square	166
Pow to the power of pow	167
Ah, the root of the cube!	169
Logarithms	170
Trigonometric Functions	171
The looming terror of radians and degrees	171
Triangle hell	173
Other Handy Math Functions	177
Working on your <code>abs()</code>	177
Getting the <code>abs()</code> of a <code>float</code>	179
Chapter 4: Not Truly Random	181
Introducing the <code>random()</code> Function	182
“Give me 100 really random numbers”	183
“Give me 100 really random numbers without much effort on my behalf this time”	184
The Diabolical Dr. Modulus	185
Introducing the modulo operator	185
What does all this have to do with random numbers?	186
Roll them bones!	187
What number is rolled most often?	188
Flip a coin 1,000 times	190
Chapter 5: While Going Loopy	193
The <code>while</code> Loop	193
Constructing a basic <code>while</code> loop	194
Turn on the incredible shrinking ray!	195
I could <code>while</code> away the hours	196
The popular <code>while(!done)</code> loop	197
The <code>do-while</code> Loop	199
Messing with Loops	201
The endless <code>while</code> loop	201
Continuing loops	202
Nested <code>while</code> loops	203

Chapter 6: More Decision Making205

The Old Switch Case Trick	205
The else-if else-if else-if else-if disease	205
Introducing the cure: switch case	207
The switch case solution	208
Aghgh! No breaks!	209
The switch case while(!done) trick	211
The Weird and Creepy ?: Construct	212
Bonus Program!	213

Chapter 7: The Goto Chapter217

What Now? Go To!	217
The Basic goto Thing	217
Where goto Is Perhaps Needed	219

Book III: Above C Level221

Chapter 1: Asking for Arrays223

Beyond Normal Variables	223
Making an array in C	224
Using an array	225
Declaring an empty array	226
Refilling an array	228
Sorting an Array	229
Sort me, quickly!	229
Sorting things out	230
Sorting an obscenely huge series of numbers	234
Arrays from Beyond the First Dimension!	235
Declaring a two-dimensional array	236
Initializing a two-dimensional array	236
Arrays in the twilight zone	239
Bonus Program!	240

Chapter 2: I Sing of Strings243

The Strings Review	243
Declaring strings	243
Reading and writing strings	244
What you may think you can do, but cannot do, with strings	246
The Truth about Strings	247
Deep inside the character array	247
Terminating a string (the NULL character)	248
Messing with strings	248
Lovely and Handy String Functions	250
An overview of all the string functions	250
Copying a string	251

Comparing two strings	253
Comparing two strings of unequal case	254
Sticking two strings together	254
How long is that string?	256
The Boggling Concept of Arrays of Strings	257
3-dimensional string arrays	259
Fixing the dwarfs	261
Chapter 3: Messing with Characters	265
Introducing the CTYPE Functions	265
Characters That Tell the Truth	267
Just a Trivial Program Example	267
Altering Text	268
The droll “make me uppercase” program	268
Yorn again?	269
Chapter 4: Stinkin’ Structures	271
Life without Structures	271
Part I: The Oz database	272
Part II: The Oz database grows	273
Part III: Arrays are not the answer	275
Multivariables!	277
Presenting the structure	277
The lowdown on structures	278
More stuff	280
Declaring and defining a structure	281
Arrays of Structures	283
Meanwhile, back in Oz	284
Predefining an array of structures	285
Copying one structure element to another	287
Structures for the Birds (Nested Structures)	289
Chapter 5: Creating Your Own Functions	293
Your Typical Function	293
Making a function in C	294
Using functions	295
The importance of prototyping	296
Functions That Don’t Func	297
Using Variables in Functions	298
The concept of local variables	298
Using global variables	300
An example of a global variable in a real, live program	301
Functions That Eat Values	303
Here it comes!	304
How to send more than one value to a function	306
Sending structures to functions	308
How to send strings (or arrays) to a function	310

Functions That Return a Value	311
Returning a value with the <code>return</code> keyword	311
Roll 'dem bones!	312
Functions That Do Both	315
The Land of No Prototyping	316

Chapter 6: Quitting Before You're Done319

Abruptly Leaving the <code>main()</code> Function	319
Many happy returns	319
Sending in your <code>return</code> early	321
Finding the nearest <code>exit()</code>	322
Abort! Abort! Abort!	323
A Most Graceful Exit	324
If we ever get out of here.	325
Registering multiple functions	326

Chapter 7: More Variable Nonsense329

The Joys of Hungarian Notation	329
Beware the <code>typedef</code> Statement!	331
The silly way to use <code>typedef</code>	331
Using <code>typedef</code> with structures	332
Where programmers get into trouble	333
Other Funky Variable Things	334
Auto	335
Const	335
Enum	336
Register	337
Static	337
Volatile	341
The State of the <code>union</code>	342
A simple <code>union</code>	342
Unions buried inside structures	344

Book IV: Advanced C347

Chapter 1: Introduction to Evil Pointers349

Basic Boring Computer Memory Stuff	349
Truly random memory	350
Buying the land, but not building the house	351
Some variables are greedier than others	352
The lowdown on <code>sizeof</code>	353
Calculating the size of an array	353
Location & location & location	355
It all boils down to this	358

Some Pointers	359
The simple definition of a pointer	359
A boring program that's suitable as an example	359
Making Mr. Pointer	360
The converted metric-conversion program	362
Snagging the address of an array	364
The Insanity of Pointer Arithmetic	366
Elementary pointer math	367
More to the pointer	370
Chapter 2: Getting to the *Point	373
Pointer Review	373
And Now, the Asterisk, Please	374
Pointers that point and peek	374
The pointer is a snoopy mailman	375
Using *pointers to Modify Variables	376
Changing a variable's value with a pointer	377
One pointer for everyone to share	378
Chapter 3: Binary Bits	381
Say Hello to Mr. Bit	381
Counting by twos	382
Bits and bytes and words and long words	385
Expressing binary values by using hexadecimal	385
Basic Bit Twiddling	387
Doing the bit hula	388
The official >> format	389
Shifting to the left	390
The Utter Inanity of Binary Logic	391
Using the AND mask	391
One OR in the water	394
Displaying Binary Values	395
The BinString() function	395
Revisiting the logical OR	397
Two Stragglers: ^ and ~	398
The exclusive OR	399
The one's complement	400
Chapter 4: The Myth of the Array	403
Pointers and Arrays	403
Death to the Array!	404
Goodbye, array notation, Part I	404
Using a pointer to fill an array	405
And now, something to truly drive you nuts	406
“What if I want to increment the value and not the memory location?”	408

The Weird Relationship between Pointers and Array Brackets	411
Goodbye, array notation, Part II	411
Removing all vestiges of array notation	413
Proof for disbelievers	414
More proof!	415
Arrays and Pointers Summary	415
Chapter 5: Pointers and Strings	417
Using Pointers to Display Strings	417
The boring way to display a string	417
A better way to display a string	418
More tricks!	420
Even more tricks!	420
One last, crazy trick	421
Distinguishing Strings from Chars	422
The lowdown and letdown of library documentation	422
And, while I'm on the subject of weird strings	425
Declaring a String by Using a Char Pointer	426
Chapter 6: Crazy Arrays of Pointers	429
Introducing the Pointer Array	429
Saving Some Space with String Pointer Arrays	431
Wasting space with string arrays	431
Making a string pointer array	432
A redundant demo program to prove that it works	434
Finding Characters in a Pointer String Array	436
More puzzles, more problems	436
Finding the pointer array without array notation	437
Pointers that point at pointers	439
Replacing the array[a][b] notation with pointers	441
Sorting Strings with Pointers	443
A simple, though incorrect, string-sorting program	444
What went wrong	445
Digging for characters	446
More sorting, more strings	447
Chapter 7: Functions and Pointers	451
Passing a Pointer to a Function	451
Basic pointer-nonsense review	451
Know the differences between pointers and values	452
Functions that eat pointers	453
The glorious advantage of passing a pointer	456
Returning a pointer from a function	457
Arrays to and from Functions	460
Passing an array to a function	460
Down with array notation!	461
How big is that array?	462

Modifying the array in a function	463
Returning an array from a function	464
Strings, Functions, and Pointers	466
Sending a string to a function	467
Returning a string from a function	468
Chapter 8: Structures, Pointers, and the Malloc Deity	471
Making Sacrifices to Malloc	471
A poor example of not allocating any memory	473
Grant me more space, O, Malloc!	474
Properly allocating space for “Howdy, Howdy, Hi”	475
Malloc’s More Useful Relatives	477
Resizing memory chunks with <code>realloc()</code>	477
If you love your RAM, set it <code>free()</code>	478
Using Pointers and Malloc to Make New Structures	480
What you need in order to create a structure	
from thin electrons	481
Your first strange structure and pointer program	482
And now, the rest of the story	484
Chapter 9: Does Anyone Have the Time?	485
No, Seriously: What Time Is It, Really?	485
Julian dates, Julianne fries	486
The Gregorian calendar	486
Greenwich Mean Time and UTC	487
“What is Zulu time?”	487
The epoch	487
Getting the Time	488
What time is it right now?	489
The <code>time()</code> function	490
What time is it right now, REALLY?	490
Getting at the Individual Time-and-Date Pieces’ Parts	491
What’s the day today?	491
Introducing the <code>tm</code> structure	492
At the tone, the time will be.	493
Just a Sec!	494
Chapter 10: Building Big Programs	497
Making Programs with Multiple Modules	497
Banishment to the command prompt!	498
How it works	498
The Tiny, Silly Examples	499
A small, yet potent, example	499
How it works	500
Sharing variables between modules	501
The Big Lotto Program	503
Making a place for your work	503
Building the main module	504

Making your own header file	504
Creating the Init module	506
Creating the Select module	507
Creating the Sort module	508
Creating the Display module	509
Putting the whole thing together	509

Chapter 11: Help!511

Debugging	512
Helpful Utilities	514
ar	515
gdb	515
grep and egrep	515
indent	516
ld	517
lint	517
make	517
touch	519

Book V: Disk Drive C521

Chapter 1: Just Your Standard I/O523

Programming without Any I/O	523
But, What Is Standard I/O?	524
Why use standard I/O devices?	525
How STDIO.H fits into the picture	525
A Demonstration of Standard I/O	526
Input redirection with <	527
Output redirection with >	528
Piping output with 	528
Using with the Underline command	529
Writing Filters	530
The rot13 filter	530
The pig Latin filter	532

Chapter 2: Interacting with the Command Line537

Reading the Command Line	537
The main() function's arguments	538
Counting command-line arguments with argc	539
Reading command-line arguments with argv	539
Testing for command-line arguments	540
Running Another Program with system()	541
Dealing with the Exit Status	542
Coughing up an exit status	543
Reading the exit status in Windows	544
Reading the exit status in Unix	545

Chapter 3: Hello, Disk!	547
Fopen the Ffile, Fplease	547
Writing a small smackerel of something to disk	547
How disk access works in C	548
Reading something from disk	551
Preventing an accidental overwrite	552
Would You Like Binary or Text with That?	554
The View command	555
Taking a dump	556
Chapter 4: More Formal File Writing and Reading	561
Formatted File Input and Output	561
Writing formatted output	562
Reading formatted information from disk	563
Writing an array to disk	564
Reading an array from disk	564
Reading and Writing File Chunks	566
The basic skeleton for your structure-to-disk program	567
Writing a structure to disk	569
Reading a structure from disk	570
The final STOCKS.C program	572
Chapter 5: Random Access Files	575
The Random Access Demonstration	575
Creating the FROOT file	576
Introducing the file pointer	578
Seeking out a specific record	579
Changing a record	581
Starting all over	582
Building a Disk-Based Database	582
Chapter 6: Folder Folderol	585
Who Knows What Lurks on Disk?	585
Grabbing Information about a File with <code>stat()</code>	586
Reading a Directory	588
Opening and closing the directory	588
Reading files from the directory	590
Regular files versus directory files	591
Directories Hither, Thither, and Yon	593
“Where the heck am I?”	593
Pulling up your digs	595
The Art of Recursion	597
The boring recursive demo program	597
A better example of recursion	599
Using recursion to dig through directories	600
Eliminating the <code>.</code> and <code>..</code> entries	602
Finally, adding recursion	603

Chapter 7: Under New File Management	607
Renaming a File	607
Grant that file a new name!	608
Checking for errors when renaming (errno)	608
Deleting a File	610
Copying or Duplicating a File	611
Moving a File (The Secret)	612

Book VI: The Joy of Linked Lists **615**

Chapter 1: Why Linked Lists?	617
A Review of Database Programming in C	617
How Linked Lists Work	619
Building the first structure	620
Adding another structure and linking to it	622
Adding the rest of the structures	624
Marking the end with a NULL	626

Chapter 2: Dawn of the Database	629
The Ubiquitous Bank Account Program	629
The BANK program	630
The promised review that's in order	632
Removing Records from a Linked List	635
Offing deadbeat records	635
Adding the deleteAccount() function	636
How the deleteAccount() function works	639

Chapter 3: Storing a Linked List on Disk	643
From Memory to Disk and Back Again	643
Saving your linked list to disk	644
Reading a linked list from disk	645
How the file-reading routine works	646
The Final Code Listing for BANK.C	648

Chapter 4: The Nightmare of the Double-Linked List	655
The Theory of the Double-Linked List	655
An Example of a Double-Linked List	656
Building the double-linked list	657
Scanning through the double-linked list	660
Deleting an Item from a Double-Linked List	661

Book VII: Appendixes.....667

**Appendix A: The Stuff You Need to Know before
Reading Everything Else in This Book669**

Setting Things Up	669
The C language compiler	669
The place to put your stuff	670
Making Programs	672
Finding your directory or folder	672
Running an editor	674
Compiling and linking	675

Appendix B: ASCII Table681

Appendix C: Answers to Exercises685

Book I: Hello, C	685
Exercise 1.3.1	685
Exercise 1.4.1	686
Exercise 1.4.2	686
Exercise 1.4.3	686
Exercise 1.4.4	686
Exercise 1.5.1	687
Exercise 1.5.2	687
Exercise 1.5.3	688
Exercise 1.5.4	688
Exercise 1.5.5	688
Exercise 1.5.6	689
Exercise 1.5.7	689
Exercise 1.6.1	690
Exercise 1.6.2	690
Exercise 1.6.3	691
Exercise 1.6.4	691
Exercise 1.7.1	692
Exercise 1.7.2	692
Exercise 1.7.3	693
Exercise 1.7.4	693
Exercise 1.7.5	693
Exercise 1.7.6	694
Exercise 1.8.1	694
Exercise 1.8.2	695
Exercise 1.8.3	696
Exercise 1.9.1	696
Exercise 1.9.2	696

Exercise 1.9.3	697
Exercise 1.9.4	697
Exercise 1.9.5	697
Exercise 1.9.6	697
Exercise 1.10.1	698
Exercise 1.10.2	698
Book II: Middle C	698
Exercise 2.1.1	698
Exercise 2.1.2	699
Exercise 2.1.3	699
Exercise 2.1.4	700
Exercise 2.1.5	700
Exercise 2.2.1	700
Exercise 2.2.2	701
Exercise 2.2.3	701
Exercise 2.2.4	702
Exercise 2.2.5	702
Exercise 2.2.6	703
Exercise 2.3.1	703
Exercise 2.3.2	703
Exercise 2.3.3	704
Exercise 2.3.4	704
Exercise 2.3.5	705
Exercise 2.3.6	705
Exercise 2.3.7	705
Exercise 2.3.8	706
Exercise 2.4.1	706
Exercise 2.4.2	707
Exercise 2.4.3	707
Exercise 2.5.1	708
Exercise 2.5.2	708
Exercise 2.5.3	708
Exercise 2.5.4	709
Exercise 2.6.1	709
Exercise 2.7.1	710
Book III: Above C Level	710
Exercise 3.1.1	710
Exercise 3.1.2	711
Exercise 3.1.3	712
Exercise 3.2.1	712
Exercise 3.2.2	713
Exercise 3.2.3	713
Exercise 3.2.4	714
Exercise 3.2.5	714
Exercise 3.3.1	714

Exercise 3.3.2	715
Exercise 3.4.1	716
Exercise 3.4.2	717
Exercise 3.4.3	717
Exercise 3.5.1	718
Exercise 3.5.2	719
Exercise 3.5.3	720
Exercise 3.5.4	721
Exercise 3.6.1	723
Exercise 3.7.1	724
Exercise 3.7.2	724
Book IV: Advanced C	724
Exercise 4.1.1	724
Exercise 4.1.2	724
Exercise 4.1.3	725
Exercise 4.1.4	727
Exercise 4.1.5	727
Exercise 4.2.1	728
Exercise 4.2.2	729
Exercise 4.3.1	729
Exercise 4.3.2	729
Exercise 4.3.3	730
Exercise 4.3.4	730
Exercise 4.3.5	731
Exercise 4.3.6	731
Exercise 4.4.1	732
Exercise 4.4.2	732
Exercise 4.4.3	732
Exercise 4.4.4	733
Exercise 4.5.1	733
Exercise 4.6.1	734
Exercise 4.6.2	734
Exercise 4.6.3	734
Exercise 4.6.4	734
Exercise 4.6.5	734
Exercise 4.6.6	735
Exercise 4.6.7	736
Exercise 4.7.1	737
Exercise 4.7.2	737
Exercise 4.7.3	737
Exercise 4.7.4	738
Exercise 4.7.5	739
Exercise 4.8.1	740
Exercise 4.8.2	741
Exercise 4.9.1	741

Exercise 4.9.2	742
Exercise 4.9.3	742
Exercise 4.9.4	743
Exercise 4.10.1	743
Exercise 4.10.2	744
Book V: Disk Drive C	744
Exercise 5.1.1	744
Exercise 5.1.2	745
Exercise 5.2.1	745
Exercise 5.2.2	746
Exercise 5.2.3	746
Exercise 5.3.1	747
Exercise 5.3.2	748
Exercise 5.3.3	748
Exercise 5.3.4	749
Exercise 5.5.1	749
Exercise 5.5.2	749
Exercise 5.5.3	750
Exercise 5.5.4	751
Exercise 5.6.1	755
Exercise 5.6.2	756
Exercise 5.6.3	756
Exercise 5.6.4	757
Book VI: The Joy of Linked Lists	758
Exercise 6.2.1	758
Exercise 6.2.2	759
Exercise 6.4.1	761
Appendix D: C Language Keywords and Operators	763
Appendix E: C Language Variable Types	767
Appendix F: Escape Sequences	769
Appendix G: Conversion Characters	771
<i>Index</i>	<i>773</i>

Introduction

Congratulations on your purchase of *C All-in-One Desk Reference For Dummies* — a tome that not only sits fat and looks impressive on your computer bookshelf, but also teaches you a heck of a lot about the C programming language.

Because few people read book introductions, I have decided to fill the following six pages with filthy limericks, most of which are patronizing to immigrants and women.

Seriously, now that I have your attention, I thought that I would ramble on briefly about this book and what you can expect from its contents.

This book provides a solid overview of the C programming language, from the basics on up through advanced concepts and topics that third-year university students would pay real money to have someone else suffer through.

Despite the *For Dummies* text on the cover, this book takes a swifter approach to learning the C language than my book *C For Dummies*, 2nd Edition. This massive work assumes, for example, that you may have a wee bit of programming experience or are just more eager to find out more about the C language or perhaps need that extra training in those advanced topics that are skimpily covered in other programming books. If that's you, you have found your book! If that's not you, you should still buy this book because three of my kids need braces badly.

Above all, the bottom line in this book is about having fun. I take things easy, introducing the C language one tidbit at a time. Rare is the long, involved program in this book. Short, punchy (and often silly) programs make finding out about C quick and easy. After all, you need only a few lines of text to see how a function or concept in C works. And, everything is peppered with a modicum of irreverence and a dash of humor. Or, could it be the other way around? Anyway, you get the idea.

Why Bother with C When C++ Is Obviously Blah-Blah-Blah?

The C programming language is like the Latin of the computer world. As with Latin, if you know C, learning other programming languages is a snap. Each of the following programming languages (and many more) has its base in C:

- ◆ C++
- ◆ Perl
- ◆ Java
- ◆ Python

When you know C, learning any of these languages is simple and painless.

Unlike Latin, however, the C language is far from dead. As one of the older computer languages, C has a rich history and a full library of routines, programs, documentation, help, and other whatnot — a rich treasure of resources for you to draw on. Rare is the program that cannot be written using simple C programming. From graphics to games and from networking to building new operating systems, you have no limitation on what you can do in C.

Most of my C language students use my books as a foundation for leaping into C++ programming — mostly because many C++ books lack the gentle hand-holding approach that my books have. In fact, if you read this book from cover to cover, I promise you that any C++ (or other programming language) book will be that much easier for you to grasp. It may not be written with my dynamic wit, but the concepts will be easier to understand.

The bottom line, of course, is *programming*. Becoming a programmer means that you have ultimate control over your computer. *You* are finally in charge, telling the dang thing exactly what to do with itself. And, like the fast idiot it is, the computer dutifully obeys your every whim.

Plus, programming gives you instant feedback. Some folks find that benefit so addicting that they end up growing beards; wearing sandals and Hawaiian shirts; consuming coffee, stale doughnuts and Doritos; and never leaving the confines of their house. And that's just the women!

About This Here Dummies Approach

Don't you hate buying a programming book and having to read through 50 pages of this and that, background information, trivia, why the author thinks he's important, and all that other crap, only to discover that the first program in the book is five pages long and really (honestly) doesn't teach you one single thing about programming?

Yeah! I hate that too!