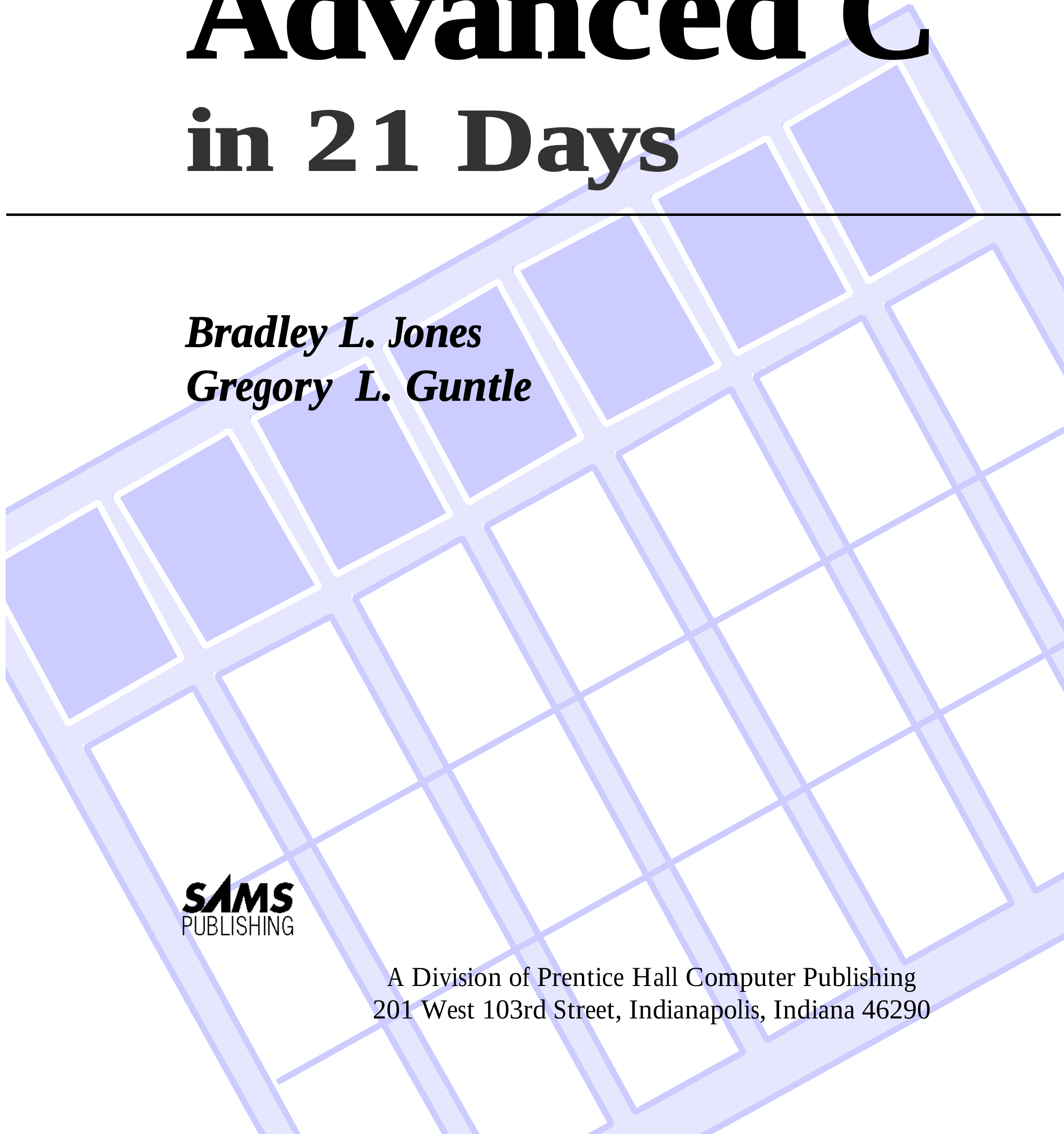
Teach Yourself
Advanced C
in 21 Days



Teach Yourself
Advanced C
in 21 Days

Bradley L. Jones
Gregory L. Guntle

SAMS
PUBLISHING

A Division of Prentice Hall Computer Publishing
201 West 103rd Street, Indianapolis, Indiana 46290

This book is dedicated to all the people who have counted the stars at
Beaubien, walked in the snows of Mt. Baldy in July, or gazed at the
eagles hovering over Ponil. Like learning a programming language,
going to such places takes hard work and dedication.
But once they are reached, the value of the hard work is realized.
To these people, this book is dedicated.

Copyright © 1994 by Sams Publishing

FIRST EDITION
SECOND PRINTING—1994

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein. For information, address Sams Publishing, 201 W. 103rd Street, Indianapolis, IN 46290.

International Standard Book Number: 0-672-30471-6

Library of Congress Catalog Number: 93-87167

97 96 95 94 4 3 2

Interpretation of the printing code: the rightmost double-digit number is the year of the book's printing; the rightmost single digit, the number of the book's printing. For example, a printing code of 94-1 shows that the first printing of the book occurred in 1994.

Composed in AGaramond and MCPdigital by Prentice Hall Computer Publishing.

Printed in the United States of America

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or servicemark.

Publisher
Richard K. Swadley

Associate Publisher
Jordan Gold

Acquisitions Manager
Stacy Hiquet

Acquisitions Editor
Gregory Croy

Development Editor
Dean Miller

Production Editor
Fran Hatton

Copy Editor
Mitzi Foster Gianakos

Editorial Coordinator
Bill Whitmer

Editorial Assistants
Carol Ackerman
Sharon Cox
Lynette Quinn

Technical Editor
Robert Zigon

Marketing Manager
Gregg Bushyeager

Cover Designer
Dan Armstrong

Book Designer
Michele Laseau

Director of Production and Manufacturing
Jeff Valler

Imprint Manager
Juli Cook

Production Analyst
Mary Beth Wakefield

Proofreading Coordinator
Joelynn Gifford

Indexing Coordinator
Johnna VanHoose

Graphics Image Specialists
Tim Montgomery
Dennis Sheehan
Sue VandeWalle

Production
Nick Anderson
Angela D. Bannan
Katy Bodenmiller
Ayrika Bryant
Lisa Daugherty
Stephanie Davis
Kimberly K. Hannel
Angela P. Judy
Greg Kemp
Stephanie McComb
Jamie Milazzo
Wendy Ott
Rochelle Palma
Nancy Sears Perry
Casey Price
Ryan Rader
Kim Scott
Susan Shepard
Tonya R. Simpson
Ann Sippel
SA Springer
Rebecca Tapley
Elaine Webb
Alyssa Yesh

Indexer
Jeanne Clark

About the Authors

Bradley Jones

Bradley Jones is a professional programmer specializing in C development. He has helped in the development of systems for several national and international corporations. As an active member of the Indianapolis Computer Society, he heads the teaching of C and C++ as a leader of the C/C++ SIG. Bradley's other writings include co-authoring the bestseller, *Teach Yourself C in 21 Days* and *Even You Can Soup Up and Fix PCs*. He is the Chairman of the *INDY PC News* magazine, and is also a regular writer on topics ranging from C programming to virtual reality.

Gregory Guntle

Gregory Guntle has been working with microcomputers for 15 years. He has been doing technical editing for nine years. Formerly, he was manager of PC programming at RCI; he now supervises the installation and use of PCs in the company.

M
T
W
R
F
S
S

Overview

Introduction

Week 1 at a Glance 1

Day 1 Storage Space: Working with Memory 3
2 Complex Data Types 31
3 Lists and Trees 67
4 Tapping into System Resources 103
5 Efficiency and Porting 133
6 Number Systems 169
7 Using Libraries 201

Week 2 at a Glance 227

Day 8 Tapping into System Resources via BIOS 229
9 Text Graphics (Using BIOS) 263
10 The *getline()* Function 295
11 Building a Routine Library 331
12 Systems Analysis and Design 353
Specification for *Record of Records!* Version 1.00 369
13 The User Interface: Screen Design 381
14 Enhancing the User Interface: Menuing 443

Week 3 at a Glance 479

Day 15 The User Interface: Action Bars 481
16 Help! 509
17 File Routines: The First Step 531
18 File Routines: Using Dynamic Data 587
19 Reporting 665
20 Finishing Touches: Testing, Debugging, Upgrading, and Maintaining 713
21 Where To Go from Here? 737

Appendix A Memory Models 753
B ASCII Chart 761
C C Keywords 771
D Operator Precedence 777
E ANSI Header Files 781
F Answers 785
Index 851

M
T
W
R
F
S
S

Contents

Introduction

Week 1 at a Glance 1

Day 1 Storage Space: Working with Memory 3

Why Start the 21 Days with Memory Usage? ... 4
Variable Storage Requirements ... 4
  Allocating at Compile Time ... 6
Dynamic Memory Allocation ... 9
  The *malloc()* Function ... 10
  The *free()* Function ... 12
  The *realloc()* Function ... 15
  The *calloc()* Function ... 17
  Allocating More than 64K ... 21
Summary ... 26
Q&A ... 26
Workshop ... 27
  Quiz ... 27
  Exercises ... 28

2 Complex Data Types 31

What Are Data Types? ... 32
  The Values of the Basic Data Types ... 33
  Creating Your Own Data Types ... 34
  Advanced Data Usage: Grouping Data ... 35
  Pointers in Review ... 44
Complex Data Types (or Complex Data Combinations) ... 52
  Pointers to Structures ... 52
  Arrays of Structures ... 55
  Variable-Length Structures ... 58
Summary ... 63
Q&A ... 64
Workshop ... 64
  Quiz ... 64
  Exercises ... 65

3 Lists and Trees 67

Linked Structures ... 68
  Using a Single-Linked List ... 69
  Using a Linked List ... 73
Stacks ... 83
Queues ... 87
  Double-Linked List ... 88

Binary Trees 89
    Using a Binary Tree 90
Summary 99
Q&A 99
Workshop 100
    Quiz 100
    Exercises 101

4 Tapping into System Resources 103

What Are System Resources? 104
Working with the Display 104
    Using ANSI Functions 105
    Using Direct Memory Access 128
    What Is BIOS? 128
Summary 130
Q&A 130
Workshop 131
    Quiz 131
    Exercises 131

5 Efficiency and Porting 133

What Are Efficiency and Portability? 134
Efficiency 134
    Compilers, Computers, and Efficiency 134
    Efficiency Versus Maintainability 134
Portability 138
    The ANSI Standard 139
Summary 164
Q&A 165
Workshop 165
    Quiz 165
    Exercises 166

6 Number Systems 169

The Importance of Number Systems 170
    Deriving the Numbers the Computer Used 170
Which Number Systems Are Important? 176
    The Decimal Number System 179
    Data in Decimal 181
    The Binary Number System 183
    The Hexadecimal Number System 190
    The Octal Number System 194
    A Final Program 196
Summary 198
Q&A 198
Workshop 199
    Quiz 199
    Exercises 200

7 Using Libraries 201

Review of Program Creation 202
How To Work with Multiple Source Files 203
How To Include Header Files with *#include* 204
What Libraries Are 207
  Working with Libraries 208
  How To Create a Library 213
  What Libraries Are Available Already 221
A Final Example
  Using the Edits Library 222
Summary 224
Q&A 224
Workshop 225
  Quiz 225
  Exercises 225

Week 2 at a Glance 227

Day 8 Tapping into System Resources via BIOS 229

System Resources in Review 230
Working with the Display 230
  A Review of Using ANSI Functions 231
  A Review of Using Direct Memory Access 231
  What Is BIOS? 232
  Using BIOS 233
  BIOS and the Cursor 237
Using BIOS for Other Functions 244
Examples of Using BIOS 253
  Compiler-Specific BIOS Functions 256
A Note on Compiler Functions Versus BIOS Functions 257
Summary 258
Q&A 258
Workshop 259
  Quiz 259
  Exercises 260

9 Text Graphics (Using BIOS) 263

Different Levels of Graphics 264
  Character Graphics 265
  Organizing Your Functions 266
Enhancing Your Cursor Functions 269
Creating New Functions 272
  Text Graphics Functions 272
  A Pausing Function 288
Updating Your TYAC Library and Header File 289
Summary 291
Q&A 292

Workshop 292
  Quiz 292
  Exercises 293

10 The *getline()* Function 295

Why Is an Entire Day Dedicated to a Single Function? 296
An Overview of the *getline()* Function 297
  A New TYAC.H Header File 297
  Understanding the Use of *getline()* 302
The Code Behind *getline()* 305
  The *get_it()* Subroutine 318
The *boop()* Function 321
The *waitsec()* Function 322
The *get_timer_ticks()* Function 323
Using *getline()* 325
Summary 327
Q&A 328
Workshop 328
  Quiz 328
  Exercises 329

11 Building a Routine Library 331

Some Important Issues 332
  The *cursor_off()* Function 332
  The *cursor_on()* Function 333
  Clearing the Screen 335
  The *grid()* Function 338
  Saving and Restoring the Screen 341
  Saving and Restoring in Practice 347
Summary 349
Q&A 349
Workshop 350
  Quiz 350
  Exercises 350

12 Systems Analysis and Design 353

Creating a Computer System 354
The Traditional Approach: Structured Systems Analysis and Design 355
  Stage 1: Preliminary Analysis 356
  Stage 2: Systems Analysis (Logical Design) 358
  Stage 3: Systems Design (Physical Design) 361
  Stage 4: Construction 361
  Stage 5: Wrap Up (Testing and Implementation) 362
Rapid or Accelerated Systems Analysis and Design 364
Prototyping 364

A Program Specification 365
    Tracking Questions 366
Summary 366
Q&A 367
Workshop 367
    Quiz 368
    Exercises 368

Specification for *Record of Records!* Version 1.00 369

Purpose 370
Objectives/Capabilities 370
Rules/Assumptions 371
Processing Components 371
Keys/Functions 372
Action Bar Items 374
Special Musical Item Functions 374
Keys/Functions 375
Database Access Requirements 376
    Technical/Implementation Notes 379

13 The User Interface: Screen Design 381

User Interfaces 382
Creating a User Interface 384
    The Benefit of Following Standards 385
    Creating a Common User Interface 386
    The Entry and Edit Standards 386
    The Function Key Standards 388
Creating an Entry and Edit Screen 390
    The Medium Code Entry and Edit Screen 403
    The Group Information Screen 414
    The Musical Items Entry and Edit Screen 425
Synopsis: Where Do You Go from Here? 438
Summary 439
Q&A 439
Workshop 440
    Quiz 440
    Exercises 441

14 Enhancing the User Interface: Menuing 443

What Is Considered a Menu? 444
Displaying a Menu 445
    The Parameters for the *display_menu()* Function 454
    The *display_menu()* Function 456
    Using the *display_menu()* Function 460
    Adding a Menu to *Record of Records!* 464
Using a Menu with Entry and Edit 472

Summary 476
Q&A 476
Workshop 477
  Quiz 477
  Exercises 477

Week Three at a Glance 479

Day 15 The User Interface: Action Bars 481

What Are Action Bars? 482
  Standards in Action Bars 483
A Simple Action Bar in Action 488
A Multi-Menu Action Bar 492
Summary 506
Q&A 506
Workshop 507
  Quiz 507
  Exercises 508

16 Help! 509

Who Needs Help? 510
What Are the Different Types of Help? 510
  Startup Help 511
  General Help 514
  Using an About Box 518
  Context-Sensitive Help 521
  Tutorials 527
  Documentation and Manuals 528
Summary 529
Q&A 529
Workshop 530
  Quiz 530
  Exercises 530

17 File Routines: The First Step 531

Storing Data in a Disk File 532
  A Review of the Terminology Used with Files 532
  File Formats 534
Working with an Indexed File 536
Using a File 538
  Preparation for the New I/O Functions 539
  Opening the FILES: *open_files()* 540
  Getting a Data Record: *get_rec()* 543
  Writing a Data Record: *put_rec()* 545

    Updating the Header Record: *update_header()* 546
    Closing the Files: *close_files()* 548
Working with the Medium Screen 550
    Adding a Record: *The add_medium_rec()* function 566
    Deleting a Record: *The del_med_rec()* function 573
    Processing a Record (Next/Previous) 579
Check Point 583
Summary 583
Q&A 584
Workshop 584
    Quiz 584
    Exercises 585

18 File Routines: Using Dynamic Data 587

File Functions and the Action Bar 588
The Musical Items Screen 602
    Preliminary Changes for the Albums Screen 602
    The Musical Items Screen 604
    Supportive Functions: The SONGS.C Listing 628
    Adding a Record: The *add_album()—rec* Function 635
    Deleting a Record: The *del_alb_rec()* Function 641
    Processing a Record (Next/Previous) 646
The Musical Items Screen's Action Bar 650
Summary 662
Q&A 662
Workshop 663
    Quiz 663
    Exercises 663

19 Reporting 665

Is Reporting Important? 666
Prototyping Reports 666
    Simple List Reports 667
    Complex Reports 669
Creating the Reports 672
    Creating the List of Medium Codes 678
    Creating the List of Group Codes 685
    Creating the Detailed Information Report 691
Building Flexibility into the Reporting 709
Summary 710
Q&A 710
Workshop 710
    Quiz 711
    Exercises 711

20 Finishing Touches: Testing, Debugging, Upgrading, and Maintaining 713

Why Test? 714
  The Complexity in Your Testing 714
  Different Types of Tests 716
  Unit Testing 716
  Integration Testing 717
  Alpha Testing 718
  Beta Testing 718
Resolving Problems 719
  Types of Problems 719
  Debugging 722
When Is an Application Complete? 732
Summary 733
Q&A 734
Workshop 735
  Quiz 735
  Exercises 735

21 Where To Go from Here? 737

What's Next? 738
  Specialized Programming 738
  Other Languages 740
  Other Platforms 744
C Utilities 744
  Graphics Libraries 746
  Communications Libraries 747
  File Management Libraries 747
  Memory Management Libraries 748
  General Libraries 748
Summary 749
Q&A 750
Workshop 750
  Quiz 751
  Exercises 751

Appendix A Memory Models 753

What Are the Memory Models? 754
  The Difference Among Memory Models 754
  The Memory Models 755

B ASCII Chart 761

C C Keywords 771

D Operator Precedence 777

E **ANSI Header Files** **781**

Additionally... 783

F **Answers** **785**

Day 1 Answers 786
Day 2 Answers 788
Day 3 Answers 793
Day 4 Answers 795
Day 5 Answers 800
Day 6 Answers 803
Day 7 Answers 807
Day 8 Answers 811
Day 9 Answers 815
Day 10 Answers 817
Day 11 Answers 820
Day 12 Answers 824
Day 13 Answers 824
Day 14 Answers 826
Day 15 Answers 829
Day 16 Answers 837
Day 17 Answers 838
Day 18 Answers 840
Day 19 Answers 843
Day 20 Answers 848
Day 21 Answers 849

Index **851**

Acknowledgments

Pulling together a book takes a great deal of effort. Many people are involved in the process both directly and indirectly. Many of the people who were involved are listed in the book, several are not.

I would like to thank many of my coworkers for their support and encouragement in my writing of this book. It is helpful to have the support of such people as Connie, Dee, Mike, Paul, Dawn R., and the others.

I would like to thank Fran Hatton. Fran was the production editor who had the task of reading the book and guiding it to its completion. Fran, along with Mitzi Foster Gianakos, offered many good suggestions that helped to make this a better book. In addition, Dean Miller should be given credit for ensuring that I turned the chapters in on time. Greg Croy deserves thanks for thinking of me when this book was proposed.

I would like to especially thank Greg Guntle whose name also appears on this book. With the help of his technical expertise, we were able to bring a new level to this book. His background helped to create many of the functions that you will find extremely useful in your programming endeavors.

I would also like to thank Ray Werner. Ray provided Greg and me the means of communicating back and forth. Ray's Bulletin Board, The Electronic Editor, was a critical factor in the quick completion of this book. I greatly appreciate Ray's help in providing the board. Ray, who is also an author of many computer books, has truly been a valuable friend.

A special thanks goes to you, the reader, for purchasing this book. It is my hope that from this book you will be able to take a basic understanding of C and learn to apply it to developing applications that can be used at a professional level.

A final thanks goes to my wife, Melissa. Because of the short time frames allowed for writing a book, most of my free time has been spent at the computer. Her support in my writing has always been a critical factor in the success of this and my other books.

Introduction

Welcome to *Teach Yourself Advanced C in 21 Days*! This introduction begins the journey into many topics on C. In this introduction, you will learn:

- ☐ Who this book is intended for.
- ☐ Why you should use **this** C book.
- ☐ How the book has been layed out.

Why Advanced C and Why 21 Days?

By picking up this book, you have shown that you are interested in the C programming language. This book has been created to begin teaching where most C books stop. Because the basics of the C language are not covered, this book has been deemed advanced.

This book works through 21 days of material. Why 21 days? Twenty-one days is enough time to cover a great deal of material without taking so long that you, the reader, lose interest. While this book may seem thick, you will find that most chapters will take from one to three hours. The time each chapter will require depends on you. If you simply read the chapter, you will only need about an hour. If you type in the code (although all of it is included on the disk) and if you work the exercises that are provided to help your understanding, then the amount of time needed will increase.

The Assumptions About the Reader

In writing this book, I have made a few assumptions. I have assumed that you are already familiar with the basics of the C language. If you are not, then this book is not for you—at least not yet. As stated before, this book picks up where most C books stop. This book does not teach the basics of C. Nearly any other C book will suffice for providing you beginning knowledge. The book, *Teach Yourself C in 21 Days*, would be a good starting point; however, any beginning level C book will suffice.

The second assumption that I have made is that you don't want to just know the C language, but that you also want to know how to apply it. While you will spend several days covering advanced topics, a majority of this book is dedicated to helping you apply the C language to real applications. It is this application of the C language that helps to differ this book from others. In addition to helping you develop an application, you will learn the concepts involved in development. As an added bonus, during the development process you will create a library of functions that you will find useful.

What This Book Is About

This book starts by building on many of the topics presented in most beginning book. Day 1 jumps right into the material by presenting memory allocation concepts. This is followed by several additional days of advanced topics ranging from linked lists to using system resources.

Warning: The first few days will present concepts that are covered in some beginning C books. The information will differ from what is presented in beginning books in that it will not start at the beginning level. Instead, I have assumed that you already have a basic understanding of the concepts. In addition, they will not stop where most beginning books stop—they will go much farther. You will see how many of the beginning concepts that you already know can become complex.

Farther into the book, you will begin learning development techniques. This will include topics such as tips for writing portable code, creating and using libraries, writing structured computer applications, debugging, and much more.

This book has been designed to aid in your full understanding of using the C language and in applying it to real world applications. You will develop a multitude of functions. You will want to use many of these functions in your C programs.

By working through the developement of a real application, you will understand how to develop your own. In fact, you will easily be able to take what you learn from this book, along with the coding examples, and create your own applications. You could be doing this in just a few weeks.

What You Will Gain from This Book

When you complete this book, you will have a library of useful functions. In addition, you will have an application that you will be able to modify to your own uses. With this application, you will also have the knowledge to develop full-fledged applications on your own.