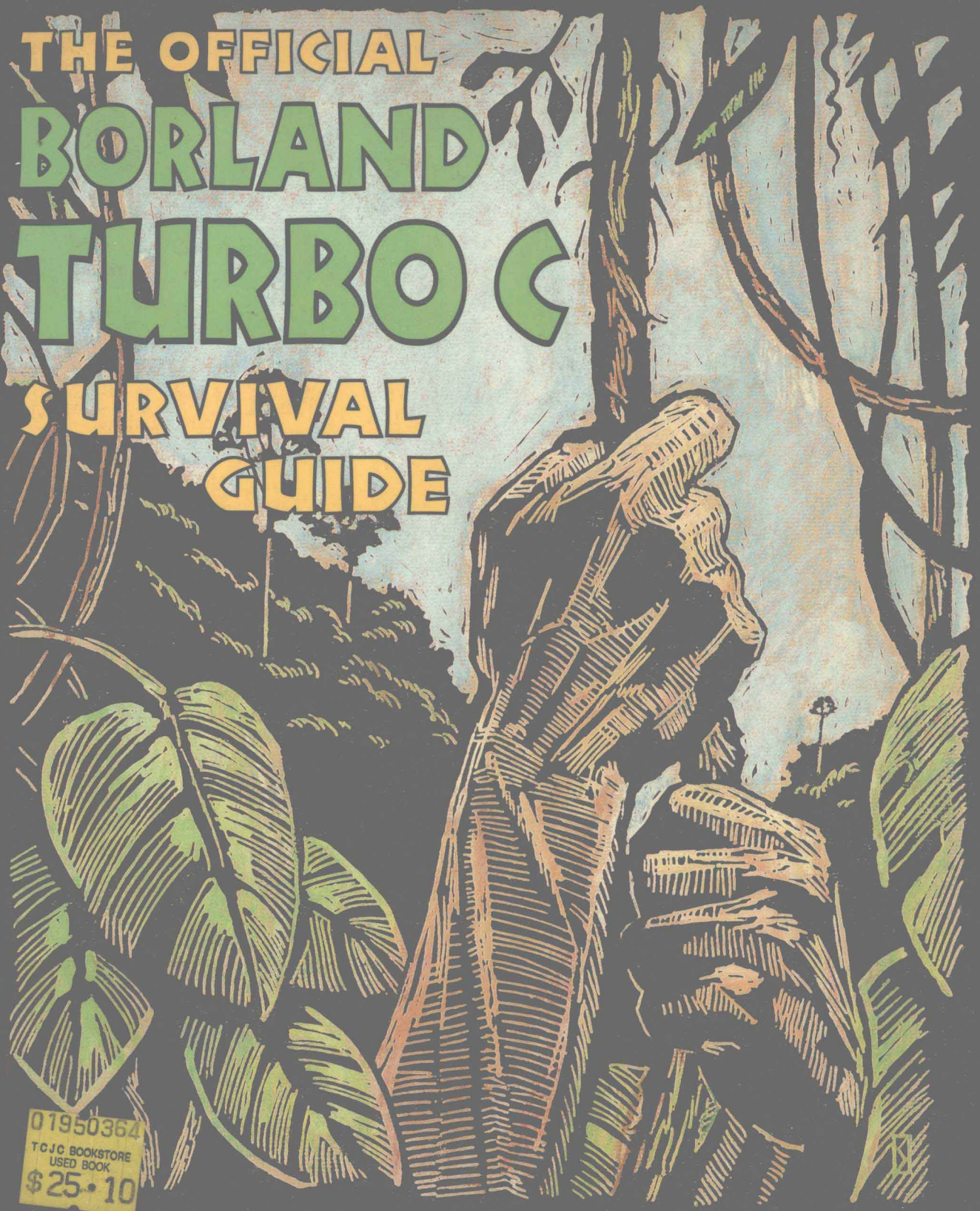
THE OFFICIAL BORLAND TURBO C SURVIVAL GUIDE



LAWRENCE H. MILLER ALEXANDER E. QUILICI

THE OFFICIAL BORLAND TURBO C SURVIVAL GUIDE

Lawrence H. Miller

University of Southern California, Information Sciences Institute

Alexander E. Quilici

University of California, Los Angeles

John Wiley & Sons

New York □ Chichester □ Brisbane □ Toronto □ Singapore

Another one for our families

Cover Illustration: Neil Shigley

Library of Congress Cataloging in Publication Data
Miller, Lawrence H.
The Official Borland Turbo C survival guide / Lawrence H. Miller. Alexander E. Quilici.
p. cm.—(Borland-Wiley Higher Education Series)
Bibliography: p.
Includes index.
ISBN 0-471-60861-0 (pbk.)
1. C (Computer program language) 2. Turbo C (Computer program) I. Quilici, Alexander E. II. Title. III Series.
QA76.73.C15M552 1988
005.13'3—dc19 6pc 88-27608
CIP

The Official Borland Turbo C Survival Guide

Borland-Wiley Higher Education Series

FOREWORD

Within a very short span of time, C has become the programming language of choice for a large number of professional programmers. And among these users, Turbo C has demonstrated the speed, power, flexibility, and accessibility that make it a superior programming environment.

It is clear, then, that there is an urgent need for a textbook that presents Turbo C to students—introducing them to the superior technology of this environment in a logically organized, highly readable text. Now Larry Miller and Alex Quilici have met that need with *The Official Borland Turbo C Survival Guide.*

Here students will find a thorough and accurate discussion of Turbo C, beginning with a tutorial introduction to the language and its environment, and continuing on through advanced topics such as memory models, portability and efficiency, and graphics programming. Presentation of the material is enhanced by hundreds of example programs that illustrate Turbo C's features, and by a set of case studies implementing common real-world applications. Pictorial descriptions of data structures and algorithms also clarify and simplify features and functions of Turbo C.

Students at various levels of experience—whether self-taught or learning in the classroom—will find that *The Official Borland Turbo C Survival Guide* can open to them the substantial power of programming in Turbo C.

Phillipe Kahn
President and Founder
Borland International, Inc.

PREFACE

C is a powerful programming language, one that is especially popular among professionals who program small personal computers. And Turbo C—with its sophisticated programming environment, its speedy compiler, and its special language features—makes programming in C a pleasure. But unfortunately C also has a reputation for being hard to learn and even harder to master. Its conciseness and complexity can overwhelm all but the most experienced programmer. *The Official Borland Turbo C Survival Guide* will keep you from slowly sinking into a sea of confusion, and speed your journey from novice to expert C programmer. You'll find this book worthwhile if:

- *You've programmed in other high-level languages such as* BASIC, FORTRAN, *or* Pascal, *and now want to program in* Turbo C. We don't start from scratch and try to teach you how to program. Instead, we assume some prior programming experience and expend most of our effort describing how C differs from other high-level languages and investigating its unique and unfamiliar features.

- *You've programmed in C but you don't yet feel as though you've mastered it.* We cover the language completely, delving deeply into many issues that are often casually examined or completely avoided. You can't be a proficient C programmer without knowing how to put together large programs, how to produce portable and efficient code, or how to use pointers to effectively organize and access data—topics we emphasize rather than ignore.

- *You're familiar with C but not with Turbo C or* ANSI *C.* Turbo C implements the new ANSI C standard, along with some of its own additional features. We examine and exploit its ANSI C extensions such as function prototypes and implicit string concatenation. We explore the special Turbo C extensions, including its memory models, pointer modifiers, and graphics library. And we explain how to use the Turbo C programming environment, including the program builder and the debugger.

- *You're looking for examples of real-world C programs or are interested in obtaining a set of tools to improve your* DOS *environment.* We provide over 175 useful programs and functions, including a file display program, an indexed data base manager, a cross referencer, a function plotter, a file compression program, an octal dump program, a program to eliminate duplicate lines, various sorting and searching programs, and more. We also provide implementations of useful

libraries, such as sets and queues, and utility functions to manage console input and output. All of these programs have been written, compiled, and executed using Turbo C version 2.0.

Special Features

This book has grown out of several years of teaching C courses to a wide range of students—from first year computer science majors to lifelong assembly language programmers, from casual computer users familiar only with BASIC to competent C hackers fine-tuning their skills. And we've concluded that there are several ways to ease the often difficult process of mastering a programming language, all of which we've incorporated here:

- *We provide complete, useful programs.* Our examples don't merely illustrate language features but illustrate them in a realistic way, as part of a useful function or program. And our examples are complete—we avoid potentially confusing program fragments and ensure that every function comes with a main program that shows how to call it and how to use its return value.

- *We provide sizeable programs.* Almost every chapter ends with a case study, a large, real-world application that cements the concepts covered in the chapter. We often construct these case studies from functions and programs written and explained earlier in the text, reinforcing our earlier examples and providing realistic examples of how bigger programs are built from smaller pieces.

- *We provide pictorial descriptions of data structures and algorithms.* Numerous illustrations help clarify complex concepts such as pointers, arrays, and dynamic allocation. These pictures simplify seemingly complicated data structures and algorithms, and ensure that our explanations are easy to follow.

- *We provide plenty of programming exercises.* We've included over 400 exercises spanning a wide range of difficulty. Some are simply modifications to our example programs that make them more efficient, more concise, more useable, or more user-friendly. Others range from small programs to sizeable programming projects, many of which are useful programs in their own right.

Organization

We divide the text into five parts. Part I is a tutorial introduction to Turbo C. Part II is detailed discussion of its basic data types (including pointers and arrays), operators, and statements. Part III covers constructing more complex programs, looking at functions, storage classes, and separate compilation. Part IV looks at data structures, from arrays of pointers to linked lists and trees. Finally, Part V covers special Turbo C features such as its graphics library and its memory models, studies real-world concerns such as portability, efficiency, and debugging.

Part I—A Tutorial Introduction to Turbo C • *Chapter 1* presents our first C program—one that computes the interest accumulating in a bank account—and uses it

to introduce the basic C data types, operators, and statements, and to explain how to use the Turbo C development environment to compile and run programs. • *Chapter 2* provides three more example programs—an extension to the interest rate accumulator, a program to reverse its input, and an implementation of insertion sort—and uses them to introduce several additional control structures and operators, and the basics behind arrays and functions. This chapter also examines input/output (I/O) redirection and explains how it turns these programs into useful tools. Together, these two chapters provide an excellent feel for what programming in Turbo C is like.

Part II—Data Types, Operators, and Statements • *Chapter 3* describes data types, placing extra emphasis on how to manipulate characters, including discussions of character input and output and character testing. It concludes with our first case study, a program to display its standard input a page at a time. • *Chapter 4* addresses operators, paying special attention to the shorthand assignment and bitwise operators, and to automatic type conversion and casting. Its case study is a program to compress and uncompress files. • *Chapter 5* studies statements, showing the most appropriate uses for each, and concluding with a case study that dumps its input in octal. • *Chapter 6* presents pointers, arrays, and strings, emphasizing the relationships between them. It also introduces the standard string functions and uses them to write a useful tool to detect duplicate input lines.

Part III—Constructing Complex Programs • *Chapter 7* focuses on functions, exploring the details of how they can communicate with each other through parameter passing and return values. It also covers the closely related topics of pointers to functions, generic functions, and functions that can take variable numbers of arguments. Its case study is a recursive implementation of binary search. • *Chapter 8* studies storage classes and separate compilation, and discusses Turbo C's program builder. Its case study shows how to create abstract data types, illustrating them with a package that implements sets. • *Chapter 9* presents the preprocessor, describing how we can use it to make our programs easier to read and debug, as well as more efficient. The case study uses macros to create a simpler interface to functions with complicated arguments.

Part IV—Organizing and Accessing Data • *Chapter 10* discusses multidimensional arrays, showing how we can use pointers to access them efficiently. It dramatically illustrates their use in an implementation of the Game of Life. • *Chapter 11* presents arrays of pointers, showing how we can use dynamic allocation to initialize them, and how we can use them to access command-line arguments. Its case study is a string sorting program that takes advantage of both. • *Chapter 12* studies structures, unions, and enumerated types, including arrays of structures, using all of them in a simple data base program. • *Chapter 13* continues our study of structures by showing how to write insertion sort using linked lists and trees, and combining both in a C program cross-referencer. • *Chapter 14* explains external files, providing complete coverage of the standard I/O library, bringing together the ideas of the chapter in an indexed data base for names, addresses, and phone numbers.

Part V—Real-World Programming Issues • *Chapter 15* shows how Turbo C's memory models and pointer modifiers support writing programs that require large amounts of memory. Its case study describes how to write programs that work well under multiple memory models. • *Chapter 16* presents common portability problems

and suggests some solutions, concluding with a set of functions for portably managing console displays. • *Chapter 17* examines efficiency, including techniques for making our programs run faster or take up less space. Its case study provides fast functions for allocating fixed-size blocks of memory and for reading in an array of integers. • *Chapter 18* is an in-depth discussion of Turbo C graphics programming that ends with a program to plot functions. • *Chapter 19* describes the debugger and uses it to debug several short programs. The book concludes with a complete set of appendices that discusses configuring Turbo C, presents the different command-line options, menu entries, and header files, and summarizes the Turbo C library functions.

Acknowledgements

Contributions by several of our friends and colleagues have greatly improved the quality of this text. We're deeply in debt to Robert Quilici for painstakingly plowing through our prepublication drafts, unearthing plenty of problems with our programs and explanations. And we're grateful to Jake Richter for his careful proofreading, to David Smallberg for his always perceptive comments and criticisms, and to Dorab Patel for his invaluable wizardry with formatting.

We would like to thank the UCLA Computer Science Department and USC's Information Science Institute for the generous use of their resources and their allowing us the time to produce this work. We would also like to thank the people at John Wiley, especially our editors, Gene Davenport and Terry Zak, who as usual have been extremely helpful and made life pleasant for their authors.

And finally, we're grateful for all of the encouragement and support given to us by our families and friends—especially Rita Grant-Miller, Tammy Merriweather, Irene Borromeo, and Doris Perl.

Lawrence H. Miller
Alexander E. Quilici

Los Angeles, California
August 1988

CONTENTS