

Bruce
Douglass
and the Blacksburg Group

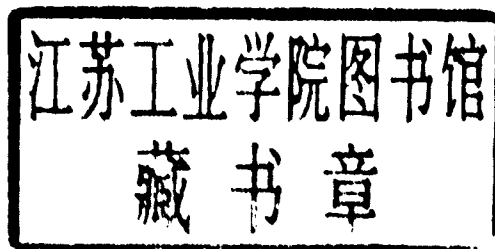
**USING
TURBO
AND
IBM
PASCAL**

An
Applications
Approach

Using Turbo and IBM Pascal

An Applications Approach

**Bruce Powell Douglass, PhD. and
the Blacksburg Group**



A Brady Book
Published by Prentice Hall Press
New York, NY 10023

Using Turbo and IBM Pascal: An Applications Approach

Copyright © 1986 by Brady Books, a division of
Simon & Schuster, Inc.

All rights reserved
including the right of reproduction
in whole or in part in any form

A Brady Book
Published by Prentice Hall Press
A Division of Simon & Schuster, Inc.
Gulf + Western Building
One Gulf + Western Plaza
New York, New York 10023

PRENTICE HALL PRESS is a trademark of Simon & Schuster, Inc.

Designed by Michael J. Rogers
Manufactured in the United States of America

2 3 4 5 6 7 8 9 10

Library of Congress Cataloging in Publication Data

Douglass, Bruce P.
Using Turbo and IBM Pascal

Includes index.

1. IBM Personal Computer—Programming 2. Turbo
Pascal (Computer program) 3. PASCAL (Computer program
language) I. Blacksburg Group II. Title.
QA76.8.I259D69 1985 005.2'65 85-28671
ISBN 0-89303-912-8

Special volume discounts are available by contacting the Special Sales
Department, Prentice Hall Press, Gulf + Western Building, One Gulf +
Western Plaza, New York, New York 10023.

ASCII table courtesy of International Business Machines, Inc.

Using Turbo and IBM Pascal

An Applications Approach

Limits of Liability and Disclaimer of Warranty

The author and publisher of this book have used their best efforts in preparing this book and the programs in it. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

Registered Trademarks

- IBM, IBM PC, IBM Pascal, IBM Personal Editor, and IBM Professional Editor are trademarks of International Business Machines.
- Turbo Pascal is a registered trademark of Borland International, Inc.
- WordStar is a registered trademark of MicroPro International Corp.

Preface

The purpose of this book is four-fold: 1. Teach the Pascal language on the IBM PC using both Turbo and IBM Pascal. 2. Teach Pascal so that users will be able to write applications and utility programs. 3. Provide an introduction to data structures and algorithms. 4. Provide useful programs for the readers of this book.

This book assumes only a small amount of expertise on the part of the reader. General familiarity with any programming language and your IBM PC will be enough to obtain a good understanding of the material presented.

There are a number of books about Pascal programming on the IBM PC. However, none of them deal with the broad range of subjects discussed here. To date, no other books deal with the most popular PC Pascal compiler, Turbo Pascal. IBM Pascal is also popular, and quite powerful, and is discussed in a number of other books. However, these other books do not show the undocumented graphics features already built into the IBM PC Pascal compiler or how to use this compiler's many powerful features. This book addresses both the Turbo Pascal and IBM Pascal users in enough depth to write serious applications programs in either dialect of Pascal.

In order to write real programs, a thorough knowledge of the methods of storing information (data structures) and methods of using this information (algorithms) is required. This book discusses and gives useful examples of a variety of data structures, including arrays, lists, and trees, and methods for using the information, including searching and sorting.

This book also provides useful Pascal programs. These examples are not only useful from an applications perspective—they illustrate the combined use of Pascal, data structures, and algorithms in the construction of programs. For example, linked lists are demonstrated with a program that manipulates polynomial expressions and binary trees are demonstrated with a program that provides a cross reference listing of other programs. By showing how real programs can be built from the elements of programming, learning is reinforced.

A diskette containing most of the programs in this book is available for \$24.95, including shipping and handling. To order, send a check or money order to:

Bruce Douglass
P.O. Box 181185
Fort Worth, TX 76118

Specify both the compiler (IBM, MS, or Turbo) and the DOS version (1, 2, or 3).

This book is dedicated to my mentor, David Hastings, for gallantly fighting off the forces of evil while I finished my PhD. May the force be with you.

Contents

- 1 Getting Started with Your Compiler / 1**
 - 1.1 Using the Compiler / 1
 - 1.2 Backing Up the Master Diskettes / 1
 - 1.3 A Tale of Two Pascal Compilers / 3
 - 1.4 Using Turbo Pascal Versions 2.0/3.0 / 3
 - Configuring the Screen Display / 4
 - Configuring the Turbo Editor Commands / 5
 - Writing and Compiling Programs with Turbo Pascal / 7
 - 1.5 Creating Work Diskettes (IBM Pascal Version 1.00) / 8
 - Installing IBM Pascal on 128K Machines with Single-Sided Drives / 9
 - A Sample Program Compilation with 128K and Single-Sided Drives / 9
 - Installing IBM Pascal on 128K Machines with Double-Sided Drives / 11
 - Installing IBM Pascal on PCs with Lots of Memory and Double-Sided Drives / 13
 - Installing IBM Pascal on PCs with Hard Disks / 14
 - 1.6 Creating Work Diskettes (IBM Pascal Version 2.0) / 14
 - 1.7 Running the Programs in This Book / 15
 - 1.8 Introduction to Pascal / 16
 - 1.9 Writing Your First Program / 17
- 2 Variables and Data Types / 21**
 - 2.1 The Elementary Data Types / 22
 - 2.2 Variables / 23
 - 2.3 The Assignment Statement / 25
 - 2.4 The VALUE Section (IBM Pascal Only) / 26
 - 2.5 Constants / 27
 - 2.6 Quiz / 30
- 3 Elementary Input/Output (I/O) / 31**
 - 3.1 Typing in the Data / 31
 - 3.2 Formatting Output with WRITE and WRITELN / 35
 - 3.3 Quiz / 38
- 4 Expressing Yourself With Pascal / 39**
 - 4.1 Operator Precedence / 41
 - 4.2 Arithmetic Logical Operators (Optional) / 43
 - 4.3 Arithmetic Functions / 45
 - 4.4 Bug Report! / 49
 - 4.5 Extended Intrinsic Functions / 50
 - 4.6 Using IBM's Extrinsic Functions / 51
 - 4.7 Boolean Expressions / 55
 - 4.8 Using Relational Operators in Boolean Expressions / 58
 - 4.9 Real Caveat / 59

- 4.10 Character Operations and Functions / 59
- 4.11 Quiz / 61
- 5 Simple and Compound Statements / 63**
 - 5.1 Conditional Statements: IF / 63
 - 5.2 Compound Statements / 64
 - 5.3 Conditional Statements: CASE / 67
 - 5.4 Quiz / 70
- 6 Loops and Jumps / 73**
 - 6.1 The FOR Loop / 73
 - 6.2 WHILE Loops / 78
 - 6.3 REPEAT Loops / 80
 - 6.4 Unconditional Branching (Jumps) / 81
 - 6.5 Quiz / 83
- 7 Procedures and Functions / 85**
 - 7.1 Declaring Procedures and Functions / 86
 - 7.2 Parameter Passing / 92
 - 7.3 Scope / 94
 - 7.4 Reference Parameters / 99
 - 7.5 Other Parameter Types (IBM Pascal) / 102
 - 7.6 Recursion / 105
 - 7.7 Procedure and Function Attributes / 111
 - EXTERN (IBM Pascal Only) / 111
 - FORWARD / 112
 - 7.8 Quiz / 113
- 8 Advanced Types / 115**
 - 8.1 User-Defined Data Types / 115
 - 8.2 Subrange Types / 119
 - 8.3 Sets and the Single IBM / 121
 - 8.4 SET Technical Note (Optional) / 127
 - 8.5 Quiz / 130
- 9 Arrays and Strings / 133**
 - 9.1 PACKed Arrays / 142
 - 9.2 A Note About Array Parameters / 144
 - 9.3 Super Arrays (IBM Pascal) / 145
 - 9.4 Strings (IBM Pascal) / 146
 - 9.5 String Functions (IBM Pascal) / 149
 - 9.6 Strings (Turbo Pascal) / 154
 - 9.7 String Functions (Turbo Pascal) / 157
 - 9.8 Quiz / 161
- 10 Records / 163**
 - 10.1 Declaring Record Types / 163
 - 10.2 Using Records / 164
 - 10.3 Use of Structured Reference Parameters / 172

-
- 10.4 Variant Records / 176
 - 10.5 Structured Constants / 181
 - 10.6 Quiz / 191
 - 11 Pointers and Dynamic Variables / 193**
 - 11.1 Declaring Pointer Types / 193
 - 11.2 Quiz / 198
 - 12 Pascal Files and Advanced I/O / 201**
 - 12.1 Opening Files / 201
 - 12.2 Special IBM Files / 212
 - 12.3 File Buffer Variables (IBM Pascal Only) / 219
 - 12.4 Reading Special Keys / 224
 - 12.5 Error Trapping (IBM Pascal Only) / 227
 - 12.6 Error Trapping (Turbo Pascal Only) / 229
 - 12.7 Random Access (Direct) Files (IBM Pascal Only) / 231
 - 12.8 Random Access (Direct) Files (Turbo Pascal Only) / 237
 - 12.9 Turbo Pascal Version 3.0's Special I/O Features / 241
 - 12.10 Quiz / 244
 - 13 Data Structures / 247**
 - 13.1 Queues / 247
 - 13.2 Stacks / 251
 - 13.3 Recursion / 265
 - 13.4 Linked Lists / 274
 - 13.5 Binary Trees / 299
 - 13.6 Quiz / 324
 - 13.7 Programming Problems / 324
 - 14 Access to Hardware with IBM Pascal / 327**
 - 14.1 Date and Time / 327
 - 14.2 DOSXQQ Function / 329
 - 14.3 Graphics with IBM Pascal / 338
 - 14.4 Video Graphics Procedures and Functions / 348
 - 15 Access to Hardware with Turbo Pascal / 359**
 - 15.1 Interrupts / 359
 - 15.2 DOS Functions / 360
 - 15.3 Graphics / 362
 - 15.4 Windows / 366
 - 15.5 Turbo Pascal Version 3.0 Advanced Graphics / 372
 - 16 Advanced Compiler Features / 377**
 - 16.1 Compiler Metacommands (IBM Pascal) / 377
 - 16.2 Compiler Metacommands (Turbo Pascal) / 388
 - 16.3 Modular Programming with IBM Pascal / 391
 - 16.4 Using the Library Manager / 398
 - 16.5 Semi-Modular Programming with Turbo Pascal / 399

Appendix ASCII Character Set / 431

Quiz Answers / 435

Index / 441

Getting Started with Your Compiler

1.1 Using the Compilers

Before you begin learning the Pascal language, you must learn how to operate the compiler. Then, when you read a program in the book, you can enter it and run the program. This makes learning the language much easier. So, read the following introduction to compiling programs before progressing to the subsequent chapters.

A *compiler* is a program that translates one language into another. The Pascal compiler translates Pascal *source code* (what you write) into machine *object code* (what the computer can do). How people think about problems is far removed from how machines can solve them. It would be tedious indeed to force people to think about problems in the same way that computers do. The whole idea of a compiler is to allow people to think about and solve their problems in a language they understand and at the same time allow the machine to perform its duty in a language it understands.

1.2 Backing Up the Master Diskettes

Using the compilers is straightforward. The first thing you must do with your original diskettes is to back them up; that is, make copies of them. Diskettes are relatively fragile things, so *never* use your master copies except to make backup copies. Work from your backup or working copies. If anything should go wrong with the working copy, you can always make another backup from your


safely-stored master. Making backup copies is easy. We'll assume you have a two-drive IBM PC. Turn on your computer and put your Disk Operating System (DOS) diskette in the first disk drive (drive A); after a few moments, the screen displays

A>

If you are using Turbo Pascal, get one blank diskette; if you are using IBM Pascal, then get three blank diskettes. These will become your backup copies. Make sure that nothing you want is on these diskettes, because we are going to make copies of the master diskettes on these diskettes. Any previous information on the diskettes will be erased.

Now, we will run the DISKCOPY program, as described in the IBM DOS Reference Manual. At the A> prompt, type in

DISKCOPY A: B:<return>

where < return> means "press the <return> key." This is the key labeled  just to the right of the "{ }" key. The PC then displays:

**Insert source diskette in drive A:
Insert target diskette in drive B:
Strike any key when ready**

Remove the DOS diskette from drive A: and put in the first master diskette in drive A:. Put a blank diskette in drive B: as well. The DISKCOPY program will copy the diskette in drive A: to drive B:. When this is completed, you will see the message:

Copy another (Y/N)?

If you are using Turbo Pascal, you are done. IBM Pascal has two more diskettes to copy; in this case, make sure you label each diskette as you copy it. Put the second master diskette in drive A: and a second blank diskette in drive B:. Now, you can make a copy of this diskette as before. Similarly, copy the third master diskette onto the third blank diskette.

Now, put your master copies in a safe place away from food, drinks, magnets, speakers, headphones, heaters, etc. Label the backup copies with diskette labels with the same name that appears on the original master diskettes: PAS1, PAS2, and PASCAL.LIB. From here on, whenever we refer to PAS1, PAS2, PASCAL.LIB, or the Turbo Pascal diskettes, understand that we refer to backup copies, *not* the master copies. Normally, a "scratch" diskette is put in drive B: when writing and compiling programs. This is a formatted diskette that holds the source program written with the editor and the object program written by the compiler. It may have other programs and files on it, provided that enough space is left for the editing and compilation programs. That is generally not a problem with Turbo Pascal, since the editor and the compiler only require about 33K of diskette memory together, but it is important for the IBM Pascal compiler.

1.3 A Tale of Two Pascal Compilers

This book is specifically designed to help you learn how to write useful Pascal programs for the IBM PC. In this book, we discuss two particular Pascal compilers. The first is offered by IBM, called IBM Pascal. It is expensive and slow for the generation of programs, but the programs produced by the compiler are quite good. This Pascal compiler provides some powerful facilities not needed by everyone, however. The other compiler is Turbo Pascal from Borland International. It is inexpensive and generates high quality programs very quickly and easily. Which compiler to use is an individual choice. Both may be used to generate professional quality programs. Both are complete and powerful implementations of the Pascal language with good enhancements to make programming easier.

IBM Pascal offers a few more features, such as separate compilation (a process by which large programs are broken up into different pieces and compiled separately) and the use of a good *linker* program to put these separately compiled pieces together into a finished product. Furthermore, IBM Pascal version 2.00 comes with a library manager, so that routines which are used often can be separately compiled and then placed in a library. Once there, they need not be recompiled again to be used in many different programs. The linker will load them as necessary.

Using the IBM compiler is fairly slow, however. The source program must be written with an editor and saved to disk. The compiler is then run. If there are no errors, then the linker is run to link together all the pieces of the program. Finally, the program can be run.

Turbo Pascal, on the other hand, is designed to be quick and easy to use. It currently does not have the ability to do separate compilation, but the compiler and editor are coresident (meaning both are in memory at the same time). If the compiler detects an error during the compilation process, it invokes the editor and positions the cursor at the position in the source code at which the error is detected. This is a great time-saver, since the lengthy process of edit-compile-link-run can be done with Turbo Pascal in a single step.

In the final analysis, the IBM Pascal compiler is somewhat more powerful than Turbo Pascal, but most people won't notice the difference. The difference they *will* notice is that they can get programs working much faster with Turbo Pascal than IBM Pascal.

1.4 Using Turbo Pascal Versions 2.0/3.0

Since Turbo Pascal is easier to use, we'll discuss how to use it first. With the IBM Pascal compiler, you must keep several programs on the compiler diskette. With Turbo Pascal, you need only two, and the second is optional. The first is called `TURBO.COM`; the second is a file of error messages (highly recommended) called `TURBO.MSG`. Turbo Pascal comes ready to run, but it

may require some customization before it is ready for your particular system. If you have a color graphics card installed in your IBM, but use a black and white monitor (one popular configuration), then you must install Turbo's screen before you can read any text it puts on the screen.

If you are using Turbo Pascal version 3.0 or later, you must create a file called CONFIG.SYS that contains the statement FILES=20. This tells PC-DOS to allocate space for 20 file buffers when DOS initially starts up. This is not required with the earlier versions of Turbo Pascal.

Configuring the Screen Display

Installing the screen parameters is quite simple. The Turbo Pascal master diskette contains a program called TINST.COM, and a file of its messages called TINST.MSG. Copy these from the master diskette to your work diskette along with TURBO.COM, TURBO.MSG, and TLIST.COM (if you want to print copies of your programs on your printer). TINST.COM is only needed for the installation procedure and can be deleted from your work diskette once Turbo Pascal is properly installed.

Run TINST.COM by typing

TINST <return>

You will see the display:

**Turbo Pascal Installation Menu
Choose installation item from the following:**

[S]creen installation | [C]ommand installation | [Q]uit

Enter S, C, or Q:

If you have a color graphics card and a monochrome monitor, then you probably won't be able to read this screen, but that's what it says. If this is the case, then follow the steps below exactly as shown:

1. Press the <S> key. The screen displays:

Choose One of the following Displays:

- 0) Default display mode**
- 1) Monochrome display**
- 2) Color display 80x25**
- 3) Color display 40x25**
- 4) b/w display 80x25**
- 5) b/w display 40x25**

Which display? (Enter no. or ^Q to exit): _

2. Press the <4> key. This chooses the 80x25 black-and-white display using a color graphics card.
3. Press the <Q> key to quit the installation program.
4. Run Turbo Pascal by entering:

TURBO <return>

You should now be able to read the screen.

An interesting feature of the Turbo system is that once the screen is installed, you must run **TURBO.COM** before running other programs in the **TURBO** package (such as **TLIST.COM** or **TINST.COM**) in order to get the screen to the set configuration. For example, if you reset the IBM PC by pressing <CTRL> <ALT> and run **TINST**, the screen reverts back to the unconfigured state. If you run **TURBO** first, exit **TURBO** by pressing the <Q> key, and then run **TINST**, the screen is properly configured again.

Configuring the Turbo Editor Commands

The Turbo Editor is initially set up to use the same command sequences as the WordStar word processing program. If you wish to your own commands, use **TINST** to redefine the commands. To do this, choose <C> from the **TINST** main menu. You are presented with each of the commands in turn, and you must enter the keys you want to press to invoke the command.

Below is a sample installation for Turbo defining it the way the author prefers it. It is mnemonic, easy to remember, and commands do not require many keystrokes. The keys on the numeric key pad are shown as <right>, <left>, <up>, <down>, <home>, <end>, <ins>, , <pgUp>, and <pgDown>. They are recorded by Turbo as two key sequences, but don't worry. Turbo records them correctly.

Terminate a command definition by pressing the <enter> key. If you err on a specific command definition, keep on entering the definitions and re-enter the [C]ommand installation procedure and correct your mistake. To keep a command definition the same, press the <enter> key without entering a new definition.

The control key is indicated by Ctrl-<key>, such as Ctrl-A or Ctrl-Z. This means "hold down the control key and press the second key." Other keys are pressed one at a time. Finally, commands that begin with <ESC> or Ctrl-<key> must be the same length. For example if one command is Ctrl-<K>Ctrl-<D>, you cannot have another command be just Ctrl-<K>, or Ctrl-<K> Ctrl-<S> <D>. However, you can have Ctrl-<P> be a command by itself.

Table 1-1 Sample key definitions for the Turbo Editor.**Cursor Movements:**

1: Character left	<left>
2: Alternate	nothing
3: Character right	<right>
4: Word left	Ctrl-L
5: Word right	Ctrl-W
6: Line up	<up>
7: Line down	<down>
8: Scroll down	<ESC> -
9: Scroll up	<ESC> +
10: Page up	<pgUp>
11: Page down	<pgDown>
12: To left on line	<end>
13: To right on line	<home>
14: To top of page	<ESC> w
15: To bottom of page	<ESC> b
16: To top of file	<ESC> t
17: To end of file	<ESC> e
18: To beginning of block	Ctrl-J Ctrl-B
19: To end of block	Ctrl-J Ctrl-E
20: To last cursor position	<ESC> =

Insert and Delete

21: Insert mode on/off	<ins>
22: Insert line	Ctrl-N
23: Delete line	Ctrl-D Ctrl-L
24: Delete to end of line	Ctrl-D Ctrl-E
25: Delete right word	Ctrl-D Ctrl-W
26: Delete character under cursor	
27: Delete left character	<back space>
28: Alternative	Nothing

Block Commands

29: Mark block begin	Ctrl-B Ctrl-B
30: Mark block end	Ctrl-B Ctrl-E
31: Mark single word	Ctrl-B Ctrl-W
32: Hide/Display block	Ctrl-B Ctrl-H
33: Copy block	Ctrl-B Ctrl-C
34: Move block	Ctrl-B Ctrl-S
35: Delete block	Ctrl-B Ctrl-D
36: Read block from disk	Ctrl-B Ctrl-R
37: Write block to disk	Ctrl-B Ctrl-P
