Using QuickBASIC 4.5,

Second Edition

Don Inman

Bob Albrecht

Using QuickBASIC 4.5,

Second Edition

Don Inman

Bob Albrecht

Osborne **McGraw-Hill**

Berkeley New York St. Louis San Francisco
Auckland Bógota Hamburg London Madrid
Mexico City Milan Montreal New Delhi Panama City
Paris São Paulo Singapore Sydney
Tokyo Toronto

Osborne **McGraw-Hill**
2600 Tenth Street
Berkeley, California 94710
U.S.A.

For information on translations and book distributors outside of the U.S.A., please write to Osborne **McGraw-Hill** at the above address.

A complete list of trademarks appears on page 419.
Screens produced with InSet, from Inset Systems, Inc.

Using QuickBASIC 4.5, Second Edition

234567890 DOC 89

ISBN 0-07-881514-2

Preface

BASIC has long been the language most used for personal programming. Most personal computer users have learned to program in BASIC because it is easy to learn and use. Over the years, many BASIC dialects have emerged from the first simple version.

QuickBASIC from Microsoft takes a giant leap beyond yesterday's BASIC, which was designed for beginners, to a truly structured and powerful, professional language. Since it has its roots in Microsoft BASIC, it is still easy to learn and use. Most of your own BASIC programs can be run from QuickBASIC. However, you will soon be rewriting old BASIC programs to take advantage of QuickBASIC's advanced features.

Using QuickBASIC 4.5, you can edit, compile, run, and debug programs within a single environment. All of QuickBASIC's programming tools are contained in one integrated package, ready for use at any time.

About This Book

This book assumes that you have previously programmed in some version of BASIC. If you have used some form of Microsoft BASIC,

that will be even more helpful. An elementary knowledge of the Microsoft Disk Operating System (referred to in this book as MS-DOS) is also assumed.

The disks containing the language and supporting files provide a powerful programming package. Included on the disks are three files that provide instant on-line documentation. These files provide help in crucial situations. One provides summaries and syntax descriptions of QuickBASIC statements and functions; another provides information on using QuickBASIC statements and functions (including code examples that you can copy and run); and the third gives explanations of QuickBASIC menus, commands, dialog boxes, and error messages. In addition, the QuickBASIC manuals *Learning to Use Microsoft QuickBASIC* and *Programming in BASIC* cover many aspects of the language.

This book offers a different point of view and a different approach to learning how to use the massive amount of information available from Microsoft's manuals and disks. A background is laid in the early chapters to provide readers with a common base from which to start. From this point a smooth transition to the rich language elements of QuickBASIC can be made.

The book contains many demonstration examples and programs to simplify the explanation of QuickBASIC procedures, statements, and functions in a meaningful context. Screen displays are used to show what is happening at critical stages in the development of the demonstrations.

As you work through *Using QuickBASIC 4.5*, you'll find it easier to learn and understand the discussions if you actually sit down at your computer and run the examples and demonstration programs. At times we describe specific steps in a given computer operation, but it is not practical to describe the steps for every conceivable computer configuration. Therefore, we describe steps for a system with two disk drives (one 3 1/2 inch and one 5 1/4 inch). It is much easier to transfer such information from a small to a large system than vice versa.

A consistent style of programming has been used in this book. Programs are easy to read and understand. The book avoids programming "tricks" and shortcuts that would detract from learning.

After explaining how to configure a QuickBASIC work disk, we describe fundamental tools of QuickBASIC. Their use is demonstrated with practical examples. You learn how to move around in QuickBASIC's menu system and how to use the menus for loading, saving, and running programs.

After learning the basics, you are ready to move on to some of QuickBASIC's more powerful control structures. Editing tools are used to delete, insert, and copy characters, words, and blocks of text and to merge complete files. Some editing tools are immediately available from the QuickBASIC editor; the more powerful ones are accessed from the Edit menu.

The features of QuickBASIC procedures, subprograms (SUB...END SUB), and function definitions (FUNCTION... END FUNCTION) are discussed and fully demonstrated. They are powerful structures that will enhance your programs. The View menu, used to access procedures after they have been created, is explained and demonstrated in many examples.

Data files, both sequential and random access, are discussed and used to show how many QuickBASIC features can be used in applications. Methods of scanning, copying, inserting, adding, and editing records within data files are described and many examples shown.

The Debug menu is explained and used to show QuickBASIC tools for debugging programs. Single-stepping through programs is demonstrated, along with the use of watchpoints, instant watch, breakpoints, and program tracing and history.

A chapter on creating and running executable files is included. Executable files have an .EXE extension and will run directly from your disk operating system (MS-DOS).

The book concludes with a chapter that discusses the creation and use of Quick libraries, which can be included with any QuickBASIC program.

Microsoft QuickBASIC is an extremely powerful language that contains rich and elegant features. This book "walks" you through the fundamental features so that you will have the background necessary to further explore QuickBASIC in an intelligent way.

Programming skills are developed by studying programs and experimenting with the elements of a language. You are encouraged to use this book as a beginning for such exploration and development.

– Don Inman
Bob Albrecht

Using QuickBASIC Convenience Disk

You may order a disk containing programs, defined functions, and procedures discussed in *Using QuickBASIC 4.5*. This disk includes

- All BASIC and QuickBASIC programs shown in this book
- Single-line and multi-line functions stored as separate files so you can easily merge them with your programs
- SUB and FUNCTION procedures stored as separate files so you can easily merge them with your programs
- A bonus! The disk includes several files not described in the book

The disk is available in two sizes: 5 1/4-inch disk $6.00
3 1/2-inch disk $7.00

Specify disk size and *Using QuickBASIC 4.5* when ordering.

Workbooks

To further assist you in the enjoyable task of learning QuickBASIC, the authors are developing an ongoing series of workbooks for self-instruction in the use of graphics and sound.

- *QuickBASIC Graphics and Sound, Book 1* is a beginner's introduction to QuickBASIC's extensive repertoire of graphics and sound features. 64 pages, $4.95.
- *Datafile Programming in QuickBASIC, Book 1* supplements the material on data files in *QuickBASIC Made Easy* and in *Using QuickBASIC 4.5*. 64 pages, $4.95.
- *QuickBASIC for Math and Science, Book 1* explores diverse applications in mathematics and science. 64 pages, $4.95.

Use the order form that follows.

Contents

1 A Review of BASIC

BASIC (Beginner's All-purpose Symbolic Instruction Code) was available as a programming language for large computers long before the first microcomputer appeared. BASIC arose from a quest to make computing power available to all people, not just to those professional programmers who normally had access to large, expensive computers. Dartmouth professors John G. Kemeny and Thomas E. Kurtz developed the original BASIC language in 1963 and 1964 as an instructional tool for training novice programmers. Their purpose was to design a language that would be easy to learn but still useful for any programming task. The success of BASIC and its widespread use are due to its simplicity, ease of use, and general-purpose power.

In 1965, BASIC became available outside of Dartmouth, initially by means of time-sharing systems. Its use later spread to the low-cost (at that time) dedicated minicomputers. When microcomputers were introduced to the public as personal computers, the computer moved from the exclusive realm of the professional programmer into the domain of the creative amateur. The only

high-level language available for these early machines was BASIC. A short history of BASIC is shown in Figure 1-1.

1956	Darsimco (Dartmouth Simplified Code), a simplified version of assembly language programming, was designed but did not catch on.
1956–57	Kemeny and Kurtz became increasingly aware of the shortcomings of batch-processing computer programs.
1957	FORTRAN appeared for experts.
Early 1960s	Kemeny and Kurtz began to seriously consider time-sharing for computer use by many users. The National Science Foundation approved funding for a project staffed by Kemeny, Kurtz, and a dozen undergraduates to develop a time-sharing system at Dartmouth College. This marked the beginning of personalized, interactive computing. Kemeny, Kurtz, and students worked with compilers, FORTRAN, ALGOL, and experimental languages, all of which influenced later development of BASIC.
1963	Dartmouth decided to enable all students to become computer literate. Kemeny and Kurtz, with help from students, began the development of a general-purpose, time-sharing computer system and a new language (BASIC) to introduce beginners to programming and serve all applications for large and small systems. The original features of BASIC were designed to • Be general-purpose in nature for writing any type of program • Allow for advanced features if needed later • Provide for user/computer interaction • Provide clear and friendly error messages • Give a fast response for small programs • Require no hardware knowledge • Shield the user from the computer's operating system

Figure 1-1. *Early BASIC history*

1964

Equipment on which the time-sharing system and BASIC were to be developed arrived in February 1964. The equipment was fully functional by March 1, 1964.

On May 1, 1964, at 4 AM., John Kemeny and a student programmer entered and ran separate BASIC programs on the new system with success. Time sharing and BASIC were born.

1964-1971

The growth of BASIC in this period predated personal computers.

BASIC at Dartmouth made the transition from a language suitable only for small programs to a language suitable for building large application programs.

Kemeny and Kurtz had sole responsibility for Dartmouth's first BASIC; then others became involved.

Two main genealogical lines grew out of the original Dartmouth BASIC. Large-machine BASIC versions were probably direct descendants of GE BASIC, which in turn descended from Dartmouth BASIC. Most small-machine BASIC versions descended from versions that first appeared on the Hewlett-Packard HP-2000 or the Digital Equipment Corporation PDP-8. GE BASIC and Dartmouth BASIC were almost the same until around 1970; Dartmouth had built its time-sharing systems around GE hardware.

Expanded PRINT statements and the introduction of the INPUT statement (almost two years after BASIC first appeared) greatly enhanced the interaction between user and computer.

The use of files for purposes other than saving programs was introduced.

Provision for "calling" external subroutines added flexibility to large programs.

By the end of 1971, Dartmouth BASIC had reached its sixth version and was a huge success.

The material for this history was based on *Back to BASIC*, by J.G. Kemeny and T.E. Kurtz (Addison-Wesley, 1985), which is recommended reading.

Figure 1-1. *Early BASIC history* (continued)

Because early microcomputers had limited memory, early versions of BASIC were crunched adaptations of Dartmouth BASIC. Today, there is at least one version of BASIC available for each brand of microcomputer. Today's versions are more complete and powerful, but they are still easy to learn and use. BASIC is now built into, or bundled with, almost every computer that goes into businesses, homes, and schools.

BASIC provides immediate interaction with the user by means of commands that cause some action on the video display; therefore, a beginner receives immediate positive reinforcement. After learning a few BASIC statements, a user can create programs in short, simple steps. The desire to solve interesting problems or create interesting video displays stimulates the desire—as well as the necessity—to learn more of BASIC's capabilities. This immediate and constant interaction between person and computer puts BASIC at the forefront of people-oriented computer languages.

Many languages are now available for microcomputers—COBOL, FORTRAN, LOGO, Pascal, Modula-2, and C, to name a few. The list of candidates in the battle for computer language supremacy goes on and on. Each language has its own group of proponents who expound the virtues of their choice.

BASIC is not a standardized language. Different versions have been created to fit unique hardware characteristics of individual computers. The programming styles of users also vary widely; people often write programs that work but are impossible for anyone other than the programmer to read. Even the programmer may have trouble reading the program later.

Proponents of other computer languages attack BASIC with the claim that it does not lend itself to well-structured programs. With a little planning and effort on the part of the programmer, however, BASIC programs can be structured into functional blocks of program lines. In addition, new BASIC interpreters and compilers, such as QuickBASIC, qualify as fully structured languages with multiline functions, procedures with both local and global variables, and structures such as IF. . .THEN. . .ELSE. . .END IF and DO. . .LOOP. Some versions of BASIC, including QuickBASIC, now have all the "goodies" that proponents of other languages have bragged about in the past.

BASIC has also been criticized because of its lack of speed. This was true in the past, since most versions of BASIC were written as

interpreters The microprocessor of a computer cannot directly execute BASIC statements; it can execute only its own binary machine language instructions. Therefore, before BASIC statements can be executed, they must be translated into the machine language of the processor.

There are two ways to provide this translation: by interpreter and by compiler. Nearly all versions of BASIC developed in the past and most current versions are interpreted. Each BASIC statement is translated (interpreted) as it is executed This line-by-line interpretation takes place every time the BASIC program is run, slowing the execution of the program.

The original version of BASIC and its further development at Dartmouth used a compiler to directly convert BASIC programs into machine language. A compiler translates (compiles) the BASIC statements before the program is run. The program is first compiled to produce an executable machine language file. When this file is run, it runs very quickly, since it is in the machine's native language The translation has already taken place, so no time is lost during the program's execution. Most commercial software operates in this way.

There are advantages and disadvantages to both methods of translation. Compiled programs often take more time to eliminate errors. Each time an error is discovered in the executable machine language file, the original BASIC program (called the *source code*) must be corrected. The program must then be recompiled before its executable file (called the *object code*) can be run again. If more errors are discovered, the original program must be corrected again and recompiled. Although the compiled code runs very swiftly, writing an error-free source program can be time-consuming. Interpreted BASIC, on the other hand, is translated line by line, and errors are quickly discovered and can be quickly corrected. The execution time is much slower than that of a compiled program, however.

QuickBASIC: The Best of Two Worlds

QuickBASIC 4.5 is the latest version of QuickBASIC from Microsoft Corporation. It runs programs much more quickly than