The Art of Business Programming

Shimon Schocken

with dBASE III PLUS & IV

The Art of Business Programming

with dBASE III PLUS and IV

Shimon Schocken
Information Systems Department
Leonard N. Stern School of Business
New York University

Mitchell McGRAW-HILL
New York St. Louis San Francisco Auckland Bogotá Caracas Hamburg
Lisbon London Madrid Mexico Milan Montreal New Delhi Paris
San Juan São Paulo Singapore Sydney Tokyo Toronto Watsonville

To My Mother
Devora Schocken

This book is printed on recycled, acid-free paper containing a minimum of 50% recycled de-inked fiber.

Sponsoring editor: Stephen Mitchell
Editorial assistant: Denise Nickeson
Director of production: Jane Somers
Production assistant: Leslie Austin
Project management: Business Media Resources
Cover design: Jim Love
Text design: Business Media Resources
Printer and binder: R.R. Donnelley and Sons

Mitchell **McGRAW HILL**, Watsonville, CA 95076

The Art of Business Programming with dBASE III PLUS and IV

2 3 4 5 6 7 8 9 0 DOC DOC 9 0 9 8 7 6 5 4 3 2

ORDER INFORMATION
ISBN 0-07-056919-3

Library of Congress Card Catalog No. 91-067525

Acknowledgments

First and foremost, I wish to thank the thousands of students and the two dozen instructors who patiently used rough drafts of the book and the software during the last three years and aided me in correcting the numerous bugs and glitches in the materials. In particular, the following NYU professors were very helpful in detecting problems and proposing solutions: Gad Ariav, Vasant Dhar, Barry Floyd, Thomas Isakowitz, Rob Kauffman, Alex Tuzhilin, and Rob Weitz. I also wish to thank my first computer science professor, Efraim Glinart of the Hebrew University of Jerusalem, whose introductory programming course took me a very long way.

Exercises 2-28, 2-29, and 2-30 were written by Professor Tuzhilin. The utility program **key-fake.com** that is included on the book's data disk was written by Charles Petzold and published by *PC Magazine*. The **drivea** program was written by Alex Bykov. If you have any comments or suggestions regarding the text or the software, please send a message to **sschocken@stern.nyu.edu**.

I hope the book and the software will serve you well, and that your students will find it interesting and enjoyable. For more information about suggested course outlines and useful teaching aids, please refer to the Instructor's Manual.

Shimon Schocken

God is in the details.

GOETHE (1749-1832)

Contents

Acknowledgments v
Introduction: To the Instructor 1
Introduction: To the Reader 9
How to Install the Data Disk on Your Computer . 13

Chapter One *Getting Started* 21
First Things First 22
The Three Faces of dBASE 22
A Glimpse of Interactive Mode 23
Interactive Number Crunching 25
Interactive File Management 26
A Bit of Programming 28
Program-Based Number Crunching 29
Program-Based Data Management 32
A Taste of ASSIST 34
The dBASE Industry 38
Additional Exercises 39

Chapter Two *Building Blocks* 43
Chapter Preview 43
Data Types 45
Constants 45
Variables 46
Numeric Variables 47
String Variables 48

Date Variables 48
System Variables 49
The Assignment Command 50
Input and Output 51
Input Commands 52
Output Commands 53
An Illustrative Input/Output Program 53
Numeric Processing 55
Numeric Operators 56
Numeric Functions 57
Numeric Expressions 57
An Illustrative Numeric Processing Program 58
String Processing 59
String Operators 60
String Functions 61
String Expressions 63
An Illustrative String Processing Program 63
Date Processing 65
Date Functions 66
Testing Programs that Use Date and Time Variables . . 67
The Role of Logic 67
Logical Relations 69
Logical Operators 70
Logical Expressions 71
An Illustrative Logic-Based Program 73
Constructing Fancy Outputs 74
Variable Naming Traps 76
Long Variable Names 76
Reserved Words 78
Additional Exercises 78

Chapter Three *Control Structures* 83
Conditional Branching (IF) 83
A Brief Diversion on Variable Declaration 85
The IF Syntax 86
The Subtle Difference Between = and = ... 89
Conditional Multi-Branching (CASE) 89
The CASE Syntax 92
CASE and IF 92
Looping (WHILE) 93
The WHILE Syntax 96
Calculating with WHILE 97
Analyzing Data with WHILE 98

Data Entry with WHILE 99
Text Processing with WHILE 101
Numeric Analysis with WHILE 103
The RETURN Command 105
Programs that Call Other Programs 107
The dBASE Interpreter as a Structured Program . 113
Variable Management 113
Variables Declaration 114
Variables Terminology 114
The Rules of the Game 115
How to Use Variables in dBASE 116
Additional Exercises 117

Chapter Four *Building a System* 123
The ATM System 124
The System's Architecture 125
The User Interface 126
The Accounts File 133
Putting It All Together 134
Additional Exercises 136
Postscript 137

Appendix *A Program Development Guide* 139
Design First, Program Later 139
Getting Help 140
Writing a Program 141
The dBASE Editor 143
Displaying a Program 145
Running a Program 145
Testing a Program 145
Debugging a Program 148
Debugging in dBASE III PLUS 149
Debugging in dBASE IV 150
Halting a Program 151
Printing a Program 151
Recording an Interactive Session 151
Putting It All Together 153

Index 155
About the Author

Introduction To the Instructor

Who Is This Book For?

This book can be used to support two types of courses: (1) required introductory courses for non-IS/CIS majors and (2) elective database management or project-oriented courses for IS/CIS majors. The book assumes the students have no previous experience with computers and no programming knowledge. Preview versions of the book have been used successfully in undergraduate, graduate, and executive-education programs in universities both in the USA and in Europe.

What Does the Book Cover?

The goal of the book is to expose the students to key *structured* programming techniques, using dBASE as a convenient, but not necessarily ideal, programming language. To that end, the book gives a *generic* coverage of such topics as data types, variables, assignments, string processing, numeric processing, input and output commands, logic, conditional branching, looping, and nesting. This material is presented in the context of day-to-day business problems in a way that resembles its use in languages like Pascal or C.

As a matter of principle, the book avoids the subjects of file processing and full-screen input/output (I/O). This is done for several reasons. First, there are numerous books that cover these aspects of dBASE, and there is no need to repeat this material here. Second, most instructors have concrete ideas on how file management should be taught in the classroom, and these ideas are probably as good as mine. Third, the full-screen user interface of dBASE is unique compared to other programming systems, and, given the book's focus on *fundamental* programming, it entails a rather confusing diversion. Furthermore, this user interface is likely to undergo dramatic changes as dBASE migrates into environments like Windows and OS/2.

This is not to say that the instructor cannot introduce the subjects of file management and full-screen I/O into the book. On the contrary, all the programs that appear in the text are highly modular, and it might be an excellent exercise to replace their line-oriented I/O commands with full-screen **SAY/GET/READ** templates.

Toward the end of the book, the students build a reduced version of a real system in which *variables* play the roles of fields and records. The system is built with all the necessary hooks to file management, and the next natural exercise would be to fill in these gaps with commands like **SEEK**, **MODIFY**, **LIST**, and so on. With the exception of this particular exercise, the book can be taught either before or after the students have been exposed to interactive file processing or to database management. Thus, the book provides a flexible module on structured programming that can be easily plugged into a variety of different IS/CIS curricula. The length of this module can be anything from four lectures to an entire semester, depending on the course objectives and time constraints.

What Is the Book's Style?

Different instructors use different methods to teach introductory IS/CIS material. The anecdotal approach consists of telling stories about how people design and use information systems in the real world. The theoretical approach concerns the methodology and theory of systems design. The practical approach attempts to bring the students to a level of competence that enables them to design reduced versions of real systems. This book offers a mix of these three delivery methods, with special emphasis on promoting analytical skills and technical literacy. The goal of the book is not to turn the students into programmers or systems analysts. Instead, the goal is to make them intelligent consumers of the types of systems they will have to order, budget, and use in their future workplaces.

The style of the book is somewhat different from that of a typical textbook. In writing it, I was influenced by the following advice of a master teacher: "Explaining is a difficult art. You can explain something so that your reader understands the words; and you can explain something so that the reader feels it in the marrow of his bones. To do the latter, it sometimes isn't enough to lay the evidence before the reader in a dispassionate way. You have to become an advocate, and use the tricks of the advocate's trade."[1]

Why Teach Programming?

Most introductory IS/CIS courses cover more or less the following material (not necessarily in this order): (1) hardware/software concepts, (2) programming, (3) data management, (4) spreadsheet modeling, and (5) other topics. Of these modules, *programming* is by far the most controversial and difficult subject to teach, to such an extent that some schools have decided to drop it altogether from their introductory curricula.

Indeed, the role of programming in business education always raises two pressing questions: (1) Is it desirable to expose novice students (and particularly nonmajors) to hands-on programming? and (2) Is it possible to cover programming in a meaningful way in, say, four to eight class meetings? After teaching about twenty introductory courses at NYU and at Wharton at the undergraduate, MBA, and executive-education levels, I am now convinced that the answer to both questions is yes.

First, the hands-on experience with programming promotes a host of educational benefits that cut across any academic major: it sharpens the students' analytical skills, it deepens their intellectual curiosity, and it develops their ability to think logically. The programming experience also teaches invaluable lessons about responsibility, planning, discipline, teamwork, and attention to details. Finally, it exposes the students to realistic business problems, and it gives them a set of tools that can be applied in their future jobs.

Second, I am convinced it is possible to teach IS/CIS material of almost any complexity and sophistication in an introductory course, *provided you find a simple and intriguing way to present it.* This, however, requires a tremendous investment in instructional materials. In addition to the mandatory class notes, which must be exceptionally clear, one has to develop tutorials, assignments, projects, case studies, application software, and system software. In short, it's a major undertaking, and that's precisely where this book enters the picture.

[1] Richard Dawkins, *The Blind Watchmaker*, Norton, 1987.

How Can I Use This Book in Introductory Courses?

I assume you already spend about half of your introductory course on such subjects as file processing, database management, and systems analysis and design. The glue that converts these "different" topics into a cohesive business information system is programming. If you already give your students an overview of programming by using BASIC, then dBASE and this book are a natural alternative to consider. If you don't teach programming because you don't believe it can be covered in a rigorous and credible way in only a few lectures, then the book might change your mind on this subject.

Since the text and the software are highly modular, you can easily tailor them to support your individual teaching style and desired coverage. Whether you want to expose your students to a survey of conceptual programming ideas, or, rather, to a rigorous view of application development, the book offers a route to accomplish this goal. Furthermore, depending on your teaching needs, the book can be used either to supplement your class instruction, or, alternatively, to serve as a self-study vehicle. So, if you don't want to spend precious contact time on the nitty-gritty details of string processing or date arithmetic, you don't have to; the students can study this material on their own, reading the appropriate sections of the book and doing the numerous exercises as they go along. This will enable you to devote your class meetings to important topics like structured programming, file management, and systems design.

How Can I Use This Book in Advanced Courses?

Instructors of elective IS/CIS courses (database management, systems analysis, and application development) often face the following dilemma: On the one hand, they want their students to be able to build reduced versions of real systems by integrating file processing and database management with structured programming techniques. On the other hand, many students come to these courses with little or no programming preparation.

The book is uniquely positioned to fill this void. Since it is written as a self-study, students can be expected to read it on their own, without class instruction. Indeed, experience has shown that readers with absolutely no computer experience have managed to gain considerable programming clout by reading the book and doing its exercises, without any external assistance. This is possible because in addition to the nuts and bolts of programming, the book covers many programming tricks and traps, and all the necessary details on designing, editing, debugging, and logging dBASE programs.

Why dBASE?

What software products should be used in introductory IS/CIS courses beyond the mandatory spreadsheet program? In other words, which software product is the most effective in demonstrating elementary programming, file processing, systems design, and database management concepts?

During the last decade I have experimented in the classroom with different combinations of Basic, Pascal, dBASE, Paradox, Framework, Lotus, and Quattro. At some point, I had a revelation: I discovered I wasn't interested in teaching software packages and programming languages per se. Instead, I was interested in communicating *ideas* and *principles.* Hence, I sat down and made a list of important topics I wanted to discuss in my course. Then I went back to the software zoo, looking for the product that could demonstrate this material in the most effective and least painful way. This product turned out to be dBASE, for a variety of pragmatic and pedagogical reasons:

- Most people think of dBASE as a powerful database management system with limited programming capabilities. In fact, dBASE is exactly the opposite: it's a powerful programming language with reasonable database management capabilities. Any Basic program can be rewritten in dBASE, resulting in tighter and more structured code. Hence, dBASE can be used to demonstrate programming as well as data management, within the same computational environment.

- Many courses already use a mixture of Basic and dBASE, illustrating programming and data management respectively. This is an undesirable situation, because the costs of switching from one computational environment to another in mid-course are enormous, from both a pedagogical and an administrative standpoint. Clearly, a consistent use of one paradigm throughout the course will serve to minimize confusion and simplify instruction.

- Most of the computers that our students will use in their future jobs will be loaded with some sort of a dBASE-compatible package. If they do any programming or data management beyond school, chances are it will be in dBASE. Basic is rapidly fading out as a commercial tool, and Pascal and C don't seem to catch up as user-oriented languages. dBASE is far from perfect, but it's still the de facto development environment for business applications on personal computers.

But Does dBASE Have a Future?

Well, it certainly has a rather impressive present, with thousands of installed applications and a software base that's probably worth several billion dollars. In a way, dBASE has become the Cobol of personal computers: many companies and consultants shun the idea of replacing it with another environment either because they dread the conversion and retraining trauma or because they've simply learned to live with it.

In all fairness, though, dBASE has many other virtues beyond this shotgun popularity. In spite of certain design peculiarities, it is a potent and relatively friendly environment. Given its vast capabilities, it is perhaps the only package that offers a smooth and linear learning curve, all the way from trivial file processing commands to structured programming to database management (not necessarily in this order, which is another educational virtue of dBASE). Therefore, learning dBASE skills is largely a sequential process: new commands and functions introduce themselves with little fanfare, and everything somehow fits together.

Although dBASE will certainly continue to evolve, in particular in the areas of user interface and database management, its essential programming core—commands like **`IF`**, **`WHILE`**, and **`CASE`**—will stay intact. This is because the programming formalism of dBASE is basically sound (in spite of some deficiencies that the book is careful to mention), and quite satisfactory for most business applications.

What Software Will the Students Have to Purchase?

The answer depends on what version of dBASE you want to use. If you wish to use dBASE III PLUS, there is no need to purchase the software. Information on obtaining the software is available from the Publisher. All the additional software which is mentioned in the text is included in the data disk provided to the instructor.

Under this setting, the students will be completely self-sufficient as far as software goes. They will be able to use dBASE and the book's software either in the school's lab or on their home computers. Once again, this will enable you to divide the book into the parts that you want to cover in class and the parts that are left to the students as homework and self-study.

If you wish to use dBASE IV, the students will have to purchase its educational version, which, unlike the educational version of dBASE III PLUS, is not free. However, please bear in mind that as far as this book is concerned, it doesn't matter which version of dBASE the students have. That is because the book uses a critical core of commands that is identical under both versions (the few exceptions are flagged in the text).

One final remark: If you are considering dBASE IV only because you have to give a brief demonstration of relational database management, please take a look at the program named **`rats.prg`** on the book's data disk. This program, which runs under both dBASE III PLUS and dBASE IV, is a relational interface which allows users to manipulate **`*.DBF`** files as tables, using relational commands like **`PROJECT`** and **`JOIN`**. If all you want to do is spend one or two lectures (and perhaps an assignment) on relational algebra, **`rats`** might be a free and self-explanatory alternative to **`SQL`**.

What's Next?

A sequel of this book, titled *The Art of Systems Design,* is presently available in a preview format (along with additional software, exercises, and transparencies). Picking up where the present book ends, the new book covers the subjects of sequential and indexed processing (both interactive and program-based), systems analysis and design, and relational database management. If you find the present book compatible with your teaching style, you may want to consider the second book as well, perhaps for other classes. For more information about obtaining a preview copy of the new book, please contact the author.

Introduction To the Reader

If you flip through the pages of this book, you will see many examples of programs written in the dBASE language. And yet this book is not about dBASE, and actually it's not about programming either. The subject of this book is how to design computer-based information systems: systems that solve business problems, like airline-reservation systems, and systems that exploit business opportunities, like frequent-flier programs. Such systems may be constructed using a variety of alternative software tools, of which dBASE is just one example.

dBASE is used in this book for four reasons. First, it's a popular software package, especially in small- and medium-sized businesses. Second, it's an effective package, if you know how to handle it. Third, it's a mainstream package, and it lends itself nicely to demonstrating general ideas about programming, systems analysis, and data management. Finally, dBASE skills are highly marketable: even though many people can use dBASE to do some elementary data juggling, only a few people in each organization can design systems in dBASE. As it turns out, these people are in high demand.

The program of the book is as follows: Chapter 1 gives a preview of what it's like to work with a software development environment, either as an end-user or as a systems developer. It also demonstrates that in order to learn the art of systems design, one has to gain a certain degree of technical competence. This is the purpose of Chapter 2, which covers programming building blocks like numeric processing, user-interface design, and logic. Chapter 3 focuses on conditional branching,