The background of the book cover is a close-up photograph of green fern fronds. A single, vibrant red maple leaf is positioned in the lower right quadrant, partially overlapping the green fern leaves. The title text is printed in a bold, yellow, sans-serif font at the top of the cover.

Structured COBOL Programming ***Interactive and Batch Processing***

A red and white library sticker is located in the bottom left corner of the cover, partially overlapping the author's name.

Bernard L. Levite



Structured COBOL Programming: Interactive and Batch Processing

Bernard L. Levite
Jefferson Technical College



boyd & fraser publishing company



An International Thomson Publishing Company

Danvers • Albany • Bonn • Boston • Cincinnati • Detroit • London • Madrid • Melbourne
Mexico City • New York • Paris • San Francisco • Singapore • Tokyo • Toronto • Washington

Acquisitions Editor: Anne E. Hamilton
Project Manager: Christopher Doran
Production Editor: Jean Bermingham
Production Services: Publication Services
Composition: Publication Services
Interior Design: Publication Services
Cover Design: Richard Pepper/Flying Pepper Design
Cover Photo: Joseph Devenney/The Image Bank
Manufacturing Coordinator: Tracy Megison



© 1995 by boyd & fraser publishing company
A division of International Thomson Publishing Inc.

ITP The ITP logo is a trademark under license.

Printed in the United States of America



This book is printed on recycled, acid-free paper that meets Environmental Protection Agency standards.

For more information, contact boyd & fraser publishing company:

boyd & fraser publishing company
One Corporate Place • Ferncroft Village
Danvers, Massachusetts 01923, USA

International Thomson Publishing Europe
Berkshire House 168-173
High Holborn
London, WC1V 7AA, England

Thomas Nelson Australia
102 Dodds Street
South Melbourne 3205
Victoria, Australia

Nelson Canada
1120 Birchmont Road
Scarborough, Ontario
Canada M1K 5G4

International Thomson Editores
Campose Eliseos 385, Piso 7
Col. Polanco
11560 Mexico D.F. Mexico

International Thomson Publishing GmbH
Königswinterer Strasse 418
53227 Bonn, Germany

International Thomson Publishing Asia
221 Henderson Road
#05-10 Henderson Building
Singapore 0315

International Thomson Publishing Japan
Hirakawacho Kyowa Building, 3F
2-2-1 Hirakawacho
Chiyoda-ku, Tokyo 102, Japan

All rights reserved. No part of this book may be used or reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems—without written permission from the publisher.

Names of all products mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective owners. boyd & fraser publishing company disclaims any affiliation, association, or connection with, or sponsorship or endorsement by such owners.

1 2 3 4 5 6 7 8 9 10 BN 8 7 6 5 4

Library of Congress Cataloging-in-Publication Data

Levite, Bernard L.

Structured COBOL programming : interactive and batch processing /
Bernard L. Levite.

p. cm.
Includes index.

ISBN: 0-87709-892-1

1. COBOL (Computer program language) 2. Structured programming.

I. Title.

QA76.73.C25L49 1994

005.13'3—dc20

94-27702

CIP

DEDICATION

To my wife, and best friend, Kim
without whose love and support this dream would not have become a reality

Preface

When I decided to develop and write a COBOL textbook, it was not by whim. I wasn't simply overwhelmed one day with an instant desire to write a book and didn't sit down the next day to begin writing. I have been teaching COBOL for over twenty years (among other subjects) and have gradually "adopted" the language as my favorite. However, over that period of time my techniques for teaching the language have changed, mostly because computer technology has experienced major changes over that period of time. I began teaching COBOL in the early 1970s using punched cards and, of course, switched gears when punched card systems faded. In the meantime, I had done some contract programming at some local businesses, giving me the opportunity to use my COBOL knowledge in real applications. One of the things I noticed was that many of the programs on which I worked used interactive processing techniques more frequently than batch. Since I was coming from a punched card environment, this was a major adjustment in my way of thinking. I must admit, however, that this experience broadened my abilities as an instructor of the COBOL language. From my programming experiences, I was all set to include interactive processing as a major part of my COBOL courses, and did begin including these concepts in the early 1980s. Textbooks at that time, however, did not cover any interactive concepts—only batch processing. As a matter of fact, interactive processing is just now beginning to surface in the newer COBOL books on the market.

This issue leads back to my main reasons for writing a COBOL textbook. I noticed through my experiences that what was being written in COBOL textbooks was not consistent with what was being used by COBOL programmers in the "real" programming world. I believed when I began this project (and still do) that the COBOL being covered in textbooks and in classrooms should be the same COBOL that is being used in real business applications. I felt I would be doing my students a major disservice to teach them only batch concepts and then have them discover, after graduation, that there is a whole other side to the language. I also noticed that when I began teaching interactive techniques, the students caught on more quickly to the various topics, mostly because interactive processing uses visual forms of input and output. Batch processing relies on input from a mass storage device, which is not visual in any way. I don't mean to imply that batch processing is a dead concept; only that in today's world it is not the only way to use the COBOL language. Hence, this textbook attempts to use both concepts heavily, which is, in my opinion, consistent with business programming techniques used today.

This textbook is written for beginning and advanced courses in COBOL and may be taught to students who are majoring in either a computer-related discipline or any other area. The text should be able to cover a two-semester course; however, some chapters may be omitted if a shorter time period is available.

Many features of this text merit some mention here:

Organization (Part I vs. Part II)

The text, you'll notice, is divided into two parts. The first includes Chapters 1 through 9, which use visual methods in input and output exclusively (keyboard input; screen and printer output). This portion of the text is geared mostly toward the basic concepts, organization, and structure of the COBOL language itself. The second part of the text (Chapters 10 through 18) begins to employ disk files and places more emphasis on business applications, in addition to more advanced COBOL concepts. As mentioned above, interactive and batch concepts are heavily covered throughout the text.

The First COBOL Program

You have probably noticed that the first complete COBOL program is not presented until Chapter 4. It has been my experience that students become overwhelmed at the sight of a COBOL program when they haven't yet been exposed to the various concepts of the divisions, sections, and other features contained therein. The result is that students are faced with a comparatively lengthy set of programming code that means very little to them. The initial reaction on the part of the student is discouragement that can and should be avoided.

This text assumes that the student has no prior programming experience or knowledge. It eases the student into the COBOL language and programming in general.

COBOL '85 vs. COBOL '74

Most COBOL compilers in use today employ the COBOL '85 standard; hence, most of the discussions and descriptions refer to that standard. Several concepts, however, include comparisons of features between the two standards, particularly where there are major differences.

Mainframe vs. Microcomputer COBOL

With the increased popularity of the microcomputer, COBOL is no longer only a mainframe-based language. Many institutions and businesses use COBOL in both environments (mainframe and microcomputers) or, in many cases, exclusively on microcomputers. The discussions in this textbook are "generic" in that they will apply in either setting. The only exceptions are the discussions regarding the `ACCEPT` and `DISPLAY` screen options, which are not standard features, and the coverage in Chapter 18 of RM/COBOL features.

COBOL Compilers

A major portion of Chapter 18 is devoted to microcomputer-based RM/COBOL and its features. There are many popular compilers available today, most of which can be used in conjunction with Chapter 18.

Adopters of the compiler version of this text (ISBN 0-87709-894-8) will receive the RM/COBOL compiler, educational version 5.2. This software includes an editor as part of the project management system RM/CO*. Full documentation for RM/COBOL-85 can be purchased from:

Liant Software Corporation
8911 Capital of Texas Highway North
Austin, TX 78759
1-800-RMCOBOL

Features

Examples and Figures This textbook, you'll notice, offers a large number of examples and figures, with discussions regarding each. There are many complete programs, and almost all of them have flowcharts and pseudocode representations.

Structured Programming Heavy coverage of structured programming concepts and techniques begins in Chapter 6 and continues throughout the remainder of the text.

Advice Paragraphs Throughout the textbook, you'll find many "Advice" paragraphs, which suggest more efficient ways of achieving various results. These should be extremely helpful to both beginning and advanced programming students.

Program Maintenance Program maintenance is explained in detail and encouraged through many program projects that require modification of existing programs.

Tables Chapter 11 covers one- and two-dimensional tables in detail. This includes thorough explanations of sequential access and direct access of table elements. The chapter also offers detailed descriptions of the `SEARCH` and `SEARCH ALL` statements.

Indexed Files Indexed files are heavily covered in Chapter 15 with file updating shown in batch mode as well as in an interactive mode. This topic illustrates one of the most powerful uses of interactive processing. How these two concepts complement each other is explained thoroughly with clear, understandable examples.

Self-Tests; Exercises; Program Projects There are several Self-Test questions at the end of each chapter; the answers are included in Appendix E at the end of the book. There are also other exercises available at the end of many chapters; in addition, most chapters offer program projects (87 projects in all), suggested solutions to which are offered in the Instructor's Manual.

Glossary A complete, thorough glossary is offered following the appendices, including definitions of terms from the ANSI-85 COBOL Standard in addition to terms from the textbook itself.

Documentation Techniques and Forms A thorough explanation of documentation techniques is included in Appendix F, with a completely documented program (from Chapter 10) offered as an example. In addition, standard documentation forms are included at the back of the book for student use.

Instructor's Manual

The Instructor's Manual offers suggested solutions to all end-of-chapter program projects. Also included are brief explanations of the topics within each chapter, so that the instructor can organize lectures to fit his or her needs and time constraints. In addition, many of the figures from the text are enlarged and included as transparency masters.

Acknowledgments

During the process of developing, organizing, and writing this textbook, there were several people who reviewed the manuscript and offered suggestions for improvement. Many thanks to the following reviewers for their input:

Cecilia Browning
Tulane University

Steve Deam
Milwaukee Area Technical College

John Humphrey
Asheville Buncombe Technical Community College

Donald Musselman
James Madison University

Jarrold Rupe
Eastern New Mexico State University

I also wish to offer my gratitude to the professional staff at boyd & fraser for their help throughout this writing experience. I especially want to thank Jim Edwards for giving me the opportunity to write for boyd & fraser to begin with; Chris Doran, my Project Manager, whose guidance and encouragement helped me to mold this text into its final form; Anne Hamilton, Acquisitions Editor; Jean Bermingham, Production Editor; and all the many other people involved in bringing this project to a successful completion.

I also wish to thank Chris Cochrane, Jerome Colburn, and the staff at Publication Services for all their help in attempting to perfect this text.

To my colleagues and my former and present students at Jefferson Technical College: Without having had the experiences of working with you, I could never have begun a project of this magnitude. Many thanks to each of you.

To my wife, Kim: Thank you for your many hours of keying and proof-reading. We always work well together, and this text is proof of what our team effort can achieve.

To my children, Jeneé and Daniel: This book proves that education and hard work can help you reach your goals. As you grow, decide on your attainable goals and then make them happen!

To my in-laws, Jane, Paul, Rick, and Lisa Hlivko: Your prayers throughout this endeavor have meant more than you will ever know.

To my mother and father, Miriam F. and David M. Levite: Your love and encouragement helped me to become what I am today.

In Memoriam
David M. Levite
1912–1992

Bernard L. Levite
Professor/Director of Data Processing
Jefferson Technical College
Steubenville, OH 43952



Introduction

The COBOL language was designed to be used for business applications, with the idea that the instructions be readable. To a person who has no knowledge of programming, a COBOL program will still be understandable to a certain degree. The complexity of a program and how organized the author was will have a great effect on just how readable the program is. An example of a COBOL statement might be

```
MULTIPLY RATE-OF-PAY BY HOURS-WORKED  
GIVING GROSS-PAY.
```

This statement does exactly what a person might think it does. As one can clearly see, COBOL is very English-oriented, even to the point of placing a period at the end of each statement.

There is more we can say about COBOL that cannot be stated about many other languages. In addition to its readability, COBOL is a very well-organized language. Many other languages, such as BASIC and FORTRAN, have many desirable features but are not designed to be well-organized. The programmer using such languages simply begins with whatever variables seem useful and “jumps right into” whatever coding seems capable of doing the job. While that accomplishes what the programmer wants to do quickly, it is not always the most efficient way to solve a problem. In a COBOL program, it is necessary to design, or “set up”, much of the program before the programmer does anything in terms of building the logic. For example, all variable quantities are defined before they are used. If a printed report is desired, or if a mass storage device is needed for later access of information, they must be defined early in the program, long before any actual data is printed or stored. Because COBOL requires that quantities and devices be defined before any actual logic is developed, the programmer is forced to organize the data. While designing and planning is a desirable part of any program, many impatient programmers prefer to just “jump in with both feet” with very little, if any, planning having taken place. COBOL prevents this kind of programming to a large degree, because the structure of a program forces the programmer to be better organized.

COBOL is a “wordy” language, and programs often become quite lengthy. This feature goes with the territory—in order for COBOL to be a readable language, as mentioned earlier, it *has* to be “wordy.” COBOL has for years been the most popular language in the business community. One of the main reasons for COBOL’s popularity (and a very important one) is that COBOL is probably the most standardized language today. This simply means that a program written for one computer will require very few changes in order to

adapt it to another (even from one vendor to another). This is a very important and desirable feature in the business world. It is not at all uncommon for a company to change computers every five or six years, sometimes more frequently and quite often a change from one vendor to another can occur. The conversion process can be quite expensive, as well as time-consuming. Other languages require major changes when converting from one computer to another, while COBOL requires minimal adjustments.

This textbook has been written with the idea of “easing” you into the language and the associated logic. Part I covers such topics as the parts of the computer and how they are used; input and output devices; the hierarchical structure of data used by the COBOL language; and an introduction to the four divisions of a program. Also included in Part I is the logic involved in writing interactive COBOL programs, using a keyboard as input and a video monitor as output, then getting into print files to allow the user to obtain printed output. Part II is more advanced, covering such concepts as sequential disk files, one- and two-dimensional tables, control break reports, sorting, and processing of indexed-sequential disk files. Structured logic is discussed in Chapter 6 and is stressed throughout the book.

You might notice that the concept of disk files has been saved until the second part of the text, while Part I dwells mostly on visual forms of input and output. This involves what is called *interactive processing*, where the user and the computer can “converse.” COBOL was believed for years to be mostly useful for *batch processing*, in which data was fed to the computer from a file (punch cards or disk) and processed with no intervention by the user. With the disappearance (almost) of punch cards and the advent of the keyboard and monitor, COBOL need not be only a batch language any more. Also, it has been the author’s experience that many beginning students have some difficulty with the idea that data can be stored on a mass storage device, such as a disk, and yet cannot be physically viewed by the user. This text allows you to “get into” the COBOL language more gradually and therefore permits you to develop a “trust” in the language and the computer through this interaction. The disk file concept then becomes easier to accept.

In many discussions throughout the text, you will find references to COBOL '85 and COBOL '74. Beginning in the 1960s, the American National Standards Institute (ANSI) attempted to maintain standards in technological areas, one of which was the development of the COBOL language. One of these sets of standards for COBOL was developed in 1968, upgraded in 1974, and then upgraded again in 1985. As mentioned above, several places throughout this book draw comparisons between the 1974 ANSI COBOL and the 1985 version of ANSI COBOL. Most compilers today conform to COBOL '85, on which this textbook is based.

All programs in this text have been tested on a DEC (Digital Equipment Corporation) MicroVAX 3400 computer, but they can be easily converted for any other interactive computer with COBOL capabilities.

I welcome any comments or suggestions anyone using this text may wish to offer.

Brief Contents

PART I Basic Concepts



Chapter 1	Introduction	3
Chapter 2	The Basics of COBOL	15
Chapter 3	Getting Started	23
Chapter 4	The PROCEDURE DIVISION—Interactive Input and Output	39
Chapter 5	Arithmetic Statements	63
Chapter 6	Structured Programming and Other PROCEDURE DIVISION Features	83
Chapter 7	Decision Making	113
Chapter 8	More Program Structure	147
Chapter 9	COBOL Programs with a Print File	176

PART 2 Business Applications



Chapter 10	Sequential Disk Files	211
Chapter 11	Tables	245
Chapter 12	Control Break Report Processing	313
Chapter 13	Sorting and Merging	349
Chapter 14	Master—Transaction File Processing	391
Chapter 15	Indexed-Sequential Disk Files	421
Chapter 16	The COPY Statement and Libraries	470
Chapter 17	The CALL Statement and Subprogramming	487
Chapter 18	Specialty Features of COBOL	503







Contents









Preface
Introduction

xiii
xix

Part I	Basic Concepts		I
CHAPTER 1 Introduction	Program Preparation Parts of the Computer	3 6	3
CHAPTER 2 The Basics of COBOL	The Hierarchy of Data The COBOL Language	15 16	15
CHAPTER 3 Getting Started	IDENTIFICATION DIVISION ENVIRONMENT DIVISION DATA DIVISION	23 24 24	23
CHAPTER 4 The PROCEDURE DIVISION— Interactive Input and Output	Sample Program with Explanation Paragraph or Module Names (Rules) Executable Statements and Options	39 43 49	39

CHAPTER 5 Arithmetic Statements 	Computation Verbs 63 MULTIPLY Statement 67 DIVIDE Statement 68 COMPUTE Statement 70 Program Example 75	63
CHAPTER 6 Structured Programming and Other PROCEDURE DIVISION Features 	Structured Logic 83 The PERFORM Statement 89 The MOVE Statement 100 The STOP RUN Statement 107	83
CHAPTER 7 Decision Making 	The IF Statement 114 The IF ... THEN ... ELSE Statement 115 Program Example 118 Multiple Conditions 126 Class Tests 128 Sign Tests 129 Condition-Names 130 Nested IF s 132 The EVALUATE Statement 137	113
CHAPTER 8 More Program Structure 	Interval Problems 147 Accumulators and Counters 161	147
CHAPTER 9 COBOL Programs with a Print File 	ENVIRONMENT DIVISION Revisited 177 DATA DIVISION Revisited 178 Sample Program 183 PROCEDURE DIVISION Revisited 189 Skeleton Program 197	176
Part II		209
CHAPTER 10 Sequential Disk Files 	Creation of a Sequential Disk File 213 Data Validation 227 Retrieval From a Sequential Disk File 228	211
Business Applications		

CHAPTER 11 Tables	 One-Dimensional Tables 245 Sequential Access of Table Elements 250 Random Access to Table Elements 278 Two-Dimensional Tables 280 Variable Length Records 298	245
CHAPTER 12 Control Break Report Processing	 Single-Level Control Break Report 313 Bilevel and Multilevel Control Breaks 331	313
CHAPTER 13 Sorting and Merging	 Character Coding and Sequence 349 The Sort Verb 351 The Merge Statement 379	349
CHAPTER 14 Master-Transaction File Processing	 Master Files 391 Updating a Master File 392 Maintaining a Master File 406	391
CHAPTER 15 Indexed-Sequential Disk Files	 Concept of Indexed Access 422 Creation of an Indexed File 422 Updating an Indexed File 426 Updating and Maintaining an Indexed File 431 Interactive Updating and Maintenance 440 Alternate Record Keys 457	421
CHAPTER 16 The COPY Statement and Libraries	 The COPY Statement 470 The Copy Library 471 Data Dictionary 485	470

CHAPTER 17 The CALL Statement and Sub-programming	The CALL Statement 487 COPY versus CALL 499	487
CHAPTER 18 Specialty Features of COBOL	The STRING Statement 503 The UNSTRING Statement 507 The INSPECT Statement 511 RM/COBOL Features 523	503
Appendices	Appendix A COBOL Reserved Word List 543 Appendix B Common Compilation Errors 547 Appendix C Common Logic Errors 551 Appendix D ASCII and EBCDIC Codes 553 Appendix E Answers to Self-Tests 555 Appendix F Documentation Techniques and Forms 559 Glossary 579 Index 595	543



Part I

BASIC CONCEPTS