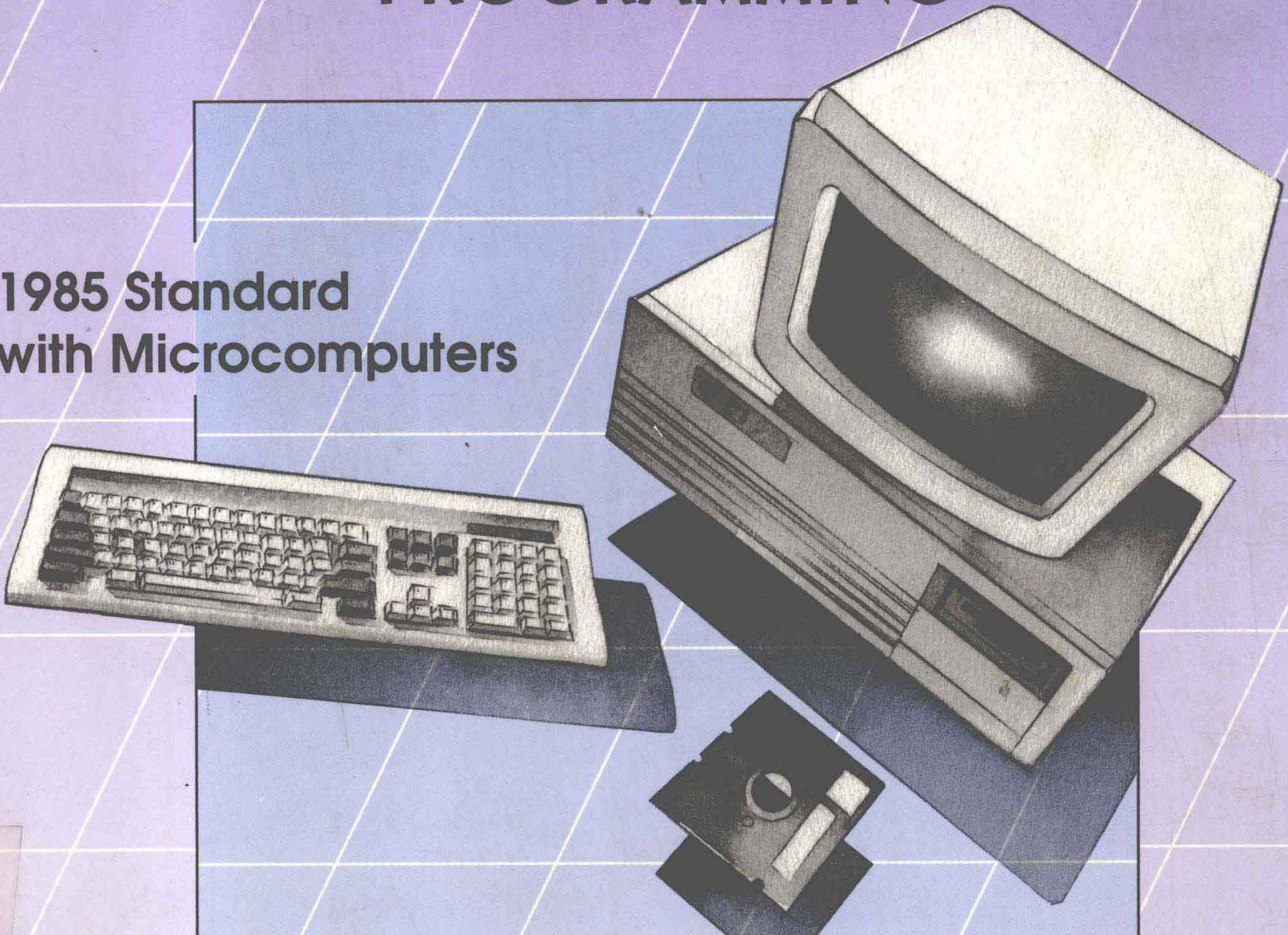
Second Edition

R/M COBOL-85 Compiler from Ryan-McFarland Included

MODERN COBOL PROGRAMMING

1985 Standard with Microcomputers



WILSON PRICE & JACK OLSON

SECOND EDITION

MODERN COBOL PROGRAMMING

WILSON PRICE & JACK OLSON

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

Mitchell **McGraw-Hill**
55 Penny Lane
Watsonville, CA 95076

Modern COBOL Programming, Second Edition

COBOL is an industry language and is not the property of any company or group of organizations. No warranty, expressed or implied, is made by any contributor or by the CODASYL Programming Language Committee as to the accuracy and functioning of the programming system and language. Moreover, no responsibility is assumed by any contributor, or by the committee, in connection therewith.

The authors and copyright holders of the copyrighted material used herein-FLOW-MATIC(trademark of Sperry Rand Corporation), Programming for the UNIVAC I and II, Data Automation Systems, copyrighted 1958, 1959, by Sperry Rand Corporation; IBM Commerical Translator Form No. F28-8013, copyrighted 1959 by IBM;FACT,DSI 27A5260-2760, copyrighted 1960 by Minneapolis-Honeywell-have specifically authorized the use of this material in whole or in part, in the COBOL specifications. Such authorization extends to the reproduction and use of COBOL specifications in programming manuals or similar publications.

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

ISBN 0-07-051044-X

Sponsoring editors: Raleigh S. Wilson and Steve Mitchell
Production supervisor: Pat Moran
Production services: BMR, Corte Madera, CA
Copy edit: Karen Richardson
Cover design: Mike Yazzolino
Interior design: Hugh Anderson, Archetype, Inc.
Interior design consultant: Mike Yazzolino
Page makeup: Archetype, Inc.
Production coordinator: Nancy Taylor Mason
Technical illustrations: Archetype, Inc.
Printer: Von Hoffmann Press, Inc.

Library of Congress Catalog Card No.90-083375

PREFACE

Goals of the Second Edition

In the Preface of the first edition of *Modern COBOL Programming,* we made the following observation:

> Historically, during the 1970s, important programming techniques (program modularization and structured methods) developed by computer scientists found their way into the business data processing environment and COBOL programming. Even though 1974 COBOL was not completely compatible with this theory, it was possible to adapt the theory to the limitations of COBOL. The effect was *very* significant in improved program quality and programmer productivity. The 1985 COBOL Standard represents another important step forward in that it includes features specifically oriented to take advantage of modern programming techniques and theory. The result is that the "kludges" previously used to achieve good structure are no longer necessary. Program code more readily follows from the program design, resulting in programs that are easier to understand, write, and maintain.

The basic philosophy of the first edition was to teach good, solid structured programming techniques utilizing COBOL-85 to the fullest. Since its publication, we have had extensive feedback from numerous satisfied colleagues (and some not-so-satisfied) with constructive comments. Often the observation has been of the nature: "The way you handled topic X was great, *but* if you had ...." The four years since we completed the first edition has given us plenty of time to think about all those topic X's. Our three primary objectives for this revision were

1. To expand the focus on problem solving using numerous example programs, each with complete documentation based on structured techniques.
2. To exploit to the fullest the flexibility that the RM/COBOL-85 compiler provides students when using the personal computer for compiling and running their programs.
3. To include (for optional use) a COBOL work environment from which all functions of entering, editing, compiling, running, and debugging a program are accomplished under a single, unified control system.
4. To expand the book to provide complete COBOL coverage for a full year of COBOL. To this end, heavy emphasis is placed on all types of file processing activities.

It is difficult to compare this book to any of the commonly used textbooks because its focus is oriented around the convenience and power of using a personal computer for COBOL (while maintaining mainframe language capabilities). Until now, the teaching strategy for COBOL has, to an extent, been dictated by the nature of the environment in which the course is being taught: batch processing. In general, the beginning COBOL student must learn much more about COBOL before he or she can write a simple program than, for example, the beginning BASIC or Pascal student.

However, with the availability of ANSI Standard COBOL on today's powerful personal computers, that constraint no longer exists. With the interactive nature of the personal computer, it is possible to use the successful language subset approach in giving the student a small, simple programming assignment using a minimum of COBOL features. Then learning can progress quickly to a variety of topics that include random access to data files as well as the traditional batch processing concepts.

Chapter by Chapter Synopsis

The best way to tell you what the book is all about is through the following set of chapter abstracts.

1 The chapter opens with a brief introductory description of the nature of COBOL. The basic steps of the programming cycle are described. At this point, the student is given a simple interactive program that he or she can compile and then run to gain a feel for these two activities. The sample program is included on the Program/Data disk that accompanies the book so that the student does not need to enter it into the computer.

2 This chapter describes a basic subset of COBOL by examining the program of Chapter 1 and the statements of which it is comprised: ACCEPT, DISPLAY, ADD, PERFORM (in-line version that is a feature of 1985 COBOL), and STOP RUN. In addition, the WORKING STORAGE SECTION of the DATA DIVISION is described. The example used is a simple program to add two numbers, display the result, and give the user the option to repeat the sequence or terminate. At the end of this chapter, the student can write programs. Note that there are no files to open and close, no input/output records to worry about, no file descriptions, and so on.

3 Basic programming standards regarding program modularization and structure to be used throughout the book are explicitly defined in this chapter.

4 The student is first introduced to the nature of files: sequential and random. The best way to remove the mystery of a data file is for the student to create one. Through the interactive nature of the personal computer, the student can write simple programs to create a data file and enter records into it through the keyboard.

5 After creating a file, the student learns how to access data from it. This includes the screen display of a sequential file's contents and of selected records from an indexed file. The student also learns how to print a listing of the records in a file.

6 Editing of data for improved appearance of output is described in this chapter.

7 Where the basic forms of COBOL's arithmetic verbs were introduced in Chapter 2, this chapter covers all aspects of arithmetic operations including the COMPUTE verb.

8 By now the student can begin to write some substantial programs. To this end a complete set of standards for structured program design is introduced. The basic form of the IF statement (and its corresponding COBOL-85 END-IF) is introduced at this point to provide more versatility in the nature of the programming problems the student can handle.

9 With the basics mastered, the focus turns to the quality appearance of printed output. Topics in this chapter include printing headings, computing report totals, and principles of page control.

10 One common form of output is hard copy; the other common form is soft copy. This chapter focuses on designing good, functional screen layouts for both input and output of data (from random file access). The X/OPEN Standard SCREEN SECTION is used.

11 The student has already used basic forms of the IF statement. This chapter covers conditional operations in detail. Topics include compound conditionals, the EVALUATE (case structure) statement, and condition names.

12 The important topic of data validation is covered in this chapter via two extensive examples. The first involves traditional batch validation of records in a file. The second is interactive in nature and involves validating data as it is being entered into a file from the keyboard.

13 This chapter covers traditional group totals via two examples: the first a single-level group total application and the second a two-level application.

14 Subscripting and table processing (single- and multiple-level) are covered in this chapter via realistic examples.

15 Sorting, covered in this chapter, include examples of a simple sort and of using input and output procedures with the sort. The flexibility provided by COBOL-85 in defining input and output procedures is reflected in the examples.

16 The manipulation of character string data in COBOL is clumsy and is consequently downplayed in many books. This chapter is devoted exclusively to this topic. It is culminated with an example of converting a delimited file to a fixed format COBOL file format.

17 COBOL-85 is rich with capabilities (extended beyond those of COBOL-74) for using subprograms and achieving program module functional and data independence. This chapter describes calling a separately compiled program together with parameter passing. It also includes an extensive example of combining several related programs as nested programs under a menu access program.

18 This chapter begins advanced file processing topics. Its topics include relative files, changing and deleting records in a file, dynamic file access, alternate keys for indexed files, and exception handling during I-O operations.

19 The focus of this chapter is on batch processing using extensive example programs: conventional sequential master-transaction processing, batch updating an indexed file, and batch maintenance of an indexed file.

20 Corresponding to Chapter 19 this chapter focuses on interactive file activities using two example programs: transaction processing and file maintenance.

21 The culmination of file processing is an extensive order entry/invoice preparation case study. This application brings together a wide variety of principles from previous chapters. It includes a menu control system that provides access to the two relatively independent components of the system: order entry and invoice preparation.

Features

Complete and Self-Contained

Perhaps the most significant feature of this book is that it is complete and self-contained. It includes a wide variety of example programs, exercises, and programming assignments. In addition, the diskettes accompanying the book contain a high-quality COBOL compiler and data sets for all programming assignments. Features of the compiler are described later in this preface.

COBOL-85 Standard

When we discuss this book with our editors and sales representatives, they commonly refer to our work as oriented around "microcomputer COBOL." To that we say "NO!" This book is based on COBOL-85—the same COBOL-85 implemented on mainframes (or any other computer). We take full advantage of scope terminators (END-IF, END-PERFORM, and so on) to use a level of structured design that was not possible with previous versions of COBOL.

Capitalizes on the Versatility of the Personal Computer

The book's principal personal computer aspect is that it takes full advantage of the PC's capability to provide a more interesting and challenging learning situation for the student. In the batch environment of the mainframe, the student's interaction with the computer is minimal—dictated by the nature of the batch environment and by the standards of the particular installation. In the interactive environment of the personal computer, the student is in control. The student can create and delete his or her own data files, write interactive programs, and obtain almost instant turn-around when compiling and debugging.

Emphasizes File Processing Throughout

Because each student is operating with his or her own set of data files and in somewhat of a private environment, data file operations have almost limitless possibilities. For instance, in Chapter 4, the student studies two interactive programs that allow a user to create a data file and enter data into it. The first program adds records to a sequential file; with only minor changes, it is modified to one that adds records to an indexed file. In Chapter 5, the student learns how to access data from the files just created. The example programs illustrate printing a report of data stored in the file and displaying the contents on the screen of a desired record (random access of an indexed file).

Uses the X/OPEN Standard Screen Section

Unfortunately, the COBOL-85 Standard does not include screen handling capabilities. However, another standard is in use, one defined by a consortium of European computer manufacturers (called X/OPEN) as part of their effort to create portability guides. Among other things, this standard defines a SCREEN SECTION for the DATA DIVISION that allows the programmer to remove formatting data for screen I-O from the PROCEDURE DIVISION and place it in the DATA DIVISION where it belongs. RM/COBOL-85 Release 5.0 (included with this book) features a SCREEN SECTION that conforms to the X/OPEN standard.

Includes 40 Complete Example Programs

Students can learn COBOL syntax by studying syntax examples; they learn how to design good structured programs by studying good structured examples. To that end, 40 complete example programs and variations are included.

Comprehensive Program Design Documentation

Each new example program includes a full definition of the program requirements, plus comprehensive program documentation including a structure chart, a flowchart, and a pseudocode solution. The same degree of documentation is used throughout; standardization is emphasized.

Extensive End-of-Chapter Materials

Each chapter includes a variety of end-of-chapter materials. Where appropriate, the chapter summary includes:

- A general summary describing the primary topics of the chapter.
- A summary of COBOL language elements, including the general syntax format of each COBOL form presented in the chapter.
- Programming conventions established in the chapter.
- A description of error prevention/detection topics relative to the chapter.

Questions and exercises are grouped according to the following types:

- Key terminology introduced in the chapter.
- General questions covering the topics of the chapter.
- Questions relating specifically to example programs described in the chapter.
- Writing program statements that test the student's ability to apply the COBOL syntax learned in the chapter.

Numerous Programming Assignments

Each chapter includes several programming assignments that relate directly to the topics covered in the chapter. Beginning with Chapter 4, each assignment includes a standardized description of the problem, its input and output requirements, and processing requirements. This is the same format used in defining example programs within the chapter.

RM/COSTAR

The software package accompanying this book includes a control system called RM/COSTAR (identified as RM/CO*). RM/CO* is described by Ryan McFarland as "... an interactive, menu-driven environment, within which you can write, edit, compile, run and debug RM/COBOL-85 programs." It is the type of environment that is sometimes referred to as "a programmer's workbench," as opposed to a "normal" or "typical" programming environment. Typical of the conveniences provided through RM/CO* is the way in which the programmer can correct compiler-detected errors. That is, the programmer brings up (on the screen) the post-compile listing, complete with error identifications and descriptions. Then he or she makes corrections by editing the *post-compile listing.* When the correction process is completed, the changes are automatically transferred to the original source program.

However, if you have your own editor for entering and editing programs and your own set of operational procedures, you may use them instead of RM/CO* with no conflicts when using the book. Placement of the detailed description of RM/CO* in Appendix C ensures this.

A Truly Instructional Package

Extensive Teaching Ancillaries

The second edition is more than simply a textbook; it is a teaching package designed to provide broad support. The package consists of

- A comprehensive Instructor's Guide that includes chapter outlines with teaching suggestions and answers to end-of-chapter questions and exercises.
- A solutions manual with a program solution, together with the program output for each programming assignment in the text. A separate section includes listings of all data files included with the text for these assignments.
- A set of transparency masters.
- A solutions disk that contains a programmed solution for each programming assignment in the book.
- Needed Ryan-McFarland manuals.

Diskettes that Accompany the Text

The text is available with software that allows students to complete their assignments on personal computers.

Data for Programming Assignments and Example Programs

A separate Program/Data disk includes data files for every programming assignment in the book. Each file includes a quantity and variety of records sufficient to test all aspects of the student's programs and to yield meaningful reports. This disk also contains the solution for every one of the 40 example programs described in the text.

The Ryan-McFarland Compiler and Runtime System

The compiler and runtime system combination included on the diskettes accompanying this book are a special instructional version of the highly successful Ryan-McFarland RM/COBOL-85, Version 5.0. The instructional version has surprisingly few limitations (over the commercial version) and is completely adequate for almost all assignments and projects that would be assigned in a COBOL course. It includes the following features:

1. COBOL programs of up to 1,000 lines of source code can be compiled.
2. File types supported include sequential, line sequential, relative, and indexed.
3. File size limitation for indexed files is 100 records; for all others, it is 1,000 records.
4. Records in the FILE SECTION can be up to 132 bytes in length.
5. Up to four file definitions may be used in any program.
6. Indexed files may include one or two keys.
7. The CALL statement can be implemented to one level. That is, a program can call another, but the called program may not call a third program.

This is indeed a powerful system for the student.

Ryan-McFarland RM/CO*

The diskettes distributed with the text also include a copy of RM/CO*, the Ryan-McFarland program manager that coordinates all activities of using and managing COBOL programs. It also includes an editor that the student can use for entering and editing programs.

For the instructor, the text and the ancillary materials represent a complete, comprehensive teaching package. For the student, the textbook together with data/software diskettes represent a complete and economical student package. This is the right way to provide for a COBOL course.

Acknowledgments

This book is obviously not the product of our minds alone. We have gained and learned from many others: students, colleagues, and fellow authors. We would like to express our special appreciation to the many individuals who believed in this project and made valuable contributions. At Business Media Resources, Raleigh Wilson provided the resources when needed and Jane Granoff provided overall project coordination. We cannot say enough about the help Karen Richardson has been; she has gone far beyond her normal copy editing duties in ensuring correctness. Nancy Taylor Mason coordinated typesetting and getting the work completed on time. Getting down to nuts and bolts, Hugh Anderson, Richard Stum, Kirk Wettlaufer, Renee Schreiner, Michael Williams and Doris Anderson of Archetype, Incorporated of Denver, Colorado did a fine job of typesetting in a remarkably short time. To all of the contributors, we offer a special "thank you."

Wilson Price
Jack Olson

CONTENTS

CONTENTS

CHAPTER 8

CHAPTER 9

CHAPTER 10

CHAPTER 18

CHAPTER 19

CHAPTER 20

CHAPTER 21

APPENDICES