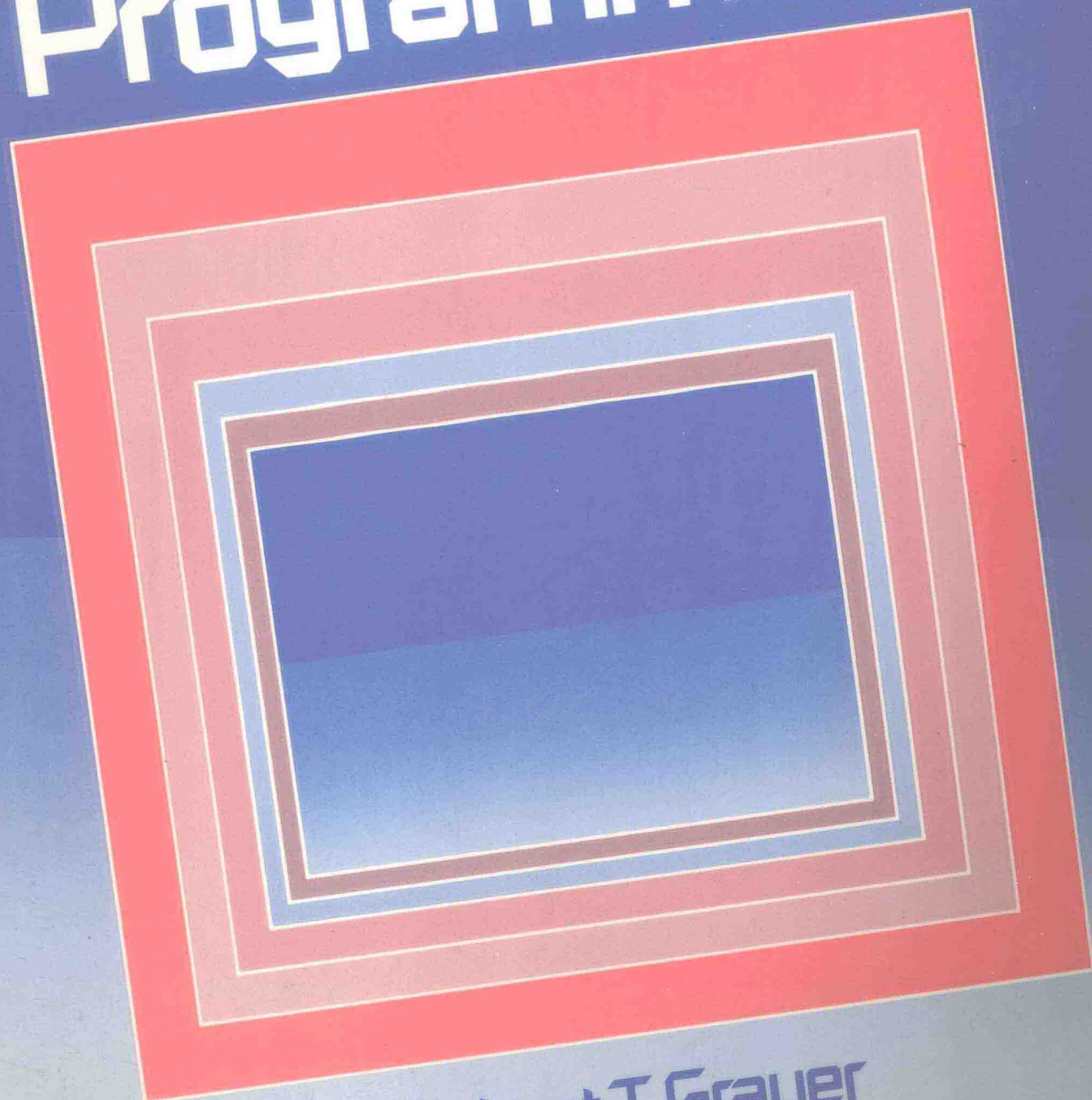


Structured COBOL Programming



Robert T. Grauer

Structured COBOL Programming

Robert T. Grauer, Ph. D.
University of Miami, Florida

PRENTICE-HALL, INC., Englewood Cliffs, New Jersey 07632

Editorial/production supervision: Lynn S. Frankel

Interior and cover design: Anne T. Bonanno

Manufacturing buyer: Gordon Osbourne

© 1985 by Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

Printed in the United States of America

10 9 8

ISBN 0-13-854217-1 01

Prentice-Hall International, Inc., London

Prentice-Hall of Australia Pty. Limited, Sydney

Editores Prentice-Hall do Brasil, Ltda., Rio de Janeiro

Prentice-Hall Canada Inc., Toronto

Prentice-Hall Hispanoamericana, S. A., Mexico

Prentice-Hall of India Private Limited, New Delhi

Prentice-Hall of Japan, Inc., Tokyo

Prentice-Hall of Southeast Asia Pte. Ltd., Singapore

Whitehall Books Limited, Wellington, New Zealand

Structured COBOL Programming

To Marion, Benjy, and Jessica

The following information is reprinted from COBOL Edition 1965, published by the Conference on Data Systems Languages (CODASYL), and printed by the U.S. Government Printing Office:

Any organization interested in reproducing the COBOL report and specifications in whole or part, using ideas taken from this report as the basis for an instructional manual or for any other purpose is free to do so. However, all such organizations are requested to reproduce this section as part of the introduction to the document. Those using a short passage, as in a book review, are requested to mention "COBOL" in acknowledgment of the source, but need not quote this entire section.

COBOL is an industry language and is not the property of any company or group of companies, or of any organization or group of organizations.

No warranty, expressed or implied, is made by any contributor or by the COBOL 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.

Procedures have been established for the maintenance of COBOL. Inquiries concerning the procedures for proposing changes should be directed to the Executive Committee of the Conference on Data Systems Languages.

The authors and copyright holders of the copyrighted material used herein:

FLOWMATIC (Trade mark of the Sperry Rand Corporation), Programming for the Univac (R) I and II, Data Automation Systems copyrighted 1958, 1959, by Sperry Rand Corporation; IBM Commercial 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 of similar publications.

Preface

Structured COBOL Programming is a truly comprehensive work, providing in a single source all subjects normally covered in the one-year COBOL sequence. The scope is extensive, ranging from an introduction to COBOL to maintaining sequential and nonsequential files. The single-volume presentation affords a unified approach and eliminates the need for separate texts in the two semesters. This is especially valuable when students transfer schools (e.g., from a community college) for the second semester, as it eliminates the problem of what may or may not have been previously covered.

Structured COBOL Programming represents several years of effort, encompassing many revisions to earlier books by the same author. It embodies suggestions from hundreds of individuals, both students and instructors, who have used one Grauer book or another. Many new features have been added, while others have been retained, with the following of special interest:

- *Immediate entry into COBOL programming*, beginning in Chapter 1. Programming is learned by doing, and the book has the student writing a complete program from the very beginning. Chapter 2 continues the discussion, by having the reader “punch and run” the program of Chapter 1 in a thorough introduction to the programming process.
- *Twenty-Two Complete COBOL Programs*, which reinforce the discussion in the text and serve as both pedagogical aids and subsequent reference material. Each illustrative program is presented in a uniform and detailed format including program narrative, record layouts, report layouts, test data, and processing specifications.
- *Increased Number of Programming Assignments*, which appear at the end of each chapter. The assignments are presented in the same format as the illustrative programs.
- *Abundant and Challenging Exercises*, at the end of each chapter. The number of exercises has been increased from earlier books, and the problems are constructed to reinforce and stimulate the reader’s ability in COBOL.
- *New Debugging Workshops*, featuring an additional eight COBOL programs to give students practice in an area where it is needed most.
- *Ten Programming Tips*, which go beyond the syntactical rules of COBOL and suggest stylistic considerations to make programs easier to read. Many of the tips take the form of coding standards and are typical of what the student will encounter in “the real world.”
- *Extensive Use of Graphic Aids*, including a two-color presentation and abundant use of tints. This enables important elements, such as COBOL syntax and 8X changes and enhancements, to be consistently highlighted from chapter to chapter.

- *New System Concept Presentations*, at the beginning of each of the advanced COBOL chapters. Instruction in COBOL has come to require additional material, beyond the language itself. These discussions include control breaks, data validation, logical versus physical records, techniques for table lookups and initialization, sorting, Report Writer, the balance line algorithm for file maintenance, and organization of indexed files.
- *Thorough presentation of structured methodology*, with emphasis on hierarchy charts and pseudocode, and deemphasis of the traditional flowchart. Material is also included on stepwise refinement, top-down testing, and Nassi-Shneiderman diagrams.
- *Substantial coverage of the 8X compiler*, which is both thorough yet nonconfusing to those familiar with COBOL 74.

APPROACH TO THE NEW COMPILER

The text is designed for users of both the 74 compiler and the new 8X version. All listings were run under COBOL 74 and are upwardly compatible. In addition each chapter contains a concluding section that presents the relevant 8X changes and enhancements. Appendices A, B, and C contain still more COBOL 8X material, including a sample program. Appendix D contains the 8X Language Summary.

The moderate approach to the new compiler was carefully chosen. It would have been possible, for example, to inundate the reader with new features of COBOL 8X to the point of making the illustrative programs totally incompatible with COBOL 74. To have done so would have greatly weakened the book's overall impact. It took many years for industry to convert from COBOL 68 to COBOL 74, and history has a tendency to repeat itself.

ACKNOWLEDGMENTS

The author wishes to thank the many people who have made this book possible. He acknowledges the contributions his students have made over the years through their many suggestions and active class participation. He is indebted to the in-depth analysis provided by his reviewers, including: Professor Earl Adams (Illinois Central College), Professor Carol C. Grimm (Palm Beach Community College), Professor Margot Hummer (University of Miami), Professor Lewis Myers, Professor James Payne (Kellog Community College), Professor Joel Stutz (University of Miami), Professor Terry Walker (University of Southwestern Louisiana), and Mr. Richard Weiland.

Cathy Benway, Jackie Clark, and Lita Fox, all of the University of Miami, helped to proofread the book. Each made a significant contribution. Ms. Sheila Grossman typed, then retyped the manuscript, always maintaining her sense of humor.

Jim Fegen and Marcia Horton are the acquisition editors responsible for this project. Lynn Frankel, the production editor, put all pieces together in her expert way. Anne Bonanno designed the book and its cover. Herb Daehnke did his usual expert job on the computer listings. Karen Fortgang put on the finishing touches. Last, and certainly not least, thanks to Joe Heider, marketing manager, for making the book a success.

A final word of thanks to the hundreds of unnamed students in MAS223 at the University of Miami who made it all worthwhile.

ROBERT T. GRAUER

Contents

PREFACE xv

1 *Introduction* 1

Overview 1
THE FIRST PROBLEM 1
 Flowcharts 4
 Pseudocode 6
A FIRST LOOK AT COBOL 7
 Elements of COBOL 11
Summary 15
True/False Exercises 16
Problems 16

2 *The Programming Process* 20

Overview 20
A PROBLEM SOLVING PROCEDURE 20
THE COBOL CODING FORM 22
USE OF A TEXT EDITOR 23
SUBMITTING A PROGRAM TO THE COMPUTER 29
 The Compile, Link, and Go Sequence 29
 Output 29
PUTTING IT TOGETHER 30
 Murphy's Law 31
 Compilation Errors 32
 Execution Errors 33
 Discussion 34
Summary 34
True/False Exercises 34
Problems 35
Projects 38

3 *The Identification, Environment, and Data Divisions* 42

Overview	42
COBOL NOTATION	42
IDENTIFICATION DIVISION	43
ENVIRONMENT DIVISION	44
DATA DIVISION	45
File Section	45
PICTURE Clause	45
Level Numbers	46
WORKING-STORAGE SECTION	47
VALUE Clause	48
Assumed Decimal Point	48
THE ENGINEERING SENIOR PROGRAM EXTENDED	49
COBOL 8X: Changes and Enhancements	52
Summary	53
True/False Exercises	53
Problems	53
Projects	56

4 *The Procedure Division* 61

Overview	61
ARITHMETIC VERBS	61
ADD	61
SUBTRACT	62
MULTIPLY	63
DIVIDE	64
COMPUTE	66
ASSUMED DECIMAL POINT	67
PROGRAMMING TIP: USE COMPUTE FOR MULTIPLE ARITHMETIC OPERATORS	68
The ROUNDED CLAUSE	69
IF	69
The ELSE Clause	69
Significance of the Period	70
Indentation in the IF Statement	71
PERFORM	72
READ	73
Placement of the READ Statement	73
WRITE	74
OPEN	75
CLOSE	75
MOVE	75
Restrictions on the MOVE Statement	76
Alphanumeric Sending Field to Alphanumeric Receiving Field	77
Numeric Sending Field to Numeric Receiving Field	77
STOP RUN	78
ENGINEERING SENIOR PROGRAM CONTINUED	78
COBOL 8X: Changes and Enhancements	81
Summary	82

True/False Exercises	82
Problems	83
Projects	87

5 *Writing a Report Program* 88

Overview	88
EDITING	88
Numeric versus Numeric Edited Fields	91
THE TUITION BILLING PROGRAM	94
Test Data	94
The Hierarchy Chart as a Design Aid	94
Pseudocode	98
The Completed Program	98
PROGRAMMING TIP: AVOID LITERALS	102
CODING STANDARDS	103
Data Division	103
Procedure Division	104
Both Divisions	105
A WELL-WRITTEN PROGRAM	106
COBOL 8X: Changes and Enhancements	106
Summary	109
True/False Exercises	111
Problems	112
Projects	115

6 *Debugging* 121

Overview	121
ERRORS IN COMPILATION	121
A SECOND EXAMPLE	126
ERRORS IN EXECUTION	130
ERROR DETECTION: THE STRUCTURED WALKTHROUGH	133
Summary	136
True/False Exercises	138
Problems	138

7 *Structured Programming and Design* 145

Overview	145
DEFINITION	145
SUFFICIENCY OF THE BASIC STRUCTURES	147
EXTENSION OF THE STRUCTURED THEOREM	147
The Case Construct	147
DO UNTIL Construct	150
THE DEVELOPMENT OF STRUCTURED PROGRAMMING	151
SURVEY OF DOCUMENTATION TECHNIQUES	152

The Traditional Flowchart	152
Pseudocode	152
Nassi-Shneiderman Charts	153
Comparison of Techniques	154
STRUCTURED DESIGN	155
Management Analogy	157
Span of Control	158
Top-Down Development	159
Coupling	160
Cohesion	161
The Yourdon Structure Chart	162
Summary	163
True/False Exercises	163
Problems	164

8 *Control Breaks* 167

Overview	167
SYSTEM CONCEPTS	167
DEVELOPING A TWO-LEVEL CONTROL BREAK PROGRAM	167
The Hierarchy Chart	169
Stepwise Refinement	169
Pseudocode	172
The Completed Program	172
DEVELOPING A TWO-LEVEL CONTROL BREAK PROGRAM	176
Hierarchy Chart	176
The Completed Program	178
Summary	182
True/False Exercises	183
Problems	183
Debugging Workshop	183
Projects	186

9 *More About the Procedure Division* 192

Overview	192
SYSTEMS CONCEPTS: DATA VALIDATION	192
Numeric Test	192
Alphabetic Test	192
Reasonableness Check	193
Consistency Check	193
Checking that a Code Exists	193
Sequence Check	193
Completeness Check	193
Date Check	193
Subscript Check	193
THE IF STATEMENT	194
Class Tests	194
Sign Test	194

Compound Tests	195
Implied Conditions	196
Condition Name Tests (88-Level Entries)	197
Nested IFs	198
NEXT SENTENCE	199
PROGRAMMING TIP: DOWNPLAY EFFICIENCY, BUT CHOOSE THE ALGORITHM CAREFULLY	200
PERFORM	201
Performing Sections	201
PERFORM THRU	202
PERFORM UNTIL	203
PERFORM VARYING	203
TIMES Option	204
DISPLAY	205
PROGRAMMING TIP: PERFORM PARAGRAPHS, NOT SECTIONS	206
Use with Microcomputers	206
ACCEPT	207
Obtaining Date and Time of Execution	207
Calculations Involving Dates	207
SIZE ERROR OPTION	208
INSPECT	209
READ INTO	211
WRITE FROM	211
PROGRAMMING TIP: USE READ INTO, WRITE FROM, AND WS BEGINS HERE	212
DUPLICATE DATA NAMES	213
Qualification	214
CORRESPONDING Option	214
A DATA-VALIDATION PROGRAM	215
COBOL 8X: Changes and Enhancements	223
Summary	223
True/False Exercises	224
Problems	224
Debugging Workshop	230
Projects	235

10 *More About the Data Division* 240

Overview	240
SYSTEM CONCEPTS	240
Logical versus Physical Records	240
COBOL Implementation	241
TABLES	242
OCCURS Clause	243
Processing a Table	243
A Second Example	244
Problems with the OCCURS Clause	245
Rules for Subscripts	245
The USAGE Clause	246
OCCURS DEPENDING ON	246
Indexes versus Subscripts	247

SET Statement	248
COPY STATEMENT	249
SUBPROGRAMS	250
A SAMPLE PROGRAM	252
PROGRAMMING TIP: PASS A SINGLE 01 PARAMETER TO A SUBPROGRAM	254
COBOL 8X: Changes and Enhancements	258
Summary	259
True/False Exercises	259
Problems	260
Debugging Workshop	262
Projects	264

11 *Table Processing* 270

Overview	270
SYSTEM CONCEPTS	270
Types of Codes	271
Characteristics of Codes	271
Sequential Table Lookup	271
Binary Table Lookup	272
Positional Organization and Direct Lookups	273
COBOL Implementation	275
Initialization by Hard Coding	275
Input-Loaded Tables	275
Sequential Table Lookup via PERFORM VARYING Statement	277
Sequential Table Lookup via SEARCH Statement	278
Binary Table Lookup via SEARCH ALL Statement	280
Direct Lookup	281
A Complete Example	282
The Completed Program	283
PROGRAMMING TIP: RESTRICT SUBSCRIPTS AND SWITCHES TO A SINGLE USE	286
TWO-LEVEL TABLES	287
Multiple OCCURS Clauses	288
PERFORM VARYING with Two Subscripts	290
A Two-Level Table Program	290
COBOL 8X: Changes and Enhancements	296
Summary	297
True/False Exercises	297
Problems	298
Debugging Workshop	300
Projects	303

12 *Sorting* 311

Overview	311
SYSTEM CONCEPTS	311
Collating Sequence	313
COBOL IMPLEMENTATION	314
SORT Verb	314

<i>Related Statements</i>	315
INPUT PROCEDURE/OUTPUT PROCEDURE Option	317
USING/GIVING OPTION	321
INPUT PROCEDURE/OUTPUT PROCEDURE VERSUS USING/GIVING	325
MERGE	325
COBOL 8X: Changes and Enhancements	328
Summary	328
True/False Exercises	329
Problems	329
Debugging Workshop	333
Projects	335

13 *Report Writer* 339

Overview	339
SYSTEM CONCEPTS	340
EXAMPLE 1: TWO CONTROL BREAKS	340
Data Division Requirements	342
Procedure Division Requirements	345
EXAMPLE 2: THREE CONTROL BREAKS	346
EXAMPLE 3: FINER POINTS OF REPORT WRITER	350
HOW REPORT WRITER WORKS	354
REPORT WRITER SYNTAX	355
TYPE Clause	355
RD (Report Description)	356
Report Groups	356
COBOL 8X: Changes and Enhancements	359
Summary	359
True/False Exercises	360
Problems	360
Debugging Workshop	362
Projects	364

14 *Sequential File Maintenance* 368

Overview	368
SYSTEM CONCEPTS	368
Balance Line Algorithm	371
COBOL Case Study	374
Top Down Testing	378
Stubs Program	379
Completed Program	385
PROGRAM MAINTENANCE	389
COBOL EXTENSIONS	391
SELECT Statement	397
COBOL 8X: Changes and Enhancements	398
Summary	398
True/False Exercises	398

Problems 399
Debugging Workshop 401
Projects 406

15 *Nonsequential File Maintenance* 411

Overview 411
SYSTEM CONCEPTS 411
COBOL REQUIREMENTS 414
 Environment Division 415
 Procedure Division 416
COBOL CASE STUDY-NONSEQUENTIAL FILE MAINTENANCE 424
COBOL 8X: Changes and Enhancements 432
Summary 433
True/False Exercises 433
Problems 433
Projects 435

APPENDIX

A *COBOL 8X: Some Changes and Highlights* 437

APPENDIX

B *COBOL 8X: Structured Programming Enhancements* 443

APPENDIX

C *COBOL 8X: Reserved Words* 453

APPENDIX

D *COBOL 8X: Reference Summary* 456

Index 476

Introduction

Overview

This book is about computer programming. In particular it is about COBOL, a widely used commercial programming language. Programming involves the translation of a precise means of problem solution into a form the computer can understand. Programming is necessary because, despite reports to the contrary, computers cannot think for themselves. Instead they do exactly what they have been instructed to do, and these instructions take the form of a computer program. The advantage of the computer stems from its speed and accuracy. It does not do anything that a human being could not do, given sufficient time.

All computer applications consist of three phases: input, processing, and output. Information enters the computer, it is processed (i.e., calculations are performed), and the results are communicated to the user. Input can come from punched cards, a diskette, magnetic tape, a hard disk, computer terminals, or any of a variety of other devices. Processing encompasses the logic to solve a problem, but in actuality all a computer does is add, subtract, multiply, divide, or compare. All logic stems from these basic operations, and the power of the computer comes from its ability to alter a sequence of operations on the basis of the results of a comparison. Output can take several forms. It may consist of the ubiquitous 11 × 14½ computer listing or printout, or it may be payroll checks, computer letters, mailing labels, magnetic tape, etc.

We begin our study of computer programming by describing a single problem and then developing the logic and COBOL program to solve it. This rapid entrance into COBOL is somewhat different from the approach followed by most textbooks, but we believe in learning by doing. There is nothing very mysterious about COBOL programming, so let's get started. □

THE FIRST PROBLEM

Our first problem involves a set of student records, one record per student. Each record contains the student's name, number of completed credits, and the student's major.

We are to produce *a list of engineering students who have completed at least 110 credits*. Our problem is to develop a COBOL program that can process the set of student records to generate the desired report.