

PROGRAMMING

M K ROY • D GHOSH • DASTIDAR

COBOL Programming



M K ROY
D GHOSH DASTIDAR
Jadavpur University
Calcutta



E8053894



Tata McGraw-Hill Publishing Company Limited
NEW DELHI

8063884

COBOL Programming



UNIVERSITY OF DELHI
LIBRARY
STATIONER, DELHI



© 1982, TATA MCGRAW-HILL PUBLISHING COMPANY LIMITED

No part of this publication can be reproduced in any form or by any means without the prior written permission of the publishers

This edition can be exported from India only by the publishers, Tata McGraw-Hill Publishing Company Limited

Sponsoring Editor : Rajiv Beri
Editorial Assistant : Ranjan Kaul
Production Assistant : Anson Babu

Published by Tata McGraw-Hill Publishing Company Limited,
12/4 Asaf Ali Road, New Delhi 110002 and printed at
Rekha Printers Private Limited, A 102/1, Okhla Industrial
Estate, Phase II, New Delhi 110020.

McGraw-Hill Offices

NEW DELHI

New York
St Louis
San Francisco
Auckland
Bogotá
Guatemala
Hamburg
Lisbon
London
Madrid
Mexico
Montreal
Panama
Paris
San Juan
São Paulo
Singapore
Sydney
Tokyo
Toronto

To our Parents

PREFACE

This book is intended for those who are interested in COBOL. It can well serve as a textbook at the undergraduate level where COBOL is included as a part of the curriculum. It can also be used by the participants of the COBOL courses offered under the Continuing Education Programme of the various Indian universities and institutions. Programmers and other computer professionals will also find the book helpful.

A knowledge of computers is not a prerequisite to follow the contents of the book. The various introductory concepts on computer systems and programming are discussed in the first two chapters. The basic aspects of the COBOL language have been introduced in the next five chapters. These have been presented in such a way that having gone through these five chapters, the reader would be able to write, at least, simple COBOL programs. Further details of the basic features as well as other essential features are covered in Chapters 8 to 14. The rest of the book deals with advanced topics.

We faced some difficulty in selecting the material for this book, because there exist different versions of COBOL. The 1974 American National Standard COBOL (called ANSI 74 or ANS 74 COBOL), which is the more recent version of COBOL, includes a number of incompatibilities with the previous version, namely, ANSI 68 COBOL. Most of the compilers that implemented ANSI 68 have been subsequently modified to incorporate some new features but they still retain some old features that are incompatible with ANSI 74 COBOL. Moreover, the ANSI standards leave many features to be defined and implemented by the implementors. As such, in this book, we have taken a middle course. We have assumed a typical implementation of our own and have described the features accordingly. In addition, at the end of each relevant chapter we have discussed the implementation differences of three COBOL compilers, namely, those for B-6700, DEC-10 and ICL-1900. This will help the reader in understanding the nature of differences that exist between one implementation and another. As regards the typical implementation of the text, the description has been based on carefully selected features of ANSI 74 and those of the said three implementations. Moreover, two other implementations, namely, IBM 370 COBOL and NCR VRX COBOL have also been taken into consideration for the said purpose. However, we have not included the implementation differences of these latter named versions, because information regarding IBM 370 COBOL is easily available elsewhere and most of the features as described here do not have much differences with the NCR VRX COBOL. In fact, we find that most of the ANSI 74 features have been faithfully implemented in NCR VRX COBOL.

To avoid any misinterpretation, we would like to spell out clearly the purpose of including implementation differences. The objective is to indicate to the reader where an implementation difference may exist and what its nature may be. The notes given under implementation differences, therefore, point out only some of the important and interesting differences. They certainly do not reveal all differences nor do they show the relative merits and demerits of the said compilers. Moreover, the information given under implementation differences are subject to change as the compilers are normally modified frequently. As such, the readers should not take these notes too seriously. In order to make use of implementation dependent features, one must consult the relevant manuals.

It will be wrong to assume that COBOL is an implementation dependent language. In fact, there are more similarities than differences. The differences arise only when one wants to make specific use of the language elements. A knowledge of implementation differences helps one to avoid writing the implementation dependent code. This makes a program portable so that it can be executed on different machines with minimum change.

COBOL is a powerful language and offers a lot of facilities. The power of COBOL cannot be fully exploited if one learns only a subset of COBOL that enables him to write an application program. This book, therefore, attempts to present the language features with as many details as may be useful. The book also covers a substantial part of ANSI 74. In fact, except for the debugging and communication modules of ANSI 74, all other language features have been included.

We thankfully acknowledge the help that we have obtained from the various published materials, a list of which is given in the bibliography. We are indebted to many of our friends and colleagues with whom we have had discussions regarding the contents and structure of this book. Notable among them are Mr. T.K. Basu of ICL (India) Pvt. Ltd., Mr. A. Banerjee, Mr. A.K. Majhi, Dr. Subrata Roy and Mr. D.P. Sinha. Our special thanks are to Sri Priyatosh Chakraborty for typing the manuscript and to Sri Samaresh Bhattacharya for drawing the figures.

Finally, we would like to include an apology to those readers who belong to the fair sex. In the text we have referred to the programmer or reader as "he". This may be taken to mean "he or she".

ACKNOWLEDGEMENTS

The following acknowledgement is reprinted from the American National Standard Programming Language COBOL, X3.23-1974 published by the American National Standards Institute, Inc.

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 Commercial Translator Form No. F 28-8013, 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.

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 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.

M K Roy
D Ghosh Dastidar

CHAPTER 10: COBOL PROGRAMMING

10.1 INTRODUCTION

The purpose of this chapter is to provide a comprehensive overview of COBOL programming. It covers the basic syntax, data types, and control structures of the COBOL language. The chapter is organized into several sections, each focusing on a specific aspect of COBOL programming. The first section discusses the general structure of a COBOL program, including the identification division, environment division, data division, and procedure division. The second section covers the basic data types and data structures supported by COBOL, such as alphanumeric, numeric, and pointer data types. The third section discusses the control structures of COBOL, including the IF, EVALUATE, and PERFORM statements. The fourth section covers the file handling capabilities of COBOL, including the OPEN, READ, WRITE, and CLOSE statements. The fifth section discusses the debugging techniques for COBOL programs, including the use of the DEBUG statement and the COBOL debugger. The sixth section covers the optimization techniques for COBOL programs, including the use of the OPTIMIZE statement and the COBOL optimizer. The seventh section discusses the portability of COBOL programs, including the use of the PORTABLE statement and the COBOL compiler. The eighth section covers the security features of COBOL, including the use of the SECURITY statement and the COBOL security manager. The ninth section discusses the performance of COBOL programs, including the use of the PERFORMANCE statement and the COBOL performance monitor. The tenth section covers the maintenance of COBOL programs, including the use of the MAINTAIN statement and the COBOL maintenance tool. The chapter concludes with a summary of the key points discussed in the previous sections.

0005894

CONTENTS

Preface



1	INTRODUCTION TO COMPUTER SYSTEMS	
1.1	Data Processing	1
1.2	Computer Organization	2
1.3	Internal Representation of Data and Instruction	3
	1.3.1 Binary Number System	3;
	1.3.2 Binary Coding	6;
	1.3.3 Coding for Numeric Data	8;
	1.3.4 Operation Codes	8
1.4	Input and Output	9
	1.4.1 Card Reader	9;
	1.4.2 Line Printer	9;
	1.4.3 Magnetic-tape Drive	11;
	1.4.4 Magnetic-disk Drives	13
1.5	Computer Software	15
	1.5.1 Assemblers and Compilers	15;
	1.5.2 IOCS (Input Output Control Systems)	16;
	1.5.3 Operating Systems	16
1.6	Data Communications	18
	<i>Exercises</i>	20
2	FILE CONCEPTS AND PROGRAM LOGIC	21
2.1	File Concepts	21
2.2	Record Layout	22
	2.2.1 Card Layout	22;
	2.2.2 Report Layout	23;
	2.2.3 General Record Layout	25
2.3	Program Logic - Algorithm	25
2.4	Flow Chart Symbols	26
2.5	Sample Flow Charts	28
2.6	Additional Flow-chart Symbols	30
2.7	More Examples of Flow Charts	32
	2.7.1 Sample Problem - Card to Tape with Validation	30;
	2.7.2 Sample Problem - Recognition of Sequence Break	32
2.8	Decision Tables	36
2.9	Documentation	39
	<i>Exercises</i>	39

3	INTRODUCTION TO COBOL	42
3.1	History of COBOL	41
3.2	Format for COBOL Programs	41
3.3	Structure of a COBOL Program	42
3.4	Character Set	45
3.5	COBOL Words	46
3.6	Data Names and Identifiers	47
3.7	Literals	47
3.8	Figurative Constants	49
3.9	Language Description Notation	49
3.10	Implementation Differences	52
	<i>Exercises</i>	53
4	IDENTIFICATION AND ENVIRONMENT DIVISION	54
4.1	IDENTIFICATION DIVISION	54
4.2	ENVIRONMENT DIVISION	55
	4.2.1 CONFIGURATION SECTION	55; 4.2.2 INPUT-OUTPUT SECTION 57
4.3	Implementation Differences	58
	<i>Exercises</i>	59
5	FIRST LOOK AT DATA DIVISION	61
5.1	Introduction	61
5.2	Level Structure	62
5.3	Data Entries	63
	5.3.1 PICTURE Clause	64; 5.3.2 VALUE Clause 66
5.4	FILE SECTION	67
5.5	WORKING-STORAGE SECTION	68
5.6	Editing	69
	5.6.1 Edit Characters for Numeric Data	69; 5.6.2 Editing of
	Alphabetic and Alphanumeric Data	75; 5.6.3 Examples of Editing 75;
	5.6.4 SPECIAL-NAMES Paragraph	76
5.7	Classes and Categories of Data	76
5.8	Implementation Differences	78
	<i>Exercises</i>	79
6	PROCEDURE DIVISION AND BASIC VERBS	82
6.1	Structure of the PROCEDURE DIVISION	82
6.2	Data Movement Verb: MOVE	84
6.3	Arithmetic Verbs	87
	6.3.1 ADD	87; 6.3.2 SUBTRACT 88; 6.3.3 MULTIPLY 89;
	6.3.4 DIVIDE	90
6.4	Sequence Control Verbs	91
	6.4.1 GO TO	92; 6.4.2 STOP 92
6.5	Input and Output Verbs	92
	6.5.1 OPEN	92; 6.5.2 READ 93; 6.5.3 WRITE 94; 6.5.4 CLOSE 95;
	6.5.5 ACCEPT	96; 6.5.6 DISPLAY 97

6.6	Condition Verb: IF	97	
6.7	Categories of COBOL Statements	101	
6.8	Implementation Difference	102	
	<i>Exercises</i>	104	
7	WRITING COMPLETE PROGRAMS		108
7.1	Introduction to Program Writing	108	
7.2	A Sample Program	109	
7.3	How to run a COBOL Program	114	
7.4	Program Testing	115	
7.5	Program Style	116	
	<i>Exercises</i>	117	
8	MORE ABOUT DATA DIVISION		119
8.1	USAGE Clause	119	
8.2	SYNCHRONIZED Clause	121	
8.3	JUSTIFIED Clause	122	
8.4	REDEFINES Clause	122	
8.5	RENAMES Clause	123	
8.6	Qualification of Data Names	124	
8.7	SIGN Clause	126	
8.8	Implementation Differences	127	
	<i>Exercises</i>	130	
9	MORE ABOUT DATA MOVEMENT VERB AND ARITHMETIC VERBS		133
9.1	Elementary and Group Moves	133	
9.2	CORRESPONDING Option	137	
	9.2.1 MOVE CORRESPONDING	137;	
	9.2.2 ADD and SUBTRACT		
	CORRESPONDING	138;	
	9.2.3 General Rules Concerning CORRES-		
	PONDING Option	139	
9.3	ROUNDED Option	140	
9.4	ON SIZE ERROR Option	141	
9.5	COMPUTE Verb	142	
9.6	Implementation Differences	144	
	<i>Exercises</i>	145	
10	CONDITIONAL AND SEQUENCE CONTROL VERBS		147
10.1	Condition	147	
	10.1.1 Relational Condition	147;	
	10.1.2 Sign Condition	149;	
	10.1.3 Class Condition	150;	
	10.1.4 Condition Name Condition	151;	
	10.1.5 Negated Simple Condition	153;	
	10.1.6 Compound Condition	153	
10.2	IF Statement	156	
	10.2.1 Nested IF Sentence	158;	
	10.2.2 Coding Style for IF		
	Sentences	160;	
	10.2.3 Decision Tables and IF Statements	161	

10.3	GO TO with DEPENDING Clause	163
10.4	ALTER Statement	163
10.5	PERFORM Statement	165
10.6	EXIT Statement	165
10.7	A Sample Validation Program	167
10.8	Implementation Differences	172
	<i>Exercises</i>	174
11	TABLE HANDLING	177
11.1	OCCURS Clause	177
11.2	Assigning Values to Table Elements	179
11.3	Multi-dimensional Tables	180
11.4	PERFORM Verb and Table Handling	182
	11.4.1 PERFORM with TIMES Option	182; 11.4.2 PERFORM with UNTIL
	Option	183; 11.4.3 PERFORM with VARYING Option
	11.4.4 PERFORM with the VARYING-AFTER Option	186
11.5	Indexed Tables and Index Names	188
11.6	SET Verb	189
11.7	SEARCH Verb	191
	11.7.1 Sorted Tables and Binary Search	195; 11.7.2 Searching
	a Multi-dimensional Table	196
11.8	OCCURS DEPENDING Clause	197
11.9	Sorting a Table	199
11.10	Index Data Item	201
11.11	Use of Indexes and Index Data Items	202
11.12	Implementation Differences	203
	<i>Exercises</i>	204
12	STRUCTURED PROGRAMMING	208
12.1	Program Design	208
12.2	Current Trends in Data Processing	208
12.3	Objective and Methodologies of Structured Programming	209
	12.3.1 Structuring of Control Flow	209; 12.3.2 Modular
	Programming	211; 12.3.3 Top-down Approach
	12.3.4 Summing-Up	214
12.4	Structured Programming in COBOL	214
	12.4.1 Three Basic Structures	214; 12.4.2 Modular Programming
	in COBOL	215; 12.4.3 Combination of Basic Structures
	12.4.4 A Complete Structured Program	218
12.5	Weaknesses of COBOL as a Language for Structured Programming	221
12.6	Structured Flow Charts	224
	<i>Exercises</i>	225
13	SEQUENTIAL FILES	227
13.1	File Characteristics	227
13.2	File-Control Entries for Sequential Files	230
13.3	File Description - Fixed-Length Records	231

13.3.1	BLOCK CONTAINS Clause 232;	13.3.2	RECORD CONTAINS Clause 233;	
	13.3.3 LABEL RECORD Clause 233;	13.3.4	VALUE OF Clause 233;	
	13.3.5 DATA RECORD Clause 234;	13.3.6	CODE-SET Clause 234;	
	13.3.7 Non-Standard Clauses 235;	13.3.8	Examples of File-Description Entries 236	
13.4	Statements for Sequential Files 236			
	13.4.1 OPEN Statement 237;	13.4.2	CLOSE Statement 237;	
	13.4.3 WRITE Statement 238;	13.4.4	REWRITE Statement 238	
13.5	Examples of Sequential File Processing (Fixed-length Records) 238			
13.6	Sequential Files with Variable-length Records 241			
	13.6.1 FD Entry for Variable-length Records 241;	13.6.2	Record Description for Variable Length Records 242;	
	13.6.3 Example of Sequential File Processing (with variable-length records) 243			
13.7	Features for Unit-Record Files 245			
	13.7.1 Special Features for Line-printer Files 245			
13.8	Special Features for Magnetic-tape Files 248			
13.9	I-O-CONTROL Paragraph 251			
13.10	DECLARATIVES and the FILE STATUS Clause 253			
13.11	Implementation Differences 257			
	<i>Exercises</i> 262			
14	SORTING AND MERGING OF FILES			264
14.1	The Simple SORT Verb 264			
14.2	File Updation 267			
14.3	Variations of Updation 272			
	14.3.1 Updation Without Insertion and More Than One Transaction Record for a Master Record 272;	14.3.2	File Matching 274;	
	14.3.3 File Merging 276			
14.4	The Simple MERGE Verb 277			
14.5	INPUT and OUTPUT PROCEDURE in SORT Statement 279			
14.6	An Example of SORT Statement with INPUT/OUTPUT PROCEDURE 282			
14.7	MERGE Verb with OUTPUT PROCEDURE 283			
14.8	SAME SORT AREA Clause 284			
14.9	MEMORY SIZE Clause 284			
14.10	Implementation Differences 284			
	<i>Exercises</i> 285			
15	MORE ABOUT STRUCTURED PROGRAMMING			287
15.1	Constrained Use of GO TO 287			
15.2	GO TO Statement and SORT-MERGE Feature 288			
15.3	GO TO with DEPENDING ON in Structured Program 288			
15.4	Summing-up 290			
15.5	Disciplined Use of COBOL Statements 291			
15.6	Control Breaks and Structured Programming 292			
	<i>Exercises</i> 301			
16	DIRECT ACCESS FILES			302
16.1	Relative Files 302			

16.1.1	FILE-CONTROL Paragraph for Relative Files	302;
16.1.2	PROCEDURE DIVISION Statements for Relative Files	303;
16.1.3	Examples of Relative File Handling	307
16.2	Indexed Sequential Files	308
16.2.1	FILE-CONTROL Paragraph for Indexed Files	309;
16.2.2	PROCEDURE DIVISION Statements for Indexed Files	310;
16.2.3	Updation of Relative and Indexed Files	314;
16.2.4	An Example of Handling Indexed File	315;
16.2.5	File Descriptions for Relative and Indexed Files	317;
16.2.6	DECLARATIVES and FILE STATUS Clause	317;
16.2.7	Direct Organization	319;
16.2.8	Selection of File Organization	319
16.3	Implementation Differences	320
	<i>Exercises</i>	324
17	CHARACTER HANDLING	326
17.1	EXAMINE Verb	327
17.2	INSPECT Verb	329
17.3	STRING and UNSTRING Verbs	332
17.4	Implementation Differences	341
	<i>Exercises</i>	341
18	REPORT WRITER	343
18.1	General Format of a Report	343
18.2	File Section — Report Clause	344
18.3	Outline of Report Section	345
18.4	Report Section — Report-description Entry	346
18.5	Report-group Description	348
18.6	PROCEDURE DIVISION Statements	354
18.7	Sample Program	357
18.8	Implementation Differences	357
	<i>Exercises</i>	363
19	COBOL SUBROUTINES	364
19.1	Structure of a COBOL Subroutine	365
19.2	The Calling of a Subroutine	366
19.3	State of a Subroutine and CANCEL Statement	368
19.4	An Example Illustrating Use of Subroutine	369
19.5	Advantages and Disadvantages of COBOL Subroutines	371
19.6	Implementation Differences	372
	<i>Exercises</i>	376
20	SEGMENTATION AND LIBRARY FACILITY	378
20.1	Segmentation	378
20.1.1	Segmentation Restrictions	379;
20.1.2	Planning for Segmentation	380

20.2	The Library Facility	381
20.3	Implementation Differences	383
	<i>Exercises</i>	383

<i>Appendix A</i>	BIBLIOGRAPHY	385
<i>Appendix B</i>	RESERVED WORD LIST	387
<i>Appendix C</i>	DATA DIVISION FORMATS	412
<i>Index</i>		416

INTRODUCTION TO COMPUTER SYSTEMS

1.1 DATA PROCESSING

Modern computers can work at very high speeds and at the same time are very reliable. The work that is normally done by a computer is called data processing. Note that there are two words, data and processing. Data are entities that relate to a certain person, task or event. Processing means performing systematic operations upon the data to make them more useful. Such manipulations that a computer can perform on the data include calculation, comparison, reading and writing. Calculation refers to the execution of arithmetic operations, such as addition, subtraction, multiplication and division. Besides calculation, a computer can compare two data to determine which one is greater or to establish whether they are equal or not. Based on the result of the comparison it can also choose one of the two courses of action. Reading is the process of accepting data from some external medium and writing is the reverse process of reading. The results of calculations can be written by the computer onto some medium such as a sheet of paper.

The operations mentioned above are only elementary operations. By combining such elementary operations it is possible for the computer to perform processing that is complex in nature. However, the primary objective of data processing is to take the help of the computer to process some given data in a desired manner. Thus a computer can be called a data-processing equipment which accepts some data as its input and produces some "processed data" as its output. The processed data is often called information.

It is of importance to note that a computer cannot do anything on its own. It must be instructed to do a desired processing. That is why it is necessary to specify a sequence of operations that a computer must perform to solve a data processing problem. Such a sequence of operations written in a language that can be understood by a computer is called a program. It is the program that controls the activity of processing by the computer and the computer performs precisely what the program wants it to do. To enable a computer to execute a program automatically it is necessary that the program must be stored somewhere within the computer in advance. This pre-storing of the program is a fundamental requirement and this is why a computer is often referred to as a stored program computer.