

ANSI STRUCTURED COBOL

An Introduction

Steve Teglovic, Jr.
Kenneth D. Douglas

ANSI STRUCTURED COBOL

An Introduction

Steve Teglovic, Jr.

University of Northern Colorado

Kenneth D. Douglas

Southwest Missouri State University

THE ASM FOUNDATION

BOOKS FOR ASIA

SAN FRANCISCO, CALIFORNIA, U.S.A.

美國亞洲基金會敬贈

1986

IRWIN

Homewood, Illinois 60430

© RICHARD D. IRWIN, INC., 1986

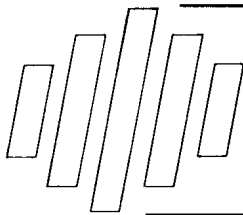
All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher.

ISBN 0-256-03286-6

Library of Congress Catalog Card No. 85-60949

Printed in the United States of America

1 2 3 4 5 6 7 8 9 0 ML 3 2 1 0 9 8 7 6



PREFACE

Industry reports show that COBOL is the most used language for business administrative programs. As a result, a good foundation of the language is needed by a potential programmer in data processing. To help lay this foundation, the authors present the material in this text using a modular approach: They start with the very basic concepts and skills needed, and progress into more difficult concepts. Using this approach, an individual quickly gains confidence in the language and develops good programming techniques.

Chapter 1 is an introduction to hardware and software concepts, and chapter 2 is an early exposure to the structured design of programs. Top-down design is stressed, along with programming structures and structure charts. Chapters 3 and 4 cover the basic components of the COBOL language and how to use standard COBOL formats for an easier understanding of the syntax of the language. In chapter 5, all of the concepts from previous chapters are brought together, and a complete program is discussed from the statement of the problem to the analysis of its output.

Because correcting program errors is an important part of COBOL, chapter 6 covers how to avoid or minimize them, how to use the error detection statements of the language for debugging statements, and how to understand the diagnostics of the language. Chapter 7 builds on the program concepts from earlier chapters and discusses the editing of output data. Chapter 8 brings arithmetic into the program solution. In chapter 9, flowcharts are used to more clearly explain the logical sequence and control of decision making statements. The validation of input data is discussed and applied in a program in chapter 10. One, two, and three level control breaks are presented, using simplified techniques in chapter 11; chapter 12 explains how to use sorting techniques. The difficult topic of table use is presented in a straightforward fashion in chapter 13. Chapter 14 deals with tape and disk file processing. Chapter 15 is a collection of additional topics that are brought together at the end of the text.

Another feature of the text is a section on programming style and on common errors encountered when using the topics in that chapter. Also, chapters 4 through 14 each use complete program examples to cover the new material. Each program has: a statement of the problem; an explanation of the data input and output; a complete program listing; and a discussion of the program logic. Documentation is covered with structure and printer spacing charts. In addition, the authors have implemented a self-documenting technique for numbering and naming procedures used in a program.

The text can be used in a variety of ways. Chapters 6, 10, and 12 can be skipped or used in a different order; they are each independent of other chapters. Chapter 15 can be drawn upon from any of the chapters, as needed, or omitted. An institution using one term to teach COBOL will find the coverage of COBOL topics in the text more than ample for its needs. Those institutions using two terms to teach COBOL will find that this text covers all of the topics necessary for the first term and has topics remaining. Therefore, it is easy to dovetail these topics with those in a second term. A different text can be used for the second course with the assurance that the student has been exposed to all of the topics necessary; or you may use the authors' sequel to this text: *ANSI Structured COBOL: Advanced*, by the same authors. The advanced text is written with the same general format

and style and completely covers all topics necessary to master the COBOL language.

The Appendixes in this text provide complete data sets of all chapter programs and problem assignments, coverage of COBOL language formats and reserved words, and explain the uses of line and full screen editors for creating programs and data sets. In addition, there are answers to selected exercises from each chapter.

An instructor's manual is available with the text. It has solutions to all of the exercises in the text and a bank of objective questions and executed programs with output results for all chapter problems. There are copies of many of the Figures and Programs in the text that can be used as transparencies. The manual saves many hours of programming and preparation time.

The authors wish to express their sincere gratitude to the manuscript reviewers for their comments, suggestions, corrections and constructive criticisms:

Susan D. Haugen
Drake University
Mick L. Watterson
Drake University
Harry Sullivan
Columbus Technical Institute
Jeane A. Schildberg
Chaffey College
Bennett L. Kramer
Massasoit Community College
Bruce M. Johnson, Jr.
Xavier University
Douglas May
Appalachian State University
John A. Morris
Lincoln Land Community College
Edward Stone
Herkimer County Community College
Judy Adamski
Grand Valley State College

Finally, we want to express our sincere gratitude to members of our families for their encouragement during the long writing process. Special thanks and love go to our wives, Mary and Cecil, for their tolerance of our behavior and for their understanding, support, and contributions to our efforts. What would we do without them!

Acknowledgment

The acknowledgment required by the American National Standards Institute for the use of copyrighted material follows:

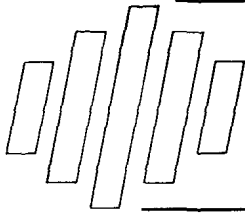
COBOL is an industry language and is not the property of any company or group of companies, or of any organization or group of organizations.

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 herewith.

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. 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.

Steve Teglovic, Jr.
Kenneth D. Douglas



CONTENTS

Chapter 1 ***Introduction to Programming*** **1**

Terms and Statements 1
Cobol History 2
Hardware Concepts 2
 Central Processing Unit 3
 Input, Output, and Auxiliary Storage Devices 4
Software Concepts 6
Batch and Interactive Processing 7
Program Execution 8
Program Development 8
Exercises 9

Chapter 2 ***Structured Design of Programs*** **11**

Terms and Statements 11
Need for Structured Design 12
Programming Structures 13
Top-Down Design 14
Use of Structured Tools 14
 Visual Table of Contents 14
 Pseudocode 17
 IPO Charts 17
 Structured Flowcharts 19
 Warnier-Orr Diagram 20
 Chapin Charts 20
 Nassi-Shneiderman 20
Common Errors 20
Problems 22

Chapter 3 ***Components of a COBOL Program*** **23**

Terms and Statements 23
Use of COBOL Coding Form 24
 Identification and Instruction 24
 Page and Line Number 24
 Comments, Continuation, and Skipping 25
 COBOL Headings and Statements 25
 Program Identification 25
 Program Coding 25
Standard COBOL Format Notation 26

Structure of COBOL	26
Data Organization	27
Level Indicators and Numbers	29
Words and Names	29
Literals	30
Figurative Constants	32
Symbols	32
Common Errors	33
Exercises	33
Problems	34

Chapter 4	<i>Program, Devices, and Data Definition</i>	36
------------------	-----------------------------------------------------	-----------

Terms and Statements	36
IDENTIFICATION Division	37
Required Entries	37
Optional Entries	38
ENVIRONMENT Division	39
CONFIGURATION Section	39
INPUT-OUTPUT Section	40
DATA Division	41
FILE Section	41
WORKING-STORAGE Section	46
Program Example	50
Statement of the Problem	50
Input	51
Output	51
Program	51
Discussion	53
Programming Style	53
Common Errors	54
Exercises	54
Problems	55

Chapter 5	<i>PROCEDURE Division</i>	58
------------------	----------------------------------	-----------

Terms and Statements	58
PROCEDURE Division	59
General Entries	59
Statement Entries	59
Program Example	74
Statement of the Problem	74
Input	74
Output	74
Structure Chart	74
Program	75
Discussion	77
Programming Style	77
Common Errors	78
Exercises	78
Problems	79

Chapter 6 *Debugging and Diagnostics* 81

Terms and Statements	81
Program Compilation	82
Types of Errors	82
Error Correction	82
Desk Check	82
The DISPLAY Statement	83
The EXHIBIT Statement	83
The TRACE Statement	87
The Debugging Module	89
Diagnostics	93
Structured Programming	98

Chapter 7 *Headings and Edited Output* 99

Terms and Statements	99
Data Movement	100
Overpunched Code for Signed Numeric Input Fields	100
The SIGN Clause	100
Moving Decimal Numeric Literals	101
Absolute and Algebraic Values	102
The JUSTIFIED RIGHT Clause	102
The Figurative Constant ALL	103
Time and Date from the CPU	104
READ Statement with INTO Option	105
Output Editing	105
Decimal Point Insertion	106
Zero Suppression	106
Comma Insertion	107
Zero Suppression with Asterisk Insertion	107
Fixed Dollar Sign Insertion	108
Fixed Minus Sign Insertion	108
Fixed Plus Sign Insertion	108
Floating Dollar, Minus, and Plus Sign Insertion	109
CR and DB Insertion	109
Alphanumeric Insertion	110
Decimal Point Scaling	110
Blanking Out a Field	111
Output Techniques	111
The WRITE Statement with Data-Name Option	111
The WRITE Statement with FROM Option	112
Double Buffering	112
Program Example	114
Statement of the Problem	114
Input	114
Output	114
Structure Chart	114
Program	115
Discussion	118
Programming Style	120
Common Errors	120
Exercises	120
Problems	122

Chapter 8 *Arithmetic***124**

Terms and Statements	124
Initialization of Numeric Fields	125
VALUE Clauses	125
Elementary Moves	125
Group Moves	125
Signed Numeric Fields	126
The ADD Statement	126
The TO Option	126
The GIVING Option	128
The SUBTRACT Statement	129
The FROM Option	129
The GIVING Option	131
The MULTIPLY Statement	133
The BY Option	133
The GIVING Option	134
The DIVIDE Statement	135
The INTO Option	135
The GIVING Option with INTO or BY	136
The REMAINDER Option	138
Arithmetic Statement Options of ROUNDED and ON SIZE ERROR	139
The ROUNDED Option	139
The ON SIZE ERROR Option	140
The COMPUTE STATEMENT	141
Simple Conditional Statement	142
Program Example	143
Statement of the Problem	143
Input	145
Output	145
Structure Chart	146
Program	146
Discussion	150
Programming Style	151
Common Errors	151
Exercises	152
Problems	153

Chapter 9 *Logical Sequence and Control***156**

Terms and Statements	156
Collating Sequence	157
Comparisons	157
Numeric	157
Nonnumeric	158
The IF Statement	158
Relational	161
Conditional	161
Sign	165
Class	166
Nested IF Statements	167
IF/ELSE	167
Alternative to Nested IF Statements	172
Compound IF Statements	175

Implied IF Statements	178
Subject	178
Operator	178
Program Example	179
Statement of the Problem	179
Input	179
Output	179
Structure Chart	179
Program	180
Discussion	183
Programming Style	184
Common Errors	184
Exercises	185
Problems	186

Chapter 10	<i>Validation of Input Data</i>	189
-------------------	----------------------------------------	------------

Terms and Statements	189
Types of Data Validation	190
Required Fields (Existence Test)	190
Justification Tests	190
The REDEFINES Clause	191
Numeric or Alphabetic Tests	192
Inspection	193
Sign Tests	194
Limit and Reasonableness Tests	195
Code Checking	195
Date Checking	196
Consistency Tests	196
InterFile Tests	196
Program Example	198
Statement of the Problem	198
Input	199
Structure Chart	199
Program	206
Discussion	212
Program Validation—Interactive Applications	213
Programming Style	213
Common Errors	213
Exercises	217
Problems	219

Chapter 11	<i>Control Break Processing</i>	223
-------------------	----------------------------------------	------------

Terms and Statements	223
One Level Control Breaks	224
Input	224
Output	225
Program Operations	225
Structure Chart	225
Pseudocode	227
Program	227
Discussion	230

Two Level Control Breaks—Program Modification	232
Program Operations	232
Structure Chart	233
Pseudocode	236
Program	237
Discussion	240
Multiple Control Breaks—Program Modification	243
Program Operations	243
Structure Chart	244
Pseudocode	245
Program	245
Discussion	249
Module Deletion—Changing Requirements	255
Pseudocode	255
Modification	257
Program	258
Detail and Summary Reporting	263
Programming Style	263
Common Errors	263
Exercises	264
Problems	264

Chapter 12 *Sorting Files*

267

Terms and Statements	267
Sorting Concepts	268
Sorting Alternatives	268
Main Storage during Use of SORT Statement	269
Use of the SORT Statement	269
Sort Components	269
Options of the SORT Statement	272
A More Complex Example	283
Common Errors	288
Exercises	289
Problems	290

Chapter 13 *Tables*

292

Terms and Statements	292
Table Concepts	293
Need for Tables	293
Definition and Creation of Tables	294
Table Access	295
Table Concepts—Single Level Tables	295
Creating	295
Accessing	298
Searching	300
Combining Two Tables into One	301
Table Concepts—Two Level Tables	301
Creating	304
Accessing	308
Searching	308

Program Example	311
Statement of the Problem	311
Input	313
Output	313
Structure Chart	313
Program	316
Discussion	320
Programming Style	322
Common Errors	322
Exercises	322
Problems	324

Chapter 14 *Tape and Disk Processing*

328

Terms and Statements	328
Physical Structure	329
Storage Capacity of Magnetic Tape	329
Storage Capacity of Magnetic Disk	329
File Organization	330
Sequential File Creation	330
Program Example	331
Program	332
Indexed-Sequential File Creation	335
The SELECT Statement	335
Program Example	337
Program	337
Indexed-Sequential File Reporting	341
Random Reporting	341
Sequential Reporting	341
Indexed-Sequential File Maintenance	349
Adds, Deletes, and Changes	350
COBOL Statements Necessary for Random Updating	350
Program Example	352
Program	353
Transaction Validation	358
Programming Style	359
Common Errors	360
Exercises	360
Problems	361

Chapter 15 *Additional Topics*

363

Terms and Statements	363
ENVIRONMENT Division	364
SPECIAL-NAMES Paragraph	364
DATA Division	367
77 Level Number	367
USAGE Clause	367
LINAGE Clause	370
Multiple Type of Records for a File	372
The COPY Statement	372

PROCEDURE Division	373
The ACCEPT and DISPLAY Statements	373
The WRITE Statement	375
The GO TO Statement	377
The PERFORM Statement	377
The GO TO/DEPENDING ON Statement	379
Qualification of Data-Names	380
CORRESPONDING (MOVE, ADD, SUBTRACT)	380
Common Errors	382
Exercises	383

Appendix A	<i>COBOL Language Formats</i>	385
------------	-------------------------------	------------

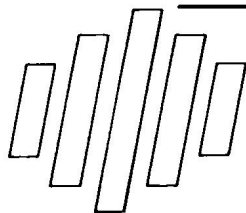
Appendix B	<i>COBOL Reserved Words</i>	402
------------	-----------------------------	------------

Appendix C	<i>Data Sets for Use with Textbook Problems</i>	404
------------	-------------------------------------------------	------------

Appendix D	<i>Answers to Selected Exercises</i>	418
------------	--------------------------------------	------------

Appendix E	<i>Text Editors</i>	426
------------	---------------------	------------

	<i>Index</i>	435
--	--------------	------------



Introduction to Programming

COBOL is an industry language rather than one developed by a specific computer vendor. It has gone through many revisions, with the industry primarily using the standards published by the American National Standards Institute (ANSI) in 1968 and 1974. Though revisions are currently under consideration, it may be some time before another version becomes standard. This text assumes the use of the 1974 standards.

COBOL is a file processing language that was developed and has been used primarily as a batch processing language. However, it is also adaptable for interactive applications.

TERMS AND STATEMENTS

Throughout the text many new terms and COBOL statements are introduced. It is a helpful reminder to list all of these terms and statements so that the student can see at a glance important new concepts that are covered. These terms are boldfaced when they are explained. The following terms are introduced in this chapter:

ANSI	JCL	Program
Arithmetic/Logic unit	Logic errors	documentation
ASCII	Magnetic disk	Punched cards
Auxiliary storage	Magnetic tape	Random processing
Batch processing	Main storage	Sequential processing
COBOL	Numeric bits	Solution planning
CODASYL	Object program	Source program
Compiler	Online processing	Syntax errors
Control section	Output devices	Temporary storage
CPU	Parity bit	Terminal
Data	Permanent storage	Testing and debugging
Documentation	Primary storage	Zone bits
EBCDIC	Problem definition	
Input devices	Programs	
Interactive processing	Program coding	
	Program debugging and testing	

COBOL History

COBOL is a unique computer language in that it was developed by the computer industry rather than by any one computer vendor. The beginnings of COBOL can be traced to the U.S. Department of Defense. The defense department was having difficulties working with the many different computers and computer languages purchased and used by their defense contractors. Because of the need for each contractor to provide contract costs, progress of the contract, and so forth, the defense department was perplexed by the monumental task of auditing the results from each defense contractor.

The Committee on Data Systems Languages (**CODASYL**) was created in 1959 to consider the possible development of a single computer language that would be useful for administrative applications. The defense department reasoned that if all defense contractors were using the same computer language, then the problem's magnitude would be lessened. CODASYL was comprised of computer users, defense department representatives, computer manufacturers, and others. The results were released in April 1960 and became known as **COBOL** (Common Business Oriented Language). COBOL was developed to be (1) standardized, (2) self-documenting, and (3) useful primarily for administrative applications.

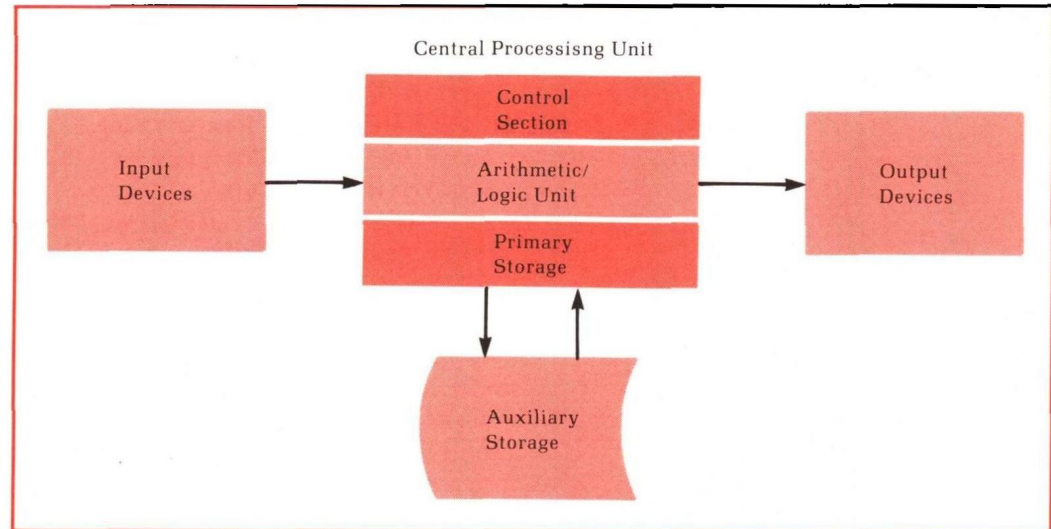
Revisions were made in 1963 and 1965 by CODASYL, and numerous revisions were made by individual computer vendors with concomitant decreases in standardization. In 1968 the American National Standards Institute (**ANSI**) prepared suggestions for better standardization. Current ANSI standard COBOL compilers use either the 1968 or the more current 1974 ANSI standards. ANSI is currently considering a new COBOL standard, but has not yet obtained agreement within the computer community as to what the new standards should be.

COBOL has become an extremely important language for administrative applications. There are literally millions of COBOL programs being used and continually modified. COBOL programming skills are expected to be needed well into the future because of the investment that firms have made in existing COBOL code. Notwithstanding what one might hear about database systems and fourth-generation computer languages, large and medium-size companies continue to recruit programmers who are competent in COBOL. Most microcomputer vendors now offer COBOL compilers, and the industry is seeing more and more application packages that are available in COBOL on smaller and smaller computers. COBOL is not dead; it seems to be thriving on an idea that is now over a quarter of a century old. Those who intend to make a career of designing administrative computer systems should make every effort to learn COBOL, and learn it well. It may be their ticket to a successful career in information processing.

Hardware Concepts

No matter what kind of computer or what kinds of programming languages are being used, the programmer must consider some hardware and software concepts. The understanding of these concepts is essential in getting the computer to accomplish a task. The concepts are all relatively simple, but necessary to understand before the user begins to look at the COBOL programming language.

In any computer system, from the smallest personal computer to the largest mainframe computer, there are four hardware components in almost any given application. These components are depicted in Figure 1-1. The four major components are **input devices**, the central processing unit (**CPU**), **output devices**, and **auxiliary or permanent storage devices**. The purpose of any input device is to send **data** and/or programs to the CPU to be processed or stored in the computer system. Similarly, the output devices receive the data and/or programs sent to them from the CPU. **Auxiliary storage** devices accept data from the CPU for more or less permanent storage of data and/or programs. All auxiliary storage devices are actu-

Figure 1-1 Hardware Components of a Computer System

ally both input and output devices, because data can be stored on or retrieved from these devices.

Central Processing Unit

The several functions of the CPU are served by three major units, the **control section**, the **arithmetic/logic unit**, and **primary or main storage**.

Control Section

The control section acts as the supervisor for the entire computer system, directing the activities of the CPU and the activities of all input and output devices, including auxiliary storage. Examples of activities controlled are the movement of data and/or programs from an input device into memory, movement of data and/or programs between auxiliary storage and primary storage, and movement of data to an output device.

Arithmetic/Logic Unit

The arithmetic/logic unit performs all mathematical functions, such as multiplication and rounding; it also gives the computer its logic capabilities, such as comparing two different values, checking the validity of the data as they are being transferred from one location to another, or allowing a program to be executed in a different sequence, enabling the programmer to change the way the program operates.

Primary Storage

Primary storage, also called memory, allows the storage of data and programs that can be immediately accessed by the computer system. Primary storage is considered temporary because of the volatile media used to store data and programs in the CPU. Volatility means that the stored data gets lost unless stored on a more permanent device. To illustrate this point, consider what happens when a hand-held calculator is turned off; all values, except those placed in a memory register, are lost. Primary storage is the same; if the computer is turned off, all data is lost. Additionally, other programs need the space as soon as one program has completed its tasks. A subsequent program overlays previously used storage locations in memory when it needs to use the locations. This overlay process has considerable importance later as the COBOL statements that manipulate data in primary storage are developed.