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

1986
IRWIN
Homewood, Illinois 60430

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.

Steve Teglovic, Jr.
Kenneth D. Douglas

CONTENTS

1

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

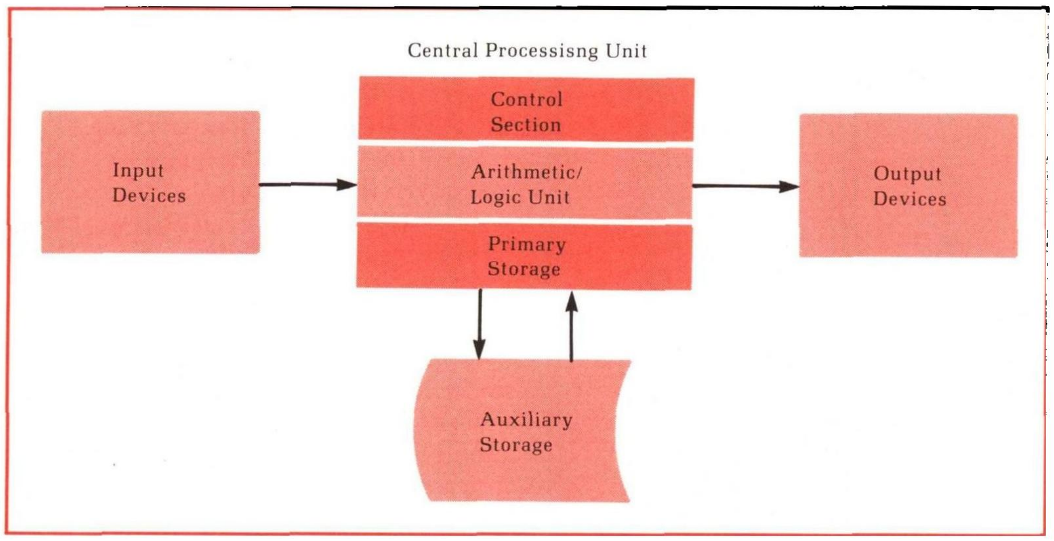
COBOL has become an extremely important language for administrative applications. There are literally millions of COBOL programs being used and continually modified. COBOL programming skills are expected to be needed well into the future because of the investment that firms have made in existing COBOL 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.