John A. Bonno

Introduction to Cobol

Introduction to COBOL

John A. Bonno
Texas A & I University

Kent T. Fields
Louisiana State University

1982

BUSINESS PUBLICATIONS, INC. Plano, Texas 75075
Irwin-Dorsey Limited Georgetown, Ontario L7G 4B3

ISBN 0-256-02287-9
Library of Congress Catalog Card No. 80–70654

Printed in the United States of America

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

Preface

This text is intended for use in a one- or two-term course in COBOL. It introduces the COBOL language as outlined in the American National Standards Institute 1974 and 1968 Standards.

The approach of this book is to present structured techniques in the writing of COBOL programs. In addition, we have attempted to introduce all of the features that are available in both the 1968 and 1974 versions of the language. Where these versions differ, we have clearly marked 1968 or 1974 in the margin of the text. Some features are available in the 1968 version that are not available in the 1974 version. The organization of the text is such that the instructor may either take the chapters in sequence as they appear or may rearrange the introduction of the material after Chapter 6. In general we believe that the introduction of enough basics to allow the student to begin coding and have an understanding of the coding process is covered in the first six chapters. While we would like to see students coding earlier than the completion of Chapter 6, it has been our experience that students engage in the coding process without understanding what is going on. Consequently, more material is presented in these early chapters than will be seen in the usual COBOL text.

At the end of each chapter we have presented illustrative programs. Although the students may not be able to code their own programs for the first five or six chapters, they may begin executing the programs that appear at the end of the chapters. This will give an idea of how COBOL programs work. All the programs at the end of the chapters have been tested.

The chapters that make up this text have been classroom tested over a series of semesters at the universities where the authors teach. The questions appearing at the end of each chapter are those that have most often arisen in discussions after the presentation of the material.

We would like to express our appreciation to Jerry Hanys of the El Paso Natural Gas Company, Ron Gillet of the Coastal Corporation, and

David Lamoreaux of PPG Industries for their assistance in classroom testing and program testing of the materials in this book.

Your comments, criticisms, and suggestions as to the contents of the text are not only appreciated but encouraged. Please write to us if you have comments to make about the contents of the work.

Acknowledgment

The following acknowledgment is made at the request of the American National Standards Institute, Inc. From ANSI X3.23-1974, American National Standard: Programming Language COBOL (1974).

COBOL is an industry language and is not the property 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.

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.

John A. Bonno
Kent T. Fields

Contents

part I · Basic concepts

chapter 1 • Introduction to COBOL

The word COBOL stands for COmmon Business Oriented Language. COBOL is one of many languages used to communicate with a computer in order to accomplish a task, particularly a business-oriented task. Although the differences between scientific and business applications are sometimes few, business problems usually have a significant amount of input (information to be analyzed) and a volume of reports which will be printed or maintained on various storage devices. It is the application to business problems, the ease of reading a COBOL program written in English-like sentences, and the wide acceptance of the language in business and government that distinguish COBOL from other languages.

section A • History, acceptance, and versatility

Prior to 1959 each manufacturer of computers designed programming languages for use on each of their machines or for use by a series of machines produced by them. In many cases, other manufacturers would be sufficiently impressed by the characteristics of a particular language designed by a competitor to adopt parts of the language or to design a similar language for use with their own machines. This proliferation of noncompatible programming languages was a source of considerable concern to large-scale computer users who owned or leased equipment produced by two or more manufacturers. Not only would programs written for one machine fail to execute on another machine of different manufacture, but also the exchange of programs and the exchange of programming experience and skilled personnel was drastically limited by the nonstandardization of languages. Early in 1959 the largest of the computer users, the U.S. government, became concerned enough to initiate some action. At a meeting called by the secretary of defense, a committee was formed to investigate the problem of establishing a standard language for use in business and government applications. This committee, made up of representatives from business, government, and computer manufacturing, was called CODASYL, an acronym for the COmmittee on DAta SYstem Languages. A subcommittee was formed to establish a COmmon Business Oriented Language, and the acronym for this committee name serves to identify both the committee and the language which it designed—COBOL.

The first COBOL language described by the committee was designed to provide an initial language for temporary use by government and industry pending design of compilers for the basic language by each of the manufacturers. This development work was several years in process, so that several versions of the "temporary" COBOL were released. These releases (by the COBOL committee) were known as COBOL–60, COBOL–61, COBOL–61 (extended), and COBOL–63.

From 1961 through 1968 the manufacturers released their own COBOL versions (for example, IBM 360 LEVEL E COBOL, superceded by LEVEL F, and so on). These versions were all within the general framework outlined by CODASYL. With the passage of time, however, they were becoming less and less standard.

In 1968 a new "standard" COBOL was introduced. It was called United States of America Standards Institute (USASI) COBOL, with a name change in 1969 to American National Standards Institute (ANSI) COBOL. All major manufacturers now support the ANSI version and, for the most part, have abandoned support of their private versions. Yet, before ANSI COBOL (1968) was off the presses, various computer manufacturers had already added extra items to the language. So in addition to the basic statements that are in ANSI COBOL, each manufacturer has added COBOL verbs (commands or procedural instructions to the computer), allowed abbreviations to be used that are not part of ANSI COBOL, added new types of PICTURES that are allowed to describe data items; and other changes. This is not bad in itself because the additions usually make the language more flexible, but the additions also prevent a "standard" ANSI COBOL program being run on any manufacturer's equipment. So long as the programmer sticks to basic COBOL statements, interface (transferability) with competitive equipment is maximized. This allows a company to have the ability to change to another company's computer equipment and still use the COBOL programs that have been written without the inconvenience of converting or rewriting programs from one manufacturer's COBOL to another's.

The ANSI STANDARD COBOL (1968) was on a five-year revision cycle. In May 1974 (a year late) the ANSI COBOL standard X3.23–1974 version was adopted as a national standard. Shortly thereafter this standard was adopted by the federal government. A copy of the standards can be obtained by writing to:

American National Standards Institute
1430 Broadway
New York, New York 10028

As with most new things, it takes time to get the ideas into actual usage. The various manufacturers must incorporate the new facets of the language for their specific hardware. Since both the 1968 version and the 1974 version are currently in use, this text deals with both versions of COBOL.

The American National Standards Institute lists 189 "substantive changes" to the 1968 COBOL. However, most of these do not affect the basic structure of COBOL. Anyone who can write the 1968 version would have little difficulty in converting to the 1974 standards and vice versa.

Most of the changes to the 1968 version of COBOL do not affect programming techniques very much. A few of the changes, however, require specific alterations of former programs if they are to be run (compiled) with the 1974 version of COBOL.

For example, the entry REMARKS is allowed in the 1968 version but not allowed in the 1974 version. In this case, both the REMARKS entry and its replacement (an asterisk) will be covered in this text. The text regarding the 1968 version will have side borders indicating 1968–1968–1968–1968, meaning this applies only to 1968 COBOL. The text regarding the 1974 version will have side borders indicating 1974–1974–1974–1974, meaning this applies only to 1974 COBOL. If the material is applicable to both, no side borders will be used.

1968–1968

Material that is enclosed inside these side borders is specifically related to the 1968 version of COBOL and not valid in the 1974 version of ANSI COBOL.

1968–1968

1974–1974

Material that is enclosed inside these side borders is specifically related to the 1974 version of COBOL and not valid in the 1968 version of ANSI COBOL.

1974–1974

In this book, an attempt has been made to write ANSI COBOL, or COBOL that will run on most computer equipment. On occasion, statements that are peculiar to various manufacturer's equipment will be discussed and separated from the rest of the COBOL statements. Burrough's

Corporation, Control Data Corporation (CDC), Digital Equipment Corporation (DEC), International Business Machines (IBM), National Cash Register (NCR), and Sperry Rand COBOL are considered in this book.

section B • An overview

The purpose of this book is to help you write COBOL programs. A COBOL *program is a series of instructions written in English-like* sentences.

The COBOL program is then run on a computer (computer system). The computer system is composed of a series of physical devices referred to as *hardware*. The separate pieces of equipment can function on their own, that is, a *card reader* can read cards. However, this is of little consequence if nothing can be done with the information. It is not enough that card readers can read cards and printers can print information on paper. The computer system must be given some intelligence so that it will read cards at the appropriate time, manipulate data from the proper locations in memory, and write the desired information. The computer hardware is given instructions (called *software*) which control such activities. These instructions (intelligence) are generally called an *operating system.*

The operating system is capable of handling COBOL programs but not by itself. You will submit a COBOL program to the computer system (the hardware) through an operating system. The operating system, however, does not understand the COBOL instructions you have written. There must be something that can convert the COBOL commands (instructions) into machine language (or machine executable instructions).

There must be additional intelligence provided to the computer so that the COBOL program can be translated from the English-like sentences into machine language. This intelligence is called a *compiler* (which is also software).

When you submit a COBOL program for the computer to run, you must tell the operating system certain facts about your program. This would include the fact that it is a COBOL program. When the operating system knows that it has a COBOL program to run, it will first use the COBOL compiler to translate the COBOL commands into machine language. It then has something which the computer (hardware) can use.

In order to communicate with the operating system, you will have to learn a few parts of another language often called Job Control Language (JCL).[1] It is the JCL that tells the operating system facts such as who you are, who should be charged with the cost of running the program, the type of language your program is using, and other information.

[1] Some systems use different terminology. For example, the UNIVAC system uses RCL (Run Control Language).

For each computer the JCL is different. Some examples of JCL are given later in this chapter. Before you can run your first program, either the computer center or your instructor will have to tell you what JCL is required for your computer system.

In summary then you have:

1. A computer system (the hardware)
2. The operating system
3. JCL (to communicate with the operating system)
4. The COBOL compiler
5. The COBOL program

$\left.\begin{matrix} 2. \\ 3. \\ 4. \\ 5. \end{matrix}\right\}$ Software

These are discussed separately in the following sections.

section C • The COBOL compiler

The computer is the electronic machine that will handle your program. This machine works in its own language, which can be electronically understood. COBOL is written in English-like words and sentences and must be translated into machine language for the computer to perform according to the instructions in your program. The translator (translator program) is referred to as a *compiler*. The compiler is activated by the computer system and it translates your program from COBOL into machine language. The compiler also provides for program listings (list of the statements in your program) and error messages (the best estimate of what is wrong with your program). These error messages are also commonly referred to as *diagnostics*.

The compiler is generally provided by the manufacturer of the computer equipment so that your COBOL program will run on that equipment. Therefore, the COBOL commands that you write must fit the compiler. Although most of the COBOL commands are the same for all manufacturers' COBOL compilers, there are nonetheless some differences. As you are shown the particular COBOL commands, the areas of difference will be pointed out.

Compile Time is the amount of time required by the COBOL compiler to translate your statement to machine executable instructions, usually measured in seconds.

section D • The computer system and the operating system

When you attempt to run a program there will be a significant number of things happen that do not directly concern you. The computer system (hardware) being directed by an operating system will take the COBOL program and translate the COBOL commands with the use of the COBOL compiler. Your COBOL instructions will then be able to communicate with the operating system so that you may use the card reader, the printer, and any other equipment necessary for your program.

When you (or the computer center operator) press the READ button on the card reader to feed the information from the cards into the computer, it is not your COBOL program that is directing the computer to read the deck. It is the operating system. The operating system takes the COBOL commands and processes them. When the COBOL compiler is finished, the operating system has a set of machine executable instructions (generated by the COBOL compiler) that it can use.

The machine executable instructions furnished by the COBOL compiler is often called an *object program* (or, if in card form, an *object deck*). After the compiler is finished, the computer system will allow the object (COBOL) program to run. By run we mean to start processing the instructions. This is often called the *execute step* (stage) or the GO step. In the GO step the object (COBOL) program can cause information to be brought in from the card reader (or other input device) or have information transferred to the printer (or other output device).

Execute time is the amount of time used by the object program to carry out the instructions of the COBOL program. Execute time is generally composed of input time, CPU time, and output time.

Input time is the amount of time necessary to bring information into the computer (input). CPU time (central processing unit time) is the amount of time required for the CPU to carry out its functions of arithmetic and/or data manipulations. Output time is the amount of time used to get the desired information out of the computer (output).

The amount of time used in the compiler step is a function of the complexity of the program. The amount of time used in the GO (or execution) step is a function of the amount of input and/or output you want.

Depending on the efficiency of the operating system you are using, it is possible for the CPU to generate several thousand lines of output for the printer in a matter of seconds.

Figure 1–1 illustrates what happens when a COBOL program is submitted to a computer system to be compiled and run.

It is not necessary for you to know how an operating system works in order to run your COBOL program, but you must know how to communicate with the operating system. This is done through the use of a Job Control Language (JCL). This JCL will be different for every operating system, although in many cases the differences may be minor. To get an idea of the great number of computer systems that exist, consider the IBM 360/370 series of computers. IBM is one of a half-dozen major computer manufacturers and the 360/370 series only one of several computer series that they either manufacture or have previously manufactured. Yet there are four different operating systems that are designed for the 360/370 alone. These are disk operating system (DOS 360/370),