

COBOL PROGRAMMING

a structured approach

Peter Abel

```
00054  
00055  
00056  
00057  
00058  
00059  
00060  
00061  
00062  
00063  
00064  
00065  
00066  
00067  
00068  
00069  
00070  
00072  
00073  
00074  
00075  
00076  
00077  
00078  
00079  
00080  
00081  
00083  
00084  
00085  
00086  
00087  
00088  
00090  
00091  
00092  
00093  
00094  
00095  
00097  
00098
```

```
PROCEDURE DIVISION.  
A000-MAIN-LOGIC SECTION.  
  OPEN INPUT MAST-FILE-IN  
    OUTPUT TRAN-FILE-IN  
  READ MAST-FILE-IN  
    AT END MOVE HIGH-VALUES TO MAST-CTL-IN.  
  READ TRAN-FILE-IN  
    AT END MOVE HIGH-VALUES TO TRAN-CTL-IN.  
  PERFORM C000-SELECT.  
  PERFORM B000-PROCESS UNTIL PROCESSING-COMPLETE.  
  DISPLAY 'END OF UPDATE'.  
  CLOSE MAST-FILE-IN  
    TRAN-FILE-IN  
    MAST-FILE-OUT.  
  STOP RUN.  
A999-END. EXIT.  
  
B000-PROCESS SECTION.  
  PERFORM D000-MAST.  
  IF TRAN-CTL-IN = LOW-CTL-KEY  
    UNTIL TRAN-CTL-IN NOT = LOW-CTL-KEY.  
  MOVE MAST-BALANCE TO MAST-AMT-OUT.  
  WRITE MAST-REC-OUT  
    INVALID KEY DISPLAY MAST-CTL-OUT, 'INVALID WRITE'.  
  PERFORM C000-SELECT.  
B999-RETURN. EXIT.  
  
C000-SELECT SECTION.  
  IF TRAN-CTL-IN < MAST-CTL-IN  
    PERFORM F000-UNMATCHED  
    UNTIL TRAN-CTL-IN NOT < MAST-CTL-IN.  
  MOVE MAST-CTL-IN TO LOW-CTL-KEY.  
C999-RETURN. EXIT.  
  
D000-MAST SECTION.  
  MOVE MAST-AMT-IN TO MAST-BALANCE.  
  MOVE MAST-REC-IN TO MAST-REC-OUT.  
  READ MAST-FILE-IN  
    AT END MOVE HIGH-  
D999-RETURN. EXIT.
```

A complete guide to COBOL. Explores DOS and OS usages, gives advanced debugging techniques, and more!

COBOL PROGRAMMING

A STRUCTURED APPROACH

PETER ABEL

British Columbia Institute of Technology



Reston Publishing Company, Inc.
Reston, Virginia
A Prentice-Hall Company

Library of Congress Cataloging in Publication Data

Abel, Peter

Cobol programming, a structured approach.

Includes index.

1. COBOL (Computer program language) I. Title.

QA76.73.C25A23 001.64'24 80-10488

ISBN 0-8359-0833-X

© 1980 by Reston Publishing Company, Inc.
A Prentice-Hall Company
Reston, Virginia 22090

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

10 9 8 7 6 5 4 3 2

Printed in the United States of America

PREFACE

Programming in COBOL: A Structured Approach offers a new and better approach to COBOL, the most commonly used business programming language. The most elementary features of COBOL are introduced first with a simple example program. After only one chapter of COBOL (Chapter 2), the student is able to write a simple program by concentrating on only the statements necessary to write such a program and by temporarily using the simple ACCEPT and DISPLAY statements for input/output. The text adds new concepts gradually with program examples, and consequently the student is never overwhelmed by the topic.

Structured programming is introduced early. Even before formally discussing the topic, the text adopts and illustrates structured programming features: indentation of code, meaningful names, and organization into main logic and subsidiary sections. By the time Chapter 7 describes structured programming, the student is already accustomed to its features.

A problem in current COBOL texts is omission of specific topics. Thus, one book omits programming style and strategy, another omits control break logic, and another omits advanced handling of tables or disk files. Ideally, one text should cover all of these essential topics, and *COBOL Programming: A Structured Approach* does just this.

Although COBOL is intended to be a universal language, there are two main reasons for the variety of COBOL versions:

1. Differences in computer design (hardware).
2. Differences in operating systems (software).

How does a textbook handle this situation? Because the IBM system is the most common computer system used in the data processing industry, the programs in this text have been designed with IBM features. However, the text points out features that are unique to IBM and that deviate from ANS standards. Your compiler may have some differences from the examples in this text in the following areas:

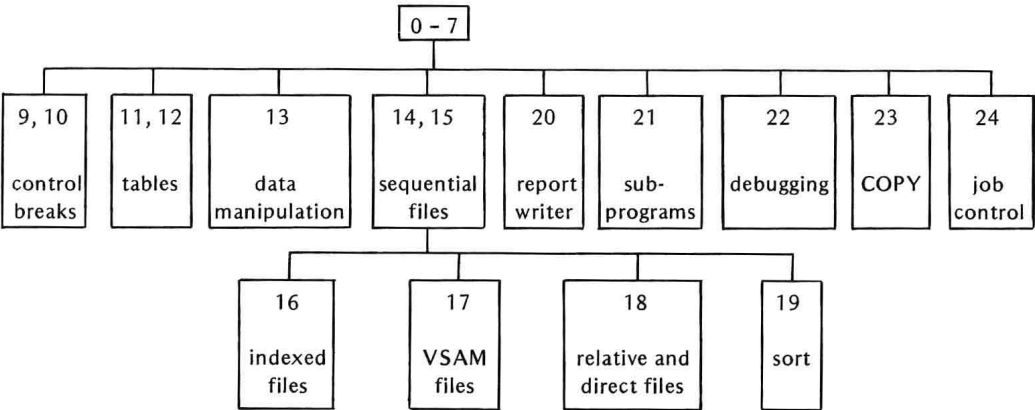
- 1. Debugging aids.
- 2. SELECT statements that define input and output files.
- 3. Definition of arithmetic data in Working-Storage.
- 4. Specifications for page overflow.
- 5. Disk devices and disk file organization methods.

These areas represent a small portion of a COBOL program; consequently, students should be able to easily adapt the material in this text to their compiler version. Students should also use the manufacturer's COBOL reference manual along with this text.

In the text, a reference to the IBM 370 or 360/370 implies the general IBM computer series based on the 360. This includes the 3000 series released in 1977 and the 4300 series released in 1979.

ORGANIZATION

Chapters 0 and 1 contain elementary material on computers and programming. Those familiar with another language can skip these chapters or use them as a review. Chapters 2 through 5 cover the basics of COBOL, representing 80 percent or more of standard program coding. Chapters 6 and 7 are concerned with programming style and the use of PERFORM in structured programming. From this point on, the chapters can be studied in various sequences. (Chapter 8, Report Design, is independent of COBOL programming and may be referenced at any time.) The following chart indicates the relationships of the chapters. For example, Chapters 9 and 10 in a box means that Chapter 9 should be read before Chapter 10.



NOTATIONS USED IN THIS TEXT

Some COBOL statements allow a number of options; the text shows their general format according to the following pattern:

Uppercase words	Words in uppercase, such as SPACES and READ, are COBOL reserved words. They must be spelled correctly and used in a program only according to COBOL specifications.
Lowercase words	Words in lowercase are generic terms that represent words or symbols that the programmer supplies.
Brackets []	Brackets indicate that the enclosed entries are optional and may be included or omitted.
Braces { }	Braces enclose two or more vertically stacked items, one of which the programmer is to select.
Ellipses ...	Ellipses immediately following a pair of brackets or braces indicate that the programmer can optionally repeat the enclosed material.

For example:

ADD	{	identifier-1	}	[{	identifier-2	}]	...	identifier-3	[ROUNDED]	.
↑		↑			↑	↑			↑			↑		
<i>required</i>		<i>choice</i>		<i>optional</i>	<i>and choice</i>			<i>can repeat</i>				<i>optional</i>		
								<i>previous entry</i>						

ACKNOWLEDGEMENT

The author is grateful for the assistance from all those who contributed typing, reviews, and suggestions and to IBM for permission to reproduce some of their copyrighted materials. The following materials are printed with permission, and with modifications, from publications copyrighted in 1972, 1973, 1975, and 1978 by International Business Machines Corporation as IBM form numbers GC20-1649, GC20-1684, GC28-6396, and GA33-1515: Figures 0-3, 8-1, 14-1, 14-7, Appendix B, and general formats of COBOL statements.

PETER ABEL

COBOL PROGRAMMING

CONTENTS

PREFACE	xiii
---------	------

PART I COMPUTER FUNDAMENTALS

0	INTRODUCTION TO COMPUTERS	3
	Introduction, 3	
	Binary Number System, 4	
	Bits, 5	
	Computer System, 6	
	Punched Cards, 10	
	Data Entry Devices, 12	
	Operating System, 14	
1	INTRODUCTION TO PROGRAMMING	16
	Introduction, 16	
	Business Applications, 16	
	Input Data, 17	
	Approach to Programming, 21	
	Program Translation, 22	
	Flowcharting, 23	

PART II BASIC COBOL

2	INTRODUCTION TO COBOL	33
	Introduction, 33	
	COBOL Coding Sheet, 34	
	Special COBOL Features, 36	
	IDENTIFICATION DIVISION, 38	
	ENVIRONMENT DIVISION, 39	
	DATA DIVISION, 40	
	PROCEDURE DIVISION, 42	
	Defining Fields, 43	
	Cross-Reference Table, 46	
	COBOL Organization, 47	
	Writing a COBOL Program, 48	
	Debugging, 48	
3	ELEMENTS OF COBOL	55
	Introduction, 55	
	Repetitive Processing, 55	
	Numeric Input Data, 59	
	Printed Arithmetic Fields, 60	
	Program Example, 61	
	Level Numbers, 61	
	Independent Data Items—Level 77, 63	
	Alphabetic Data—Picture A, 63	
	Alphanumeric Data—Picture X, 64	
	Arithmetic Data in Working-Storage, 66	
	MOVE Statement, 68	
	Program Example, 69	
	Debugging, 71	
4	ARITHMETIC AND LOGIC	74
	Introduction, 74	
	Arithmetic, 74	
	SIZE Errors, 79	
	COMPUTE Statement, 79	
	Editing Arithmetic Data, 81	
	The IF Statement, 85	
	Program Example, 87	
	Figurative Constants, 91	
	Error Prevention, 92	

5	FULL INPUT/OUTPUT	96
	Introduction, 96	
	INPUT-OUTPUT Section, 99	
	FILE Section, 100	
	OPEN Statement, 101	
	CLOSE Statement, 101	
	READ Statement, 102	
	WRITE Statement, 103	
	Buffers, 105	
	Error Prevention, 106	
 PART III PROGRAM CONTROL AND LOGIC		
6	LOGIC TESTS AND PERFORM	111
	Introduction, 111	
	Condition Names—Level 88, 111	
	The PERFORM Statement, 113	
	Program Example, 116	
	Page Overflow, 117	
	Advanced IF Logic, 121	
	The GO TO Statement, 124	
	GO TO DEPENDING, 125	
	Date, 125	
	Error Prevention, 126	
7	PROGRAMMING STRATEGY AND STYLE	128
	Introduction, 128	
	Structured Programming, 128	
	Program Control, 131	
	Programming Objectives, 132	
	COBOL Programming Practices, 133	
	Program Efficiency, 140	
	Program Example, 141	
8	REPORT DESIGN	148
	Introduction, 148	
	Computer Printouts, 148	
	Principles of Good Report Design, 150	
	Design Features, 152	
	Printer Layout, 157	
	Control Register, 158	

9	CONTROL BREAK LOGIC—I	161
	Introduction, 161	
	Sequence Checking, 162	
	File Processing and Control Breaks, 162	
	Program Example, 167	
	Error Testing, 169	
	Totals Only, 173	
	Error Prevention, 177	
10	CONTROL BREAK LOGIC—II	178
	Introduction, 178	
	Two-Level Control Breaks, 178	
	Program Example, 181	
	Three-Level Control Breaks, 186	
	Page Overflow, 192	
	Programming Strategy, 193	
 PART IV ADVANCED TOPICS		
11	TABLE PROCESSING—I	199
	Introduction, 199	
	Defining Tables, 199	
	Sequential Tables, 201	
	Table Searching, 206	
	Indexes, 207	
	PERFORM VARYING Statement, 208	
	SEARCH Statement, 209	
	Program Example, 210	
	Related Tables, 215	
	Two- and Three-Dimensional Tables, 215	
	Rules for Subscripting and Indexing, 217	
	Debugging Tips, 218	
12	TABLE PROCESSING—II	222
	Introduction, 222	
	Loading Tables, 222	
	Program Example, 225	
	Sorting Table Entries, 230	
	Binary Search, 232	
	Calculation of Days Between Dates, 237	
	Decimal Accumulation, 239	

13	DATA MANIPULATION	242
	Introduction, 242	
	Qualified Names, 242	
	CORRESPONDING Operations, 244	
	JUSTIFY RIGHT, 246	
	REDEFINES, 246	
	RENAMES—Level 66, 247	
	EXAMINE, INSPECT, and TRANSFORM, 249	
	STRING AND UNSTRING, 256	
PART V	FILE ORGANIZATION METHODS	
14	TAPE AND DISK SEQUENTIAL FILES	263
	Introduction, 263	
	Magnetic Tape, 264	
	Program Example—Creating a Tape File, 267	
	Disk Storage, 270	
	Program Example—Creating a Disk File, 272	
	Variable Length Records, 274	
15	FILE MERGING AND UPDATING	278
	Introduction, 278	
	Data Processing Systems, 278	
	Multiple Input Devices and Files, 281	
16	INDEXED SEQUENTIAL FILE ORGANIZATION	294
	Introduction, 294	
	Indexed Sequential Characteristics, 295	
	Processing an Indexed File, 296	
	Programming an ISAM File, 298	
	Program Example, 302	
17	VIRTUAL STORAGE ACCESS METHOD—VSAM	306
	Introduction, 306	
	Storage of Records—Control Interval, 306	
	Key-Sequenced Data Sets, 307	
	Creating a Key-Sequenced Data Set, 309	
	Sequential Reading of a Key-Sequenced Data Set, 309	
	Random Reading of a Key-Sequenced Data Set, 311	
	Sequential Updating of a Key-Sequenced Data Set, 311	
	Random Updating of a Key-Sequenced Data Set, 311	
	Entry-Sequenced and Relative Record Data Sets, 313	

18	RELATIVE AND DIRECT FILES	315
	Introduction, 315	
	Relative Files, 315	
	Direct Files, 318	
	Summary of Disk File Organizations, 323	
19	COBOL SORT	325
	Introduction, 325	
	Sort Characteristics, 326	
	Sort Requirements for USING and GIVING, 326	
	Sort Requirements for Input and Output Procedures, 329	
	Sort Registers, 331	
PART VI	MISCELLANEOUS TOPICS	
20	THE REPORT WRITER FEATURE	337
	Introduction, 337	
	Program Example, 337	
	Report Description (RD) Entry, 342	
	Report Group Description Entries, 343	
	Report Groups, 345	
	Control Breaks, 346	
	Program Example, 347	
21	SUBPROGRAMS	352
	Introduction, 352	
	Program Linkage, 353	
	Common Data, 353	
	Entry Points, 354	
	Program Example, 356	
	Linkage Editor, 358	
22	DEBUGGING	361
	Introduction, 361	
	IBM 360/370 Debugging Aids, 361	
	CDC Debugging Aids, 363	
	ANS Debugging Standard, 364	
	Statement Option, 365	
	Storage Dumps, 366	

23	THE COPY STATEMENT	375
	Introduction, 375	
	Example Use of COPY, 375	
24	JOB CONTROL	378
	Introduction, 378	
	DOS Job Control, 378	
	OS Job Control, 383	
	Appendices	
A	KEYPUNCH PROCEDURE	389
B	LIST OF IBM ANS COBOL RESERVED WORDS	393
C	IBM 360/370 PROGRAM CHECKS	396
	INDEX	398

PART I

COMPUTER FUNDAMENTALS

CHAPTER 0

INTRODUCTION TO COMPUTERS

OBJECTIVE: *To examine the internal characteristics of the digital computer, its input/output devices, and the function of the stored program.*

INTRODUCTION

Computers, like other automated devices, were developed to replace human labor. There are two classes of computers, *analog* and *digital*. The *analog* computer measures physical variables such as rotation speed, water pressure, and electric current. Speedometers, steam pressure gauges, and barometers are good examples of analog computers. The *digital* computer works with digits to perform calculations. Unlike the analog computer that can perform only one function, the digital computer can perform many functions through the use of a *stored program*, which is a set of instructions that the computer executes. Another program can replace the stored program at any time to perform a different function. This text uses the term *computer* to mean the common digital computer.

The modern digital computer using the stored program was developed in the 1940s to provide fast, accurate calculations to solve complex problems. Since that time the computer has made its impact in two main areas: scientific and business. In the scientific area, the computer performs calculations for applications such as bridge and building designs, simulation models of the national economy, and statistical studies of the population. In the business area, the computer processes and controls large volumes of data (data processing). Common business applications include accounting, billing, sales analysis, inventory and production control, and airline reservations. In addition, government agen-