

# COBOL

## From Micro to Mainframe

Preparing for the New Millennium



ROBERT T. GRAUER  
CAROL VAZQUEZ VILLAR  
ARTHUR R. BUSS

THIRD EDITION

Micro Focus® Coverage Included



# COBOL

**From Micro to Mainframe**

**Preparing for the New Millennium**

T H I R D   E D I T I O N

**ROBERT T. GRAUER**

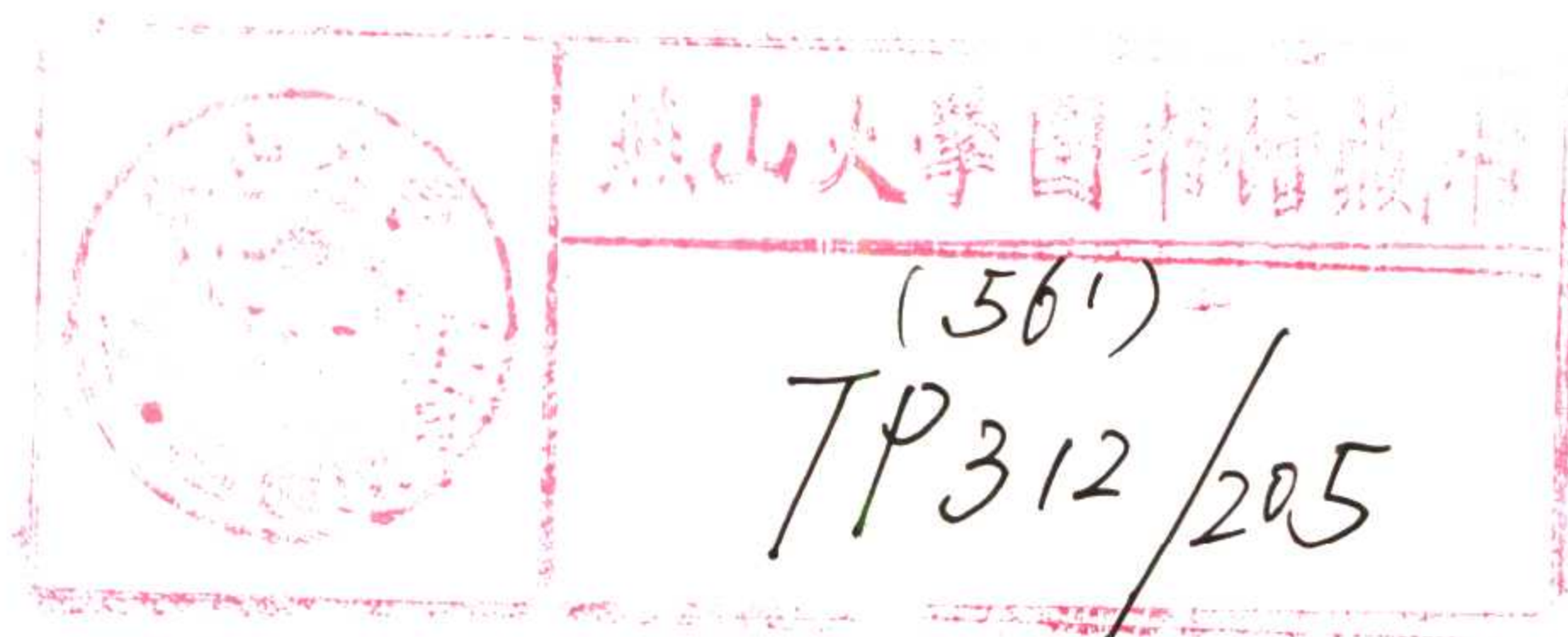
University of Miami

**CAROL VAZQUEZ VILLAR**

Andersen Consulting

**ARTHUR R. BUSS**

William Jewell College



**PRENTICE HALL**  
Upper Saddle River, NJ 07458



0316026

85



## Library of Congress Cataloging-in-Publication Data

Grauer, Robert T.

COBOL: from micro to mainframe; preparing for the new millennium, /Robert T. Grauer, Carol Vazquez Villar, Arthur R. Buss.

p. cm.

Includes index.

ISBN: 0-13-790817-2

I. COBOL (Computer program language). I. Vazquez Villar, Carol

II. Title.

CIP Data available.

## *To Marion, Benji, and Jessica To Mario, Mom & Dad, and Fefa To Pat, Betsy, and Carolyn*

Acquisitions editor: LAURA STEELE

Publisher: ALAN APT

Editor-in-chief: MARCIA HORTON

Production editor: IRWIN ZUCKER

Managing editor: BAYANI MENDOZA DE LEON

Director of production and manufacturing: DAVID W. RICCARDI

Copy editor: MARJORIE SHUSTAK

Art director: HEATHER SCOTT

Cover designer: JOHN CHRISTIANA

Cover illustration: PATRICE VAN ACKER

Manufacturing buyer: DONNA SULLIVAN

Editorial assistant: KATE KAIBNI

© 1998, 1994, 1991 by Prentice Hall, Inc.

Upper Saddle River, NJ 07458

Reprinted with corrections August, 1999.

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

The author and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

**TRADEMARK INFORMATION:** Micro Focus® Personal COBOL™ for Windows™ is a registered trademark of Micro Focus Ltd. Windows® and Windows 95® are registered trademarks of Microsoft Corporation. Windows NT™ is a trademark of Microsoft Corporation.

Printed in the United States of America

10 9 8 7 6 5

**ISBN 0-13-790817-2**

Prentice-Hall International (UK) Limited, London

Prentice-Hall of Australia Pty. Limited, Sydney

Prentice-Hall of Canada, Inc., Toronto

Prentice-Hall Hispanoamericana, S. A., Mexico

Prentice-Hall of India Private Limited, New Delhi

Prentice-Hall of Japan, Inc., Tokyo

Prentice-Hall Asia Pte. Ltd., Singapore

Editora Prentice-Hall do Brasil, Ltda., Rio de Janeiro

# Preface

---

*COBOL: From Micro to Mainframe* is a truly comprehensive work, providing in a single source all subjects normally covered in the one-year COBOL sequence. The scope is extensive, ranging from an introduction to COBOL, to maintaining sequential files and nonsequential files. The text also shows the new directions for COBOL in a chapter about Object-Oriented COBOL and an appendix devoted to the proposed changes in the COBOL 2000 standard.

All programs in the book can be run on personal computers or with minor modifications on mainframes or other platforms. The beauty of COBOL is that it can operate on any platform. This text provides instruction in ANS XOPEN standard COBOL. The one exception is the use of Micro Focus Object-Oriented COBOL in Chapter 20, since the final ANS standard has not yet been adopted.

## Improvements in the Third Edition

The third edition responds to the requests of many students and instructors to provide access to Windows-based tools while maintaining the proven approach to teaching COBOL. Features of this edition include the following:

- The text has been modified to show the development of programs in a Windows environment. While the essential characteristics of COBOL remain unchanged, the development tools have not. This edition provides examples using one of the most popular Windows development tools available: Micro Focus® Personal COBOL™ for Windows™.
- Chapter 19 has been added to explain the Year 2000 problem. This chapter discusses the sources of the problem, shows why it is a problem, and discusses several techniques to correct the problem. At the end of the chapter, we provide a list of World Wide Web sites where further information may be obtained.
- Another new chapter (Chapter 20) demonstrates the concepts of Object-Oriented COBOL. This new approach to COBOL promises to be a way for companies to maintain the value of their legacy COBOL programs while still being able to use the benefits of object-oriented programming.
- Appendix A provides extensive coverage of the Micro Focus Personal COBOL for Windows. In addition to explaining every menu item and button, this appendix also includes a brief tutorial allowing the student to experience process of creating a program.
- Appendix B provides a guide to installing Personal COBOL for Windows and several techniques to make using the product easier.
- Coverage of COBOL 2000 and intrinsic functions has been added in Appendix E. The 1989 extensions to COBOL 85 allow the use of predefined functions that had been missing in COBOL. This appendix also discusses the changes anticipated in COBOL 2000.
- In Appendix G, there are 32 new projects for student programming assignments. Many of these projects build on previous tasks allowing the

student to experience the development of systems and the performance of maintenance.

- Various chapters have been changed to incorporate the changes in debugging and editing techniques used with a Windows programming environment as opposed to using a DOS compiler and debugger.

## Benefits and Features

All of the features that have made the second edition successful have been retained and carried over into the third edition. These include:

- Immediate entry into COBOL programming, beginning in Chapter 1. Programming is learned by doing, and the book has students writing a complete program from the very beginning. Chapter 2 continues the discussion by having them execute the program of Chapter 1 in a thorough introduction to the programming process.
- Over 30 illustrative COBOL programs reinforce the discussion in the text and serve as both pedagogical aids and subsequent reference material. Every program is presented in a uniform and detailed format, including program narrative, record layouts, report layouts, test data, and processing specifications.
- A thorough discussion structured methodology, hierarchy charts, pseudocode, and top-down testing is presented in Chapter 3 and followed throughout. Students learn the proper way to develop programs early on and follow the procedure throughout the text.
- Every COBOL program in the text as well as data files for the student projects are available for download from a special World Wide Web site: [http://www.prenhall.com/grauer\\_cobol](http://www.prenhall.com/grauer_cobol).
- The availability of the sample listings enables students to reproduce and/or modify any of the programs without the tedium of data entry and further enhances the learning experience.
- An abundance of short-answer (true-false and fill-in) questions, COBOL problems, and programming projects for every chapter, with answers to the odd-numbered questions provided in Appendix F.
- Programming tips, dispersed throughout the text, that go beyond the syntactical rules of COBOL, and suggest stylistic considerations to make programs easier to read and maintain.
- Extensive use of graphic aids, featuring a two-color presentation, with figures to further clarify the presentation. Where Micro Focus Personal COBOL is discussed, actual pictures of the screens assist the student in understanding the user interface.
- System concept presentation at the beginning of most chapters, as COBOL instruction has come to require additional material beyond the language itself. There are detailed discussions of control breaks, data validation, techniques for table lookups and initialization, storing, the balance line algorithm for file maintenance and the organization of indexed files.
- While focusing on the proven techniques of structured programming and the established syntax of COBOL 85, the text also introduces the concepts of object-orientation and previews the significant changes in *COBOL 2000*.

## Software and Supplements

The following software and supplements are available from Prentice Hall:

- **SOFTWARE**—Micro Focus Personal COBOL for Windows 3.1 with object-orientation and Personal Dialog System. Compatible with Windows95 and WindowsNT, Personal COBOL provides all the tools to help you learn and use COBOL. The software includes an integrated editor, compiler and animator for creating, debugging and executing COBOL programs. Prentice Hall offers an affordable package of *COBOL: From Micro to Mainframe, Third Edition* with the Micro Focus Personal COBOL Compiler. Please order ISBN 0-13-975178-5.
- **WEB SITE**—Download every COBOL program in the text as well as data files for the nearly on hundred student projects from the *COBOL: From Micro to Mainframe* web site at: [http://www.prenhall.com/grauer\\_cobol](http://www.prenhall.com/grauer_cobol).
- **Instructor's Resource Manual** (ISBN# 0-13-081513-6)
- **Prentice Hall Custom Test**. Based on the powerful testing technology developed by Engineering Software Associates, Inc. (EAS), Prentice Hall Custom Test allows the educator to create and tailor the exam to their own needs. Please order ISBN# 0-13-081515-2

## Acknowledgments

We are especially grateful to our editors at Prentice Hall, Laura Steele, Alan Apt, and Marcia Horton, without whom this project would not have been possible. We also want to thank the many other individuals who helped produce the third edition. Irwin Zucker, who supervised the production, Kate Kaibni, editorial assistant, who worked hard to provide us with timely chapter reviews, and Joel Berman, our marketing manager at Prentice Hall, who developed the innovative campaign to make this book a success.

We also want to acknowledge our reviewers, who through their comments and constructive criticism, made this a far better book:

Robert V. Binder, Robert Binder Systems Consulting, Inc.  
Dinon Boyer, University of Akron  
Georgia Brown, Northern Illinois University  
Jan De Lassen, Brigham Young University  
Ida M. Flynn, University of Pittsburgh  
Frank T. Gergelyi, NJIT  
Ken Goldsmith, University of Miami  
Tom Gorecki, St. Charles Community College  
Carol C. Grimm, Palm Beach Community College  
Monica Holmes, Central Michigan University  
Ann W. Houck, Pima Community College  
David Lee  
James W. Payne, Kellogg Community College  
Nicholas Ross, University of Illinois at Chicago  
Wendell I. Pope, Utah State University  
Daniel H. Rindfleisch, Computer Specialist with Federal Government  
Daniel R. Rota, Robert Morris College  
Richard H. Saracusa, Northeastern University  
Ron Teemley, DeVry Institute of Technology  
Donat Valcourt, Northeastern University  
Ron Williams, McLennon Community College  
Jackie Zucker, University of Miami

A final word of thanks to you, our readers, for choosing this book. Please feel free to contact us with any comments or suggestions via email.

Robert Grauer  
rgrauer@umiami.miami.edu

Carol Vazquez Villar

Arthur R. Buss  
bussa@william.jewell.edu

# Contents

---

Preface iii

---

## Chapter 1: Introduction 1

Overview	2	Test Data	13
The First Problem	2	Elements of COBOL	13
Programming Specifications	3	Reserved Words	13
Required Logic	5	Programmer-Supplied Names	14
Flowcharts	6	Literals	15
Pseudocode	8	Symbols	16
A First Look at COBOL	9	Level Numbers	16
Identification Division	11	Picture Clauses	17
Environment Division	11	A Second Look at COBOL	17
Data Division	11	Summary	19
Procedure Division	12	Fill-in	20
		True/False	20
		Problems	21

---

## Chapter 2: From Coding Form to Computer 25

Overview	26	Errors in Compilation	33
From Coding Form to Computer	26	Errors in Execution	35
The COBOL Coding Form	28	Errors in Data Input	37
Use of an Editor	28	Evolution of COBOL	38
The Compile, Link, and Execute Sequence	30	There's Always a Reason	40
Learning by Doing	32	Summary	41
Errors in Entering the Program	33	Fill-in	41
Errors in Operating System Commands	33	True/False	42
		Problems	43

---

## Chapter 3: A Methodology for Program Development 47

Overview	48	Sufficiency of the Basic Structures	56
The Tuition Billing Problem	48	Expressing Logic	57
Structured Design	50	The Traditional Flowchart	57
Evaluating the Hierarchy Chart	52	Pseudocode	57
Completeness	53	Warnier-Orr Diagrams	59
Functionality	54	Top-Down Testing	61
Span of Control	54	Summary	66
Structured Programming	54	Fill-in	67
		True/False	68
		Problems	68



---

## Chapter 4: The Identification, Environment, and Data Divisions 73

Overview	74	WORKING-STORAGE SECTION	82
COBOL Notation	74	The Tuition Billing Program	84
Identification Division	75	Programming Specifications	84
Environment Division	76	COBOL Entries	87
CONFIGURATION SECTION	76	Limitations of COBOL-74	90
INPUT-OUTPUT SECTION	76	Summary	90
Data Division	77	Fill-in	92
FILE SECTION	77	True/False	92
		Problems	93

---

## Chapter 5: The Procedure Division 97

Overview	98	The ROUNDED Clause	109
OPEN	98	The SIZE ERROR Clause	110
CLOSE	99	COMPUTE	110
READ	99	ADD	112
Placement of the READ Statement	100	SUBTRACT	112
WRITE	100	MULTIPLY	114
STOP RUN	102	DIVIDE	115
MOVE	102	Programming Tip: Use the COMPUTE Statement	116
Restrictions on the Move Statement	103	Assumed Decimal Point	117
Alphanumeric Field to Alphanumeric Field	103	The Tuition Billing Program	118
Numeric Field to Numeric Field	104	Test Data	126
Group Moves	105	Hierarchy Chart	127
PERFORM	105	COBOL Program Skeleton	128
IF	106	Limitations of COBOL-74	128
The ELSE Clause	106	Summary	130
Indentation	106	Fill-in	131
EVALUATE	109	True/False	132
Arithmetic Statements	109	Problems	133

---

## Chapter 6: Debugging 139

Overview	140	Tips for Debugging	160
Errors in Compilation	140	DISPLAY Statement	161
Common Compilation Errors	149	The Structured Walkthrough	162
Errors in Execution	151	Summary	163
File Status Codes	156	Fill-in	164
Another Run Time Error	158	True/False	165
Logic Errors	159	Problems	165



---

## **Chapter 7: Editing and Coding Standards 169**

- Overview 170
  - Editing 170
    - The Decimal Point 172
    - Zero Suppression 172
    - Dollar Signs 174
    - Comma 174
    - Asterisks for Check Protection 175
    - Insertion Characters 175
    - Synopsis 175
  - Signed Numbers 176
    - CR and DB 176
    - Plus and Minus Signs 177
  - BLANK WHEN ZERO Clause 177
  - The Tuition Billing Program Revisited 178
  - Coding Standards 179
    - Data Division 179
    - Programming Tip: Avoid Literals 180
    - Procedure Division 181
    - Programming Tip: Use Scope Terminators 182
    - Both Divisions 183
  - A Well-Written Program 184
  - Summary 189
  - Fill-in 190    True/False 191    Problems 191
- 

## **Chapter 8: Data Validation 195**

- Overview 196
  - System Concepts: Data Validation 196
  - The IF Statement 197
    - Relational Condition 198
    - Class Test 199
    - Sign Test 200
    - Condition-Name Test (88-Level Entries) 200
    - Compound Test 200
    - Hierarchy of Operations 201
    - Implied Conditions 203
    - Nested IFs 203
    - NEXT SENTENCE 205
  - ACCEPT Statement 206
    - Calculations Involving Dates 206
  - The Stand-Alone Edit Program 207
    - Programming Specifications 208
    - Error Messages 211
    - Pseudocode 211
    - Hierarchy Chart 212
    - The Completed Program 212
  - Limitations of COBOL-74 221
  - Summary 223
  - Fill-in 224    True/False 224    Problems 225
- 

## **Chapter 9 : More About the Procedure Division 229**

- Overview 230
- PERFORM 231
  - TEST BEFORE/TEST AFTER 231
  - In-line Perform 232
  - Performing Sections 232
  - PERFORM THRU 232
  - Programming Tip: Perform Paragraphs, Not Sections 233
- READ 234
  - False-Condition Branch 234
  - READ INTO 235
- WRITE FROM 236
- INITIALIZE 236
- String Processing 237
  - INSPECT 237
- STRING 238
- UNSTRING 240
  - Reference Modification 240
- ACCEPT 242
- Duplicate Data Names 243
  - Qualification 244
- MOVE CORRESPONDING 245
- The Car Billing Program 246
  - Programming Specifications 246
  - Program Design 248
  - The Completed Program 249
- Limitations of COBOL-74 258
- Summary 258
- Fill-in 259    True/False 260    Problems 260



**Chapter 10: Screen I-O****265**

- Overview 266
- ACCEPT 266
  - Programming Tip: Micro Focus Level 78—The Use of COBOL Constants 267
- DISPLAY 269
- The Tuition Billing Program Revisited 270
  - Programming Specifications 270
  - Hierarchy Chart 271
  - Pseudocode 271
  - The Completed Program 273
  - Programming Tip: The Hidden Power of the Alt key 274
- Car Validation and Billing Program 278
  - Programming Specifications 279
  - The Screen Section 280
  - Hierarchy Chart 284
  - Pseudocode 284
  - The Completed Program 287
- Limitations of COBOL-74 297
- Summary 297
- Fill-in 298    True/False 299    Problems 299

**Chapter 11: Introduction to Tables****301**

- Overview 302
- Introduction to Tables 302
  - OCCURS Clause 303
  - Processing a Table 304
  - PERFORM VARYING 304
- A Second Example 306
  - Problems with the OCCURS Clause 308
  - Rules for Subscripts 308
  - Relative Subscripting 308
  - USAGE Clause 309
- OCCURS DEPENDING ON 310
- The Student Transcript Program 311
  - Programming Specifications 311
  - Program Design 313
  - The Completed Program 314
- Indexes versus Subscripts 321
  - The SET Statement 322
- Limitations of COBOL-74 322
- Summary 325
- Fill-in 326    True/False 326    Problems 327

**Chapter 12: Table Lookups****331**

- Overview 332
- System Concepts 332
  - Types of Codes 333
  - Characteristics of Codes 333
  - Sequential Table Lookup 334
  - Binary Table Lookup 335
  - Positional Organization and Direct Lookups 336
- Initializing a Table 336
  - Hard Coding 336
  - Input-Loaded Tables 338
- Table Lookups 339
  - PERFORM VARYING Statement 340
  - SEARCH Statement 340
- Programming Tip—Restrict Subscripts and Switches to a Single Use 342
- SEARCH ALL Statement 344
- Direct Lookup 344
- Range-Step Tables 345
- A Complete Example 347
  - Programming Specifications 347
  - Program Design 349
  - The Completed Program 350
- Limitations of COBOL-74 357
- Summary 357
- Fill-in 358    True/False 359    Problems 360



**Chapter 13: Multilevel Tables****363**

- Overview 364
- System Concepts 364
- COBOL Implementation 366
- One-Level Tables 366
  - PERFORM VARYING 366
- Two-Level Tables 368
  - Errors in Compilation 369
  - PERFORM VARYING 370
- A Sample Program 373
  - Programming Specifications 373
  - Program Design 375
  - The Completed Program 375
- Three-Level Tables 380
  - PERFORM VARYING 382
- A Sample Program 384
  - Programming Specifications 384
  - The Completed Program 386
- Table Lookups 390
- A Calorie Counter's Delight 392
  - Programming Specifications 392
  - Range-Step Tables 392
  - The Completed Program 394
- Limitations of COBOL-74 398
- Summary 398
- Fill-in 399    True/False 399    Problems 400

**Chapter 14: Sorting****403**

- Overview 404
- System Concepts 405
- Collating Sequence 405
  - Embedded Sign 406
- COBOL Implementation 408
  - SORT Statement 409
  - SD (Sort Description) 410
  - RELEASE and RETURN 410
  - Programming Specifications 411
- USING/GIVING Option 414
- INPUT PROCEDURE/OUTPUT PROCEDURE
  - Option 419
- Comparing Options 426
- MERGE Statement 426
- Limitations of COBOL-74 428
- Summary 428
- Fill-in 429    True/False 430    Problems 431

**Chapter 15: Control Breaks****435**

- Overview 436
- System Concepts 436
  - Running versus Rolling Totals 440
- One-Level Control Breaks 443
  - Programming Specifications 443
  - Hierarchy Chart 444
  - Pseudocode 446
  - The Completed Program 446
- Two-Level Control Breaks 451
  - Hierarchy Chart 451
  - Pseudocode 452
- The Completed Program 454
- Three-Level Control Breaks 460
  - Hierarchy Chart 460
  - Pseudocode 462
  - The Completed Program 463
  - Programming Tip: How to Write a Control Break
    - Program 470
- Limitations of COBOL-74 471
- Summary 471
- Fill-in 472    True/False 472    Problems 473



---

**Chapter 16: Subprograms**
**475**

Overview 476  
 Subprograms 477  
   Called and Calling Programs 477  
   COPY Statement 479  
   Calling BY CONTENT and BY REFERENCE 480  
   Programming Tip: Use COPY to Pass  
     Parameters 481  
 INITIAL Clause 482  
 A System for Physical Fitness 482  
   Programming Specifications 482  
   Hierarchy Chart 485  
   Pseudocode 485

The Completed Programs 486  
   Main Program (FITNESS) 486  
   Input Program (INPUTSUB) 490  
   Weight-Range Program (WGTSUB) 495  
   Training Program (TRAINSUB) 498  
   Display Program (DSPLYSUB) 498  
   Time Program (TIMESUB) 503  
 The Linkage Editor 504  
   Problems with the Linkage Editor 505  
 Limitations of COBOL-74 506  
 Summary 508  
 Fill-in 509    True/False 509    Problems 510

---

**Chapter 17: Sequential File Maintenance**
**515**

Overview 516  
 System Concepts 516  
   Sequential versus Nonsequential Processing 518  
   Periodic Maintenance 518  
 Data Validation 519  
   Programming Specifications 520  
   Designing the Program 523  
   The Completed Program 524  
 Sequential File Maintenance 528

Programming Specifications 528  
 The Balance Line Algorithm 529  
 Designing the Hierarchy Chart 531  
 Top-Down Testing 535  
   The Stubs Program 535  
   The Completed Program 540  
 Summary 545  
 Fill-in 546    True/False 546    Problems 547

---

**Chapter 18: Indexed Files**
**549**

Overview 550  
 System Concepts 550  
 COBOL Implementation 554  
 Creating an Indexed File 556  
   Programming Specifications 556  
   Pseudocode 557  
   The Completed Program 557  
 Additional COBOL Elements 559  
   OPEN 559  
   READ 560  
   WRITE 562  
   REWRITE 562  
   DELETE 562

Maintaining an Indexed File 563  
   Programming Specifications 563  
   Hierarchy Chart 564  
   Pseudocode 566  
   The Completed Program 566  
 Alternate Record Key 570  
   Programming Specifications 570  
 Concatenated Key 573  
   The START Statement 574  
 Limitations of COBOL-74 574  
 Summary 576  
 Fill-in 577    True/False 578    Problems 578



---

## **Chapter 19: The Year 2000 Problem** **583**

Overview	584	Leap-Year Problem	594
The Year 2000 Problem	584	Retirement Program Revisited	594
Date Arithmetic	590	Summary	599
COBOL Intrinsic Calendar Functions	591	Fill-in	600
		True/False	600
		Problems	601

---

## **Chapter 20: Object-Oriented COBOL Programming** **603**

Overview	604	The StudentDM Class	621
The Next Generation of COBOL	605	The StudentDM Instance Definition	625
The Development of Structured Programming	606	The Student Class	627
Terminology	607	The Person Class	630
The Object-Oriented versus Structured Paradigm	608	The Student UI Class	633
Student-Look-UP Program	612	The Student PRT Class	635
The Registrar Class	616	Conclusion	639
Classes and Inheritance	619	Summary	640
ProcessRequests Method	619	Fill-in	641
		True/False	641
		Problems	642

---

## **Appendix A: Micro Focus Personal COBOL for Windows: Users Guide and Tutorial** **643**

## **Appendix B: Getting Started** **697**

## **Appendix C: Reserved Words** **709**

## **Appendix D: COBOL-85 Reference Summary** **711**

## **Appendix E: COBOL 1997, 1998, 1999, 2000, or ...?** **737**

## **Appendix F: Answers to Odd-Numbered Exercises** **745**

## **Appendix G: Projects** **753**

## **Index** **893**



# 1

# Introduction

## CHAPTER OUTLINE

Overview

### **The First Problem**

Programming Specifications

### **Required Logic**

Flowcharts

Pseudocode

### **A First Look at COBOL**

Identification Division

Environment Division

Data Division

Procedure Division

### **Test Data**

### **Elements of COBOL**

Reserved Words

Programmer-Supplied Names

Literals

Symbols

Level Numbers

PICTURE Clauses

### **A Second Look at COBOL**

Summary

Fill-in

True/False

Problems



## O B J E C T I V E S

After reading this chapter you will be able to:

- Define the terms: field, record, and file.
- Name two techniques used to express program logic.
- Identify the four divisions of a COBOL program.
- State the six COBOL language elements.
- State the rules for creating a programmer-supplied name; distinguish between examples of valid and invalid names.
- State the difference between numeric and nonnumeric literals; recognize valid and invalid examples of each.
- Follow the logic of a simple program as expressed in a flowchart or pseudocode.

## O V E R V I E W

This book is about computer programming. In particular, it is about COBOL, a widely used commercial programming language. Programming involves the translation of an algorithm (a precise means of solving a problem) into a form the computer can understand. Programming is necessary because, despite reports to the contrary, computers cannot think for themselves. Instead, they do exactly what they have been instructed to do, and these instructions take the form of a computer program. The advantage of the computer stems from its speed and accuracy. It does not do anything that a human being could not do, given sufficient time and memory capacity.

We begin our study of computer programming by describing a simple problem and then developing the logic and COBOL program to solve it. This rapid entrance into COBOL is somewhat different from the approach followed by most textbooks, but we believe in learning by doing. There is nothing very mysterious about COBOL programming, so let's get started.

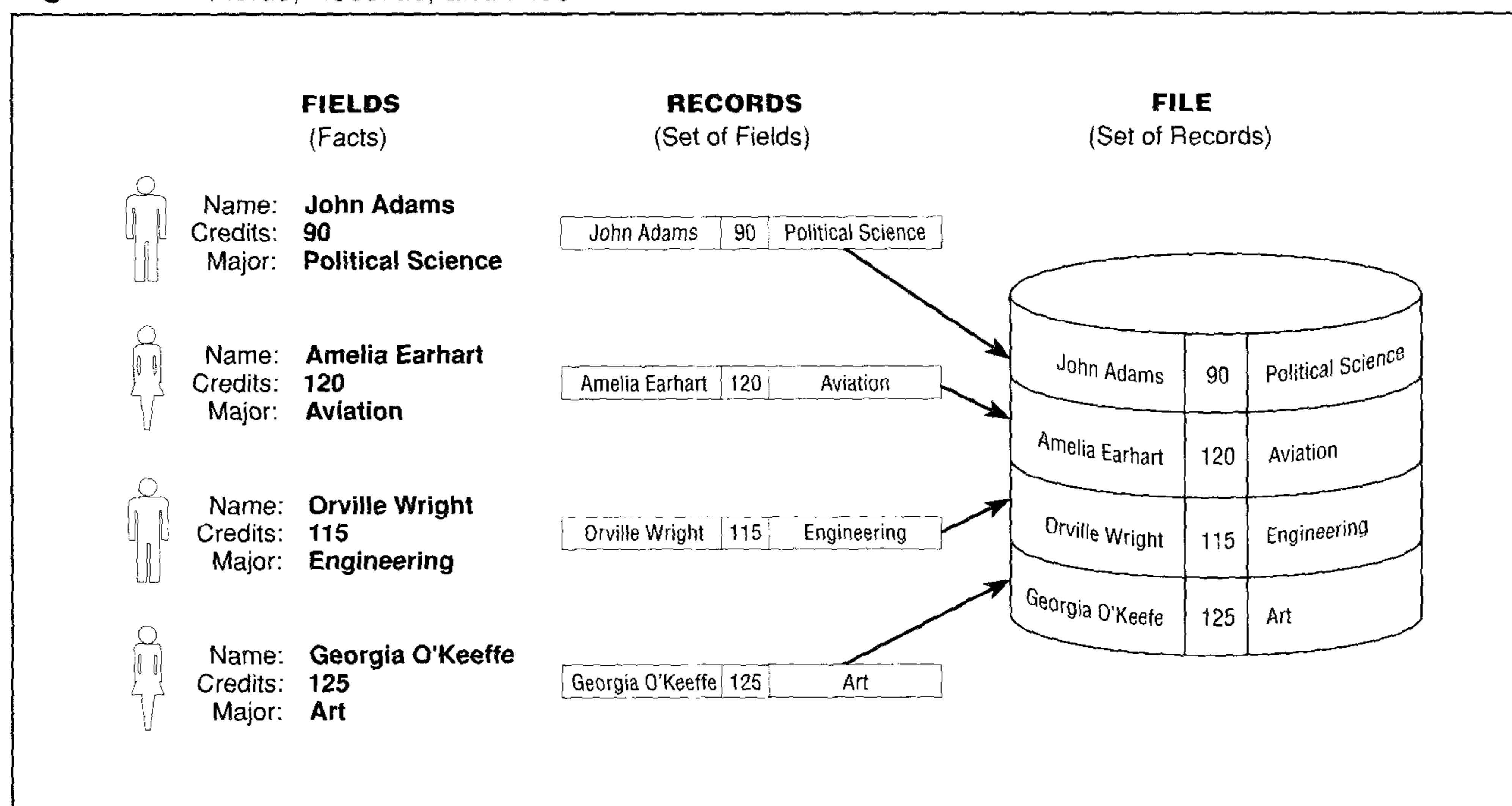
---

### The First Problem

Our first problem is set in the context of a university, and involves a set of student records, one record per student. Each record contains the student's name, number of completed credits, and major. Implicit in this statement are the definitions of three fundamental terms: field, record, and file. A *field* is a basic fact, such as the name, address, major, grade point average, or number of completed credits. A *record* is a set of fields, and a *file* is a set of records. Thus, if there were 1,000 students, there would be 1,000 records (one for each student), each consisting of five fields, and comprising a single student file.

To clarify this relationship, we create four hypothetical students for our problem: John Adams, Amelia Earhart, Orville Wright, and Georgia O'Keeffe. There are many facts about each of our students, but our problem utilizes only three:



**Figure 1.1** Fields, Records, and Files

name, major, and credits completed. Figure 1.1 represents these concepts in pictorial fashion. Each fact about each student comprises a single field. The three fields collectively make up that student's record. The four records (one for each of our students) compose the student file.

The problem is to process the file of student records and produce a list of engineering students who have completed more than 110 credits. It is a typical problem, in that its solution will address the three elements common to all computer applications: input, processing, and output. As shown in Figure 1.2, the student file, just defined, is the *input*; this file is *processed* by determining which students are engineering majors with more than 110 credits; and consequently, a report is created as *output*, reflecting these students.

The input to a computer program; that is, the precise arrangement of the various fields in each incoming record, has to be specified exactly. Figure 1.3a is a common way to communicate this information, and shows that the student's name is contained in positions 1–25, the number of credits in positions 26–28, and the student's major in positions 29–43. Note too, that every record in a given file must have the identical record layout.

In similar fashion, the report produced as output is also precisely designed. Figure 1.3c shows a print layout chart, in which descriptive information appears on line one, with the names of selected students in columns 9–33 of subsequent lines. Observe also that the location of the name field is different in the input and output records (positions 1–25 and 9–33, respectively), and that each input record contains three fields, but that each line of output has been designed to contain only one field.

## Programming Specifications

It is important that programming specification—that is, the *input*, *processing*, and *output* requirements—be presented in a clear and unambiguous fashion.