



# ADVANCED Structured COBOL

---

Gary S. Popkin

# ADVANCED STRUCTURED COBOL

**Gary S. Popkin**

New York City Technical College

A GIFT OF  
**THE ASIA FOUNDATION**  
BOOKS FOR ASIA  
SAN FRANCISCO, CALIFORNIA, U.S.A.  
美國亞洲基金會敬贈



**Kent Publishing Company**

A Division of Wadsworth, Inc.  
Boston, Massachusetts

## **To Mickey and Frank**

for many years of  
inspiration, help, and friendship

**Senior Editor:** Richard C. Crews  
**Production Editor:** Rachel Hockett/Cobb-Dunlop  
Publisher Services, Inc.  
**Cover Designer:** Glenna Lang  
**Production Coordinator:** Linda Siegrist

### **Kent Publishing Company**

A Division of Wadsworth, Inc.

© 1983 by Wadsworth, Inc., 10 Davis Drive, Belmont, California 94002. All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transcribed, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher, Kent Publishing Company, 20 Park Plaza, Boston, Massachusetts 02116.

Printed in the United States of America

3 4 5 6 7 8 9 — 87 86 85

### **Library of Congress Cataloging in Publication Data**

Popkin, Gary S.

Advanced structured COBOL.

Includes index.

1. COBOL (Computer program language) 2. Structured programming. I. Title. II. Title: Advanced structured C.O.B.O.L.

QA76.73.C25P667 1983

001.64'24

82-21278

ISBN 0-534-01394-5

# PREFACE

The primary objective of this book is to teach advanced COBOL programming skills to students who have already had one beginning COBOL course. Seven main topics are covered in 12 chapters. The topics are: report writer, table handling (tables of one, two, and three dimensions), sorting and merging, file maintenance (sequential, indexed, and relative files), string processing, subprograms, and debugging. Forty-four complete, working COBOL programs are shown in facsimile, along with their inputs and outputs. Additional figures show the JCL needed to run the programs under OS, and the commands needed to create several different VSAM files.

The programs are written to comply with the highest level of the 1974 American National Standard COBOL and with IBM OS/VS COBOL. The programs should therefore run on most compilers in use today. Where the ANSI standard and the IBM compiler are incompatible, as in a few minor instances in report writer, incompatibilities were resolved in favor of the compiler so that the programs shown in this book would run.

There is no discussion in this book of older systems that may not have all the capabilities of 1974 standard COBOL. If you are using one of the older systems, you may find that some of the features used in some of the sample programs in this book are not available to you. If this is the case, you will have to develop alternative methods to code those features. For example, if your COBOL system does not allow the slash (/) as an editing character, you will have to modify some of the sample programs to avoid using it.

This text assumes that the student is familiar with the topics usually covered in a beginning COBOL course, such as those contained in Chapters 1–8 of my book *Introductory Structured COBOL Programming* (Kent, 1981). Professional programmers who are working in a language other than COBOL, and who know some COBOL and wish to become proficient in it, will probably also find this book suitable.

The book is organized in modular fashion. Although I recommend covering the chapters in the order in which they appear in the book, an advanced course could be taught with only the following chapters required, in this order:

4. Sorting and Merging
5. Magnetic File Media
6. Processing Sequential Master Files

All other chapters can be covered at any time after their prerequisite topics are covered. The remaining chapters and their prerequisites are:

<i>Chapter</i>	<i>Prerequisite</i>
1. Report Writer	None
2. One-Dimensional Tables	None
3. Two- and Three-Dimensional Tables	One-Dimensional Tables
7. Indexed Files	Processing Sequential Master Files
8. Relative Files	Processing Sequential Master Files and One-Dimensional Tables
9. Introduction to VSAM File Processing	Indexed Files or Relative Files

<i>Chapter</i>	<i>Prerequisite</i>
10. String Processing	One-Dimensional Tables
11. Subprograms	None
12. The Debugging Feature	One-Dimensional Tables

I recommend that you cover Report Writer first and then use it in all programs in all subsequent chapters, as I have done. This approach has the substantial advantage of simplifying all programs and permitting the student to concentrate on a substantive topic. For example, in programs on file maintenance, use of Report Writer greatly simplifies the coding relating to the printing of error reports and transaction registers, and permits the student to concentrate on the logic of file processing. If for any reason you cannot or do not wish to use Report Writer, students will have to code the functions of page overflow, control breaks, line formatting, and line spacing by hand. Methods of coding such functions are covered fully in beginning COBOL textbooks and are not repeated here. See, for example, Chapters 2 and 6 of *Introductory Structured COBOL Programming* for full treatment of those topics, with programming examples.

The approach used to teach programming skills is practical and intuitive. Abstract statements are kept to a minimum. Examples are used abundantly, especially examples of complete programs. Exercises are placed within the body of each chapter (except Chapter 9) to reinforce each topic where it is discussed. Each chapter contains a summary and fill-in exercises, and 11 of the 12 chapters have review exercises which cover all the topics of the chapter.

Each chapter begins with a list of key points that the student should expect to learn from the chapter, followed by a list of key words contained in the chapter. At the first appearance of each key word in the body of the chapter, the word appears in bold face, where it is explained and used in context.

An Instructor's Manual contains teaching suggestions for each chapter, answers to the chapter exercises, and numerous transparency masters for classroom use.

I wish to thank these people for their help in putting this book together: Myron R. Myerson, Instructor of Data Processing at New York City Technical College, who wrote many of the program examples, provided programming ideas, and made valuable suggestions on the manuscript; Shameze Sultan and Ronald Charles, then both students at NYCTC, who wrote many program examples; Frank M. Rand, Chairman of the Data Processing Department at NYCTC, who provided me with the necessary time and equipment; Michael L. Trombetta of Queensboro Community College for providing research materials and programming suggestions; Louise Moran of New York Life Insurance Company for providing programming suggestions; and James Cronen of the NYCTC Computer Center for valuable assistance in preparing the manuscript.

The author also wishes to thank the following reviewers for their many useful suggestions in the preparation of the manuscript:

Robert D. Chenoweth  
Ocean County College

David W. Chilson  
Bowling Green State University

George W. Elder  
City College of New York

Richard E. Frost  
State University College, Potsdam, NY

Mike Marlow  
Illinois State University

Linda L. Rice  
Saddleback College

Lawrence Scott  
State University College of New York

Mike Swafford  
Tulsa Junior College

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 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-Honneywell

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.

Gary S. Popkin

# CONTENTS

## 1. REPORT WRITER AND DECLARATIVES 1

The Report Section and Report  
Groups 2  
A Final Total Using Report  
Writer 6  
Multiple Control Breaks Using  
Report Writer 13  
Declaratives 22  
The SUM Clause 39  
Summary 39  
Fill-in Exercises 40  
Review Exercises 40

## 2. ONE-DIMENSIONAL TABLES 45

A Program Using  
Subscripting 46  
Controlling the Value of a  
Subscript with a PERFORM  
Statement 53  
Tables 55  
The SET Statement 66  
The SEARCH Statement 66  
A Program Using an  
Argument-Function  
Table 67  
The SEARCH ALL  
Statement 73  
Format 2 of the SET  
Statement 74  
Summary 75  
Fill-in Exercises 75  
Review Exercises 75

## 3. TWO- AND THREE- DIMENSIONAL TABLES 79

A Program with a  
Two-dimensional Table 80  
The PERFORM Statement 88  
A Two-dimensional Table with  
VALUE Clauses 91  
The PICTURE Character P 98  
Searching a Table for a Range of  
Values 101  
Relative Indexing 108  
Programming for a  
Three-dimensional  
Table 109  
Summary 120  
Fill-in Exercises 120  
Review Exercises 120

## 4. SORTING AND MERGING 123

Using the SORT Verb 124  
The SORT Verb with USING and  
GIVING 129  
Using an OUTPUT  
PROCEDURE 130  
A Program with an OUTPUT  
PROCEDURE 130  
The SORT Statement 134  
The RETURN Statement 135  
Using an INPUT  
PROCEDURE 136  
A Program with an INPUT  
PROCEDURE 136  
The RELEASE Statement 141  
The MERGE Statement 142

Using the MERGE  
Statement 142  
COLLATING SEQUENCE and  
Alphabet Name 154  
Summary 155  
Fill-in Exercises 156  
Review Exercises 156

## **5. MAGNETIC FILE MEDIA 159**

Advantages of Magnetic  
Media 160  
Uses of Magnetic Media 161  
A Master File 161  
Validity of Data on a Master  
File 162  
Creating a Sequential Master  
File 163  
Building a Master Record in  
Storage 171  
Deleting Records from a  
Sequential File 179  
Summary 190  
Fill-in Exercises 190  
Review Exercises 191

## **6. PROCESSING SEQUENTIAL MASTER FILES 193**

Listing Selected Records from a  
Master File 194  
An Update Program with  
Changes and  
Deletions 203  
The Level-66 Entry 216  
A Complete Update  
Program 218  
Listing the Contents of a  
Sequential File on  
Tape 230  
Summary 233  
Fill-in Exercises 233  
Review Exercises 234

## **7. INDEXED FILES 239**

A Summary of File  
Terminology 240  
Creating an Indexed File on  
Disk 241  
The DISPLAY Statement 250  
Updating an Indexed File 251  
A Program to Update an Indexed  
File 252  
Using Dynamic Access 266  
A Program Using Dynamic  
Access 268  
The START Statement 276  
The Sequential READ  
Statement 276  
Listing the Complete Contents of  
an Indexed File 278  
Other Operations on Indexed  
Files 281  
Summary 281  
Fill-in Exercises 282  
Review Exercises 282

## **8. RELATIVE FILES 287**

Using a Relative File 288  
Creating a Relative File  
Sequentially 288  
Updating a Relative File  
Randomly 293  
Processing a Relative File  
Sequentially 307  
An Application Using  
Randomizing 314  
Creating a Relative File  
Randomly 315  
Updating a Randomized Relative  
File 323  
Listing the Contents of a Relative  
File 334  
Other Operations on Relative  
Files 338  
Summary 338  
Fill-in Exercises 338  
Review Exercises 339



## **9. INTRODUCTION TO VSAM FILE PROCESSING 341**

References 342  
 Access Method Services 343  
 VSAM Catalogs 343  
 JCL with VSAM 344  
 Access Method Services  
   Commands 345  
 Creating an Indexed File 347  
 Using an Indexed File 356  
 Creating an Alternate Index with  
   AIXBLD 356  
 Creating a Relative File 361  
 Summary 362  
 Fill-in Exercises 363

## **10. STRING PROCESSING 365**

The STRING Statement 365  
 Using the STRING Verb 366  
 Another Application of the  
   STRING Verb 372  
 The UNSTRING Statement 378  
 Using the UNSTRING  
   Statement 379  
 The Format of the UNSTRING  
   Statement 386  
 Using STRING and UNSTRING  
   Together 387  
 Using the INSPECT Verb 393  
 The INSPECT Statement 397  
 Summary 398  
 Fill-in Exercises 399  
 Review Exercises 400

## **11. SUBPROGRAMS 401**

The Uses of Subprograms 401  
 Passing Data Between  
   Programs 402  
 Writing a Subprogram in  
   COBOL 403

Writing a Calling Program 406  
 The CALL Statement 410  
 Passing Data in Both  
   Directions 410  
 The Use of the CANCEL  
   Statement 416  
 Summary 417  
 Fill-in Exercises 418  
 Review Exercises 418

## **12. THE DEBUGGING FEATURE 419**

Debugging Lines 419  
 A Program with Debugging  
   Lines 420  
 Debugging Sections 430  
 The Special Register DEBUG-  
   ITEM 430  
 Execution of Debugging  
   Sections 431  
 Using Debugging Sections 434  
 The USE FOR DEBUGGING  
   Sentence 440  
 Use the Debugging Feature 441  
 Summary 441  
 Fill-in Exercises 441  
 Review Exercises 442

## **Appendix A. American National Standard List of COBOL Reserved Words 443**

## **Appendix B. Complete ANSI Reference Summary 445**

## **Appendix C. OS Job Control 469**

## **Glossary 483**

## **Index 499**

# REPORT WRITER AND DECLARATIVES

Here are the key points you should learn from this chapter:

1. The concept of automatic generation of coding for report output
2. The features of the COBOL Report Writer
3. How to use Report Writer to produce a variety of reports
4. How to use declaratives in connection with Report Writer

Key words to recognize and learn:

Report Writer	FIRST DETAIL
declarative	absolute LINE NUMBER clause
Report Section	relative LINE NUMBER clause
REPORT IS	SUM
report name	sum counter
RD	group indication
report description entry	LAST DETAIL
report group	FOOTING
TYPE	PAGE-COUNTER
DETAIL	LINE-COUNTER
REPORT HEADING	GROUP INDICATE
PAGE HEADING	NEXT GROUP
CONTROL FOOTING	ALL
PAGE FOOTING	NEXT PAGE
REPORT FOOTING	DECLARATIVES
LINE	USE
PLUS	BEFORE REPORTING
COLUMN	SUPPRESS
SOURCE	PRINT-SWITCH
INITIATE	END
GENERATE	CONTROL HEADING
TERMINATE	absolute NEXT GROUP clause
CONTROL	relative NEXT GROUP clause
FINAL	UPON
PAGE	nonprintable item

The **Report Writer** feature of COBOL makes programming for report output easier. The Report Writer provides, automatically, coding that would otherwise have to be written step by step by the programmer. Using Report Writer, the programmer can describe certain characteristics that the output report is to have, and Report Writer generates the coding needed to make the report look that way. For example, Report Writer can provide all the page overflow coding needed in a

program. The programmer need only tell Report Writer how big the page is, and Report Writer provides coding that will count lines as they print, test for page overflow, and skip to a new page and print headings when necessary. Report Writer can also provide coding that will take totals. If a total line is to print at the end of a report, the programmer need not code any of the totaling logic but just tell Report Writer which fields are to be totaled, and all the necessary coding will be provided automatically.

Report Writer also contains control break logic. If control breaks are needed on a report, the programmer need only say what the control fields are, and Report Writer provides coding to test for control breaks and print the appropriate total lines.

Another feature of Report Writer that makes programming easier is the way that output line formats are specified in Report Writer. Just tell Report Writer in which print position each field is to print, and it provides all the necessary coding. There is no need to count **FILLER** spaces.

There have been many schemes to automatically produce programs that print reports. But Report Writer is more than a means to produce reports. Report Writer is the printing component of the whole powerful COBOL language. In using Report Writer, the programmer does not give up any of the capabilities of COBOL; the programmer still has command over all the output editing features, the nested IF, and compound conditions. Thus Report Writer should be used in any COBOL program that produces printed output. For no matter what else the program might be doing, no matter how large the program or how involved the logic, and no matter the volume of printed output produced by the program, Report Writer should handle the formatting and printing of all lines.

Even though Report Writer is extremely flexible and powerful, it sometimes happens that the programmer needs one or more features that Report Writer does not contain. Any such features can be hand-coded using a **declarative** section, as will be shown later in the chapter.

## The Report Section and Report Groups

The first program we will write using Report Writer reads cards and prints the contents of each one, reformatted, on one line. There are no page or column headings. The input format for this program is shown in Figure 1.1, and the output format in Figure 1.2.

**Figure 1.1** Input format for Program 38.

[illegible]

**Figure 1.2** Output format for Program 38.

[illegible]

Program 38 is shown in Figure 1.3. In the Data Division we see the **Report Section**, beginning at line 00490. The Report Section must appear whenever Report Writer is used, and it must be the last section in the Data Division. In the File Section, the output file FD entry, line 00420, has no level-01 entry associated with it. Instead, the **REPORT IS** clause in the FD entry tells COBOL that Report Writer will be writing out a report on this file. The **REPORT IS** clause gives a name to the report that is being produced. In this program the **report name** is **EMPLOYEE-REPORT**. The rules for making up report names are the same as for making up file names.

**Figure 1.3** Program 38.

0-CBI RELEASE 2.3 JULY 24, 1978

IBM OS/VS COBOL

```

00010 IDENTIFICATION DIVISION.
00020
00030 PROGRAM-ID.
00040     PRG38.
00050
00060 *AUTHOR.  MYRON R. MYERSON.
00070 *
00080 *      THIS PROGRAM READS A DECK OF CARDS AND
00090 *      PRINTS THE CONTENTS OF EACH CARD ON ONE LINE.
00100 *
00110 *****
00120
00130 ENVIRONMENT DIVISION.
00140
00150 CONFIGURATION SECTION.
00160 SOURCE-COMPUTER.
00170     IBM-370.
00180 OBJECT-COMPUTER.
00190     IBM-370.
00200
00210 INPUT-OUTPUT SECTION.
00220 FILE-CONTROL.
00230     SELECT EMPLOYEE-DATA-FILE-IN      ASSIGN TO CARDIN.
00240     SELECT EMPLOYEE-DATA-FILE-OUT     ASSIGN TO PRINTER.
00250
00260 *****

```

(continued)

Figure 1.3 (end)

```

00270
00280 DATA DIVISION.
00290
00300 FILE SECTION.
00310 FD EMPLOYEE-DATA-FILE-IN
00320 LABEL RECORDS ARE OMITTED
00330 RECORD CONTAINS 80 CHARACTERS.
00340
00350 01 EMPLOYEE-DATA-RECORD-IN.
00360 05 SOCIAL-SECURITY-NUMBER-IN PIC X(9).
00370 05 EMPLOYEE-NUMBER-IN PIC X(5).
00380 05 ANNUAL-SALARY-IN PIC X(7).
00390 05 EMPLOYEE-NAME-IN PIC X(25).
00400 05 FILLER PIC X(34).
00410
00420 FD EMPLOYEE-DATA-FILE-OUT
00430 LABEL RECORDS ARE OMITTED
00440 REPORT IS EMPLOYEE-REPORT.
00450
00460 WORKING-STORAGE SECTION.
00470 01 MORE-INPUT PIC X VALUE "Y".
00480
00490 REPORT SECTION.
00500 RD EMPLOYEE-REPORT.
00510
00520 01 REPORT-LINE
00530 TYPE DETAIL
00540 LINE PLUS 1.
00550 05 COLUMN 5 PIC X(5) SOURCE EMPLOYEE-NUMBER-IN.
00560 05 COLUMN 12 PIC X(9) SOURCE SOCIAL-SECURITY-NUMBER-IN.
00570 05 COLUMN 25 PIC X(25) SOURCE EMPLOYEE-NAME-IN.
00580 05 COLUMN 52 PIC X(7) SOURCE ANNUAL-SALARY-IN.
00590
00600 *****
00610
00620 PROCEDURE DIVISION.
00630
00640 CONTROL-PARAGRAPH.
00650 PERFORM INITIALIZATION.
00660 PERFORM MAIN-PROCESS UNTIL MORE-INPUT IS EQUAL TO "N".
00670 PERFORM TERMINATION.
00680 STOP RUN.
00690
00700 INITIALIZATION.
00710 OPEN INPUT EMPLOYEE-DATA-FILE-IN,
00720 OUTPUT EMPLOYEE-DATA-FILE-OUT.
00730 INITIATE EMPLOYEE-REPORT.
00740 PERFORM READ-A-CARD.
00750
00760 MAIN-PROCESS.
00770 GENERATE REPORT-LINE.
00780 PERFORM READ-A-CARD.
00790
00800 TERMINATION.
00810 TERMINATE EMPLOYEE-REPORT.
00820 CLOSE EMPLOYEE-DATA-FILE-IN,
00830 EMPLOYEE-DATA-FILE-OUT.
00840
00850 READ-A-CARD.
00860 READ EMPLOYEE-DATA-FILE-IN
00870 AT END
00880 MOVE "N" TO MORE-INPUT.

```

Every report name given in the File Section must appear in the Report Section as part of an **RD** entry (**report description entry**). Within the RD entry and the level numbers that follow it, many characteristics of the report are described. Report Writer uses these descriptions of the report to create the necessary coding.

Each level-01 entry following the RD entry describes a different type of line, or a group of related lines, that may appear on the report. The line or lines appearing under a level-01 entry is called a **report group**. In Program 38, the level-01 entry, shown in line 00520, describes a report group called REPORT-LINE. In this case, the report group consists of only one line. The rules for making up report group names are the same as for making up data names.

The report group **REPORT-LINE** is shown as **TYPE DETAIL**, meaning that this is a detail line on the report. Some other **TYPEs** of report groups that may be described are:

- a. **REPORT HEADING**, one or more lines to print only once on the report at the beginning
- b. **PAGE HEADING**, one or more lines to print at the top of every page
- c. **CONTROL FOOTING**, one or more lines to print after a control break has been detected
- d. **PAGE FOOTING**, one or more lines to print at the bottom of each page before skipping to a new page
- e. **REPORT FOOTING**, one or more lines to print at the end of the report

The clause **LINE PLUS 1**, at line 00540, tells Report Writer that the detail lines on this report are to be single-spaced. The clause **LINE PLUS 1** means that each detail line is to be printed on whatever **LINE** the previous one was printed, **PLUS 1**. One way or another, you must always explicitly tell Report Writer where to put every line it prints.

We come now to the level-05 entries, beginning with line 00550, where we describe the individual fields that make up **REPORT-LINE**. Using the **COLUMN** clause, we tell Report Writer the print position where each field on the line begins. The column numbers shown in these **COLUMN** clauses were taken directly from the print spacing chart in Figure 1.2. There is no need for **FILLER** entries or for counting the number of blanks between fields.

The **SOURCE** clause tells Report Writer where the data come from in the Data Division to fill each field. The **SOURCE** of a print field may be any identifier anywhere in the File Section or Working Storage Section or certain fields in the Report Section.

The Procedure Division introduces three new verbs. The **INITIATE** statement, line 00730, must be issued to initialize the report file. It must be issued after the output file has been **OPENed** in the usual way and before a **GENERATE** verb is issued for that report. The **INITIATE** statement must include the report name as it appears in the **RD** entry. The **GENERATE** verb may be used to print a report group. In this case we have only the one report group **REPORT-LINE**. The statement **GENERATE REPORT-LINE**, at line 00770, does everything: It blanks the output areas that should be blank, moves data from the input area to the print line, single-spaces the paper, and writes a line.

After the complete report is written, a **TERMINATE** statement must be issued for the report name, as shown in line 00810. This must be done before the file is **CLOSEd**. Program 38 was run with the input data shown in Figure 1.4 and produced the output shown in Figure 1.5.

**Figure 1.4** Input data for Program 38.

```
100040002105035000000MORALES, LUIS
101850005108904651000JACOBSON, MRS. NELLIE
201110008112774302000GREENWOOD, JAMES
209560011116643953000COSTELLO, JOSEPH S.
301810014120513604000REITER, D.
304870017124383255000MARRA, DITTA E.
401710020128252906000LIPKE, VINCENT R.
407390023132122557000KUGLER, CHARLES
502070026135992208000JAVIER, CARLOS
505680029139861859000GODMAN, ISAAC
604910032143731510000FELDSOIT, MS. SALLY
608250035147601161000BUXBAUM, ROBERT
703100038151470812000DUMAY, MRS. MARY
708020041155340463000SMITH, R.
803220044159210114000VINCENF, MATTHEW J.
901050047163084235000THOMAS, THOMAS T.
```

**Figure 1.5** Output from Program 38.

```
10503 100040002 MORALES, LUIS 5000000
10893 101850005 JACOBSON, MRS. NELLIE 4651000
11277 201110008 GREENWOOD, JAMES 4302000
11664 209560011 COSTELLO, JOSEPH S. 3953000
12051 301810014 REITER, D. 3604000
12438 304870017 MARRA, DITTA E. 3255000
12825 401710020 LIPKE, VINCENT R. 2906000
13212 407390023 KUGLER, CHARLES 2557000
13599 502070026 JAVIER, CARLOS 2208000
13986 505680029 GODMAN, ISAAC 1859000
14373 604910032 FELDSOIT, MS. SALLY 1510000
14760 608250035 BUXBAUM, ROBERT 1161000
15147 703100038 DUMAY, MRS. MARY 0812000
15534 708020041 SMITH, R. 0463000
15921 803220044 VINCENF, MATTHEW J. 0114000
16308 901050047 THOMAS, THOMAS T. 4235000
```







Program 39 is shown in Figure 1.8. The Working Storage Section, beginning on line 00470, contains fields that we will need for the results of arithmetic.

The RD entry, line 00550, has a few clauses that we are seeing for the first time. The **CONTROL** clause tells Report Writer that **FINAL** totals are to be printed. In a later program we will see how totals for minor, intermediate, and major control breaks are indicated in the **CONTROL** clause. The **PAGE** clause is required if you want to control the vertical spacing of lines on the page. From the print spacing chart in Figure 1.7, you can see that the first detail line of the report is to print on line 8 of the page, and we indicate this to Report Writer by saying **FIRST DETAIL 8**. We are also required to tell Report Writer how many lines can fit on a page, and here we arbitrarily said 50.

Figure 1.8 Program 39.

0-CBI RELEASE 2.3 JULY 24, 1978

IBM OS/VS COBOL

```

00010 IDENTIFICATION DIVISION.
00020
00030 PROGRAM-ID.
00040     PRG39.
00050
00060 *AUTHOR.  MYRON F. MYERSON.
00070 *
00080 *     THIS PROGRAM PRINTS A REPORT TITLE, COLUMN HEADINGS,
00090 *     DETAIL LINES, AND A FINAL TOTAL LINE.
00100 *
00110 *****
00120
00130 ENVIRONMENT DIVISION.
00140
00150 CONFIGURATION SECTION.
00160 SOURCE-COMPUTER.
00170     IBM-370.
00180 OBJECT-COMPUTER.
00190     IBM-370.
00200
00210 INPUT-OUTPUT SECTION.
00220 FILE-CONTROL.
00230     SELECT ORDER-FILE-IN          ASSIGN TO CARDIN.
00240     SELECT ORDER-REPORT-FILE-OUT  ASSIGN TO PRINTER.
00250
00260 *****
00270
00280 DATA DIVISION.
00290
00300 FILE SECTION.
00310 FD ORDER-FILE-IN
00320     LABEL RECORDS ARE OMITTED.
00330
00340 01 ORDER-RECORD-IN.
00350     05 CUSTOMER-NUMBER-IN          PIC X(7).
00360     05 PART-NUMBER-IN             PIC X(8).
00370     05 FILLER                     PIC X(7).
00380     05 QUANTITY-IN               PIC 999.
00390     05 UNIT-PRICE-IN             PIC 9(4)V99.
00400     05 HANDLING-IN               PIC 99V99.
00410     05 FILLER                     PIC X(45).
00420
00430 FD ORDER-REPORT-FILE-OUT
00440     LABEL RECORDS ARE OMITTED
00450     REPORT IS DAILY-ORDER-REPORT.
00460
00470 WORKING-STORAGE SECTION.
00480 01 MORE-INPUT                     PIC X          VALUE "Y".
00490 01 TAX-RATE                       PIC V99          VALUE .07.
00500 01 MERCHANDISE-AMOUNT-W          PIC 9(6)V99.
00510 01 TAX-W                         PIC 9(4)V99.
00520 01 ORDER-TOTAL-W                 PIC 9(7)V99.

```