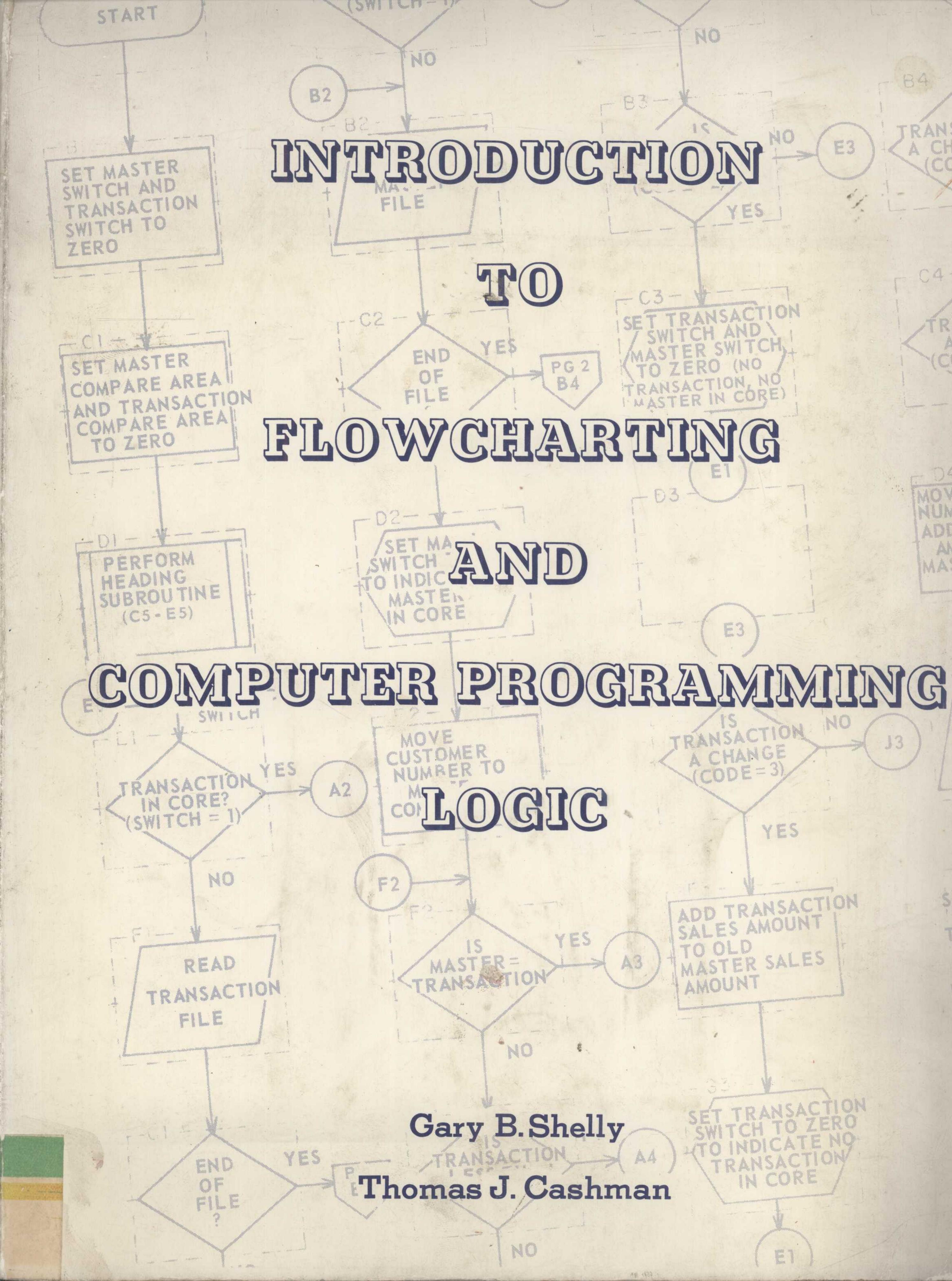

START
NO
B2
SET MASTER SWITCH AND TRANSACTION SWITCH TO ZERO
FILE
NO
E3
YES
SET MASTER COMPARE AREA AND TRANSACTION COMPARE AREA TO ZERO
END OF FILE
YES
PG 2 B4
SET TRANSACTION SWITCH AND MASTER SWITCH TO ZERO (NO TRANSACTION, NO MASTER IN CORE)
E1
PERFORM HEADING SUBROUTINE (C5 - E5)
SET MASTER SWITCH TO INDICATE MASTER IN CORE
E3
SWITCH
TRANSACTION IN CORE? (SWITCH = 1)
YES
A2
MOVE CUSTOMER NUMBER TO
IS TRANSACTION A CHANGE (CODE = 3)
NO
J3
YES
NO
F2
READ TRANSACTION FILE
IS MASTER = TRANSACTION
YES
A3
ADD TRANSACTION SALES AMOUNT TO OLD MASTER SALES AMOUNT
NO
END OF FILE ?
YES
TRANSACTION
A4
SET TRANSACTION SWITCH TO ZERO TO INDICATE NO TRANSACTION IN CORE
NO
E1

INTRODUCTION TO FLOWCHARTING AND COMPUTER PROGRAMMING LOGIC

Gary B. Shelly

Thomas J. Cashman

INTRODUCTION

TO FLOWCHARTING AND COMPUTER PROGRAMMING LOGIC

By:

Gary B. Shelly
Educational Consultant
Instructor, Long Beach City College

&

Thomas J. Cashman, CDP, B.A., M.A.
Long Beach City College
Long Beach, California

ANAHEIM PUBLISHING COMPANY
2632 Saturn, St., Brea, CA 92621
(714) 993-3700

Tenth Printing
June 1981

Printed in the United States of America

ISBN 0-88236-345-X

Library of Congress Catalog Card Number: 72-95674

10 9 8 7 6 5 4 3 2

INTRODUCTION TO
FLOWCHARTING
AND
COMPUTER PROGRAMMING
LOGIC

ANAHEIM PUBLISHING COMPANY

Specialist in Data Processing Textbooks

INTRODUCTION TO DATA PROCESSING

Introduction to Computers and Data Processing, Shelly & Cashman
Our Computerized Society, Logsdon & Logsdon
Introduction To Flowcharting and Computer Programming Logic, Shelly & Cashman

BASIC

Introduction To BASIC Programming, Shelly & Cashman
Programming In BASIC, Logsdon
Programming In BASIC With Applications, Logsdon

STRUCTURED COBOL

Introduction To Computer Programming Structured COBOL, Shelly & Cashman
Advanced Structured COBOL Program Design and File Processing, Shelly & Cashman

COBOL

Introduction To Computer Programming ANSI COBOL, Shelly & Cashman
ANSI COBOL Workbook, Testing & Debugging Techniques & Exercises, Shelly & Cashman
Advanced ANSI COBOL Disk/Tape Programming Efficiencies, Shelly & Cashman

RPG II

Computer Programming RPG II, Shelly & Cashman

RPG

Introduction To Computer Programming RPG, Shelly & Cashman

SYSTEMS ANALYSIS AND DESIGN

Business Systems Analysis and Design, Shelly & Cashman

ASSEMBLER LANGUAGE

Introduction To Computer Programming IBM System/360 Assembler Language, Shelly & Cashman
IBM System/360 Assembler Language Workbook, Shelly & Cashman
IBM System/360 Assembler Language Disk/Tape Advanced Concepts, Shelly & Cashman

FORTRAN

Introduction To Computer Programming Basic FORTRAN IV-A Practical Approach, Keys

PL/I

Introduction To Computer Programming System/360 PL/I, Shelly & Cashman

JOB CONTROL - OPERATING SYSTEMS

DOS Utilities Sort/Merge Multiprogramming, Shelly & Cashman
OS Job Control Language, Shelly & Cashman
DOS Job Control for Assembler Language Programmers, Shelly & Cashman
DOS Job Control for COBOL Programmers, Shelly & Cashman

FLOWCHARTING

Introduction To Flowcharting and Computer Programming Logic, Shelly & Cashman

PREFACE

In the past, one of the difficulties in training programming students has been in developing the ability to apply the knowledge of the specific characteristics of a computer language and the coding "mechanics" to the logical solution of a given problem. Typically students have become proficient in the use of the instructions comprising the programming language yet they have not had training in the area where most programming difficulties occur, the logic of programming!

This text is designed to provide instruction in the basic logic and flowcharting of business applications programming. By applying the concepts presented in this text students should be able to efficiently analyze and flowchart the logic of typical business programming problems no matter how complex.

Beginning with an introduction to the computerized processing of a business application and the role of the programmer in designing business application programs, the text continues with the following subject matter: Basic Input/Output, Crossfooting and Final Totals, Comparing, Control Codes in Input Records, Report Headings, Sub-routines and Programmed Switches, Control Breaks, Sequential File Updating, Table Search including both Sequential and Binary Search Techniques, and In-Core Sorting.

Each of these topics is illustrated through the use of an actual business application problem. The input to the computer and the desired output are illustrated so that the student is aware of the exact processing which is to be accomplished. In addition to a detailed analysis of flowcharting logic, the processing within the computer is also illustrated in a step-by-step sequence so that the processing which will occur is easily seen. There is no assumption that the student will be able to visualize what is happening. Each step and the resultant contents of main storage and the output produced is shown in detail. In addition to the logic which is illustrated, the student is introduced to the correct techniques of flowcharting a program—the proper symbols to be used, the methods of illustrating program logic and the use of flowcharting worksheets.

At the conclusion of each chapter, there are two student flowcharting assignments. The first assignment is similar to the problem solved in each chapter and should serve to reinforce the concepts and methods contained in the problems illustrated in the text. The second flowcharting assignment presents a more complex extension of the logic contained in each chapter and should serve as an important learning problem for the student. He will be required to apply the concepts studied to a problem which differs enough to require a very thorough understanding of the logic of programming.

This textbook may be used as the primary textbook in a course in computer programming logic or as a supplementary textbook in data processing or related courses which are designed to develop an understanding and appreciation of computer problem solving methods. Because of the design of the textbook, moving from basic data processing concepts such as input/output operations and systematically to more complex problems such as file updating, the book may be used effectively as a supplementary textbook in programming language courses. Each of the assignments with the input defined and the output illustrated, provide excellent programming assignments. The logic and programming techniques illustrated are programming language independent and apply to any of the machine oriented or high level languages currently available today.

When the student completes the study of the material contained within the text, he will possess a knowledge of the programming logic required to solve the large majority of business applications problems. By completing the student assignments at the conclusion of the chapters, the student will have had the opportunity to apply his knowledge in meaningful and practical applications.

Gary B. Shelly

Thomas J. Cashman

TABLE OF CONTENTS

CHAPTER 1 Introduction To Problem Solving 1

CHAPTER 2 Basic Input/Output Operations 19

CHAPTER 3 Basic Arithmetic Operations 35

CHAPTER 4 Comparing 51

CHAPTER 5 Control Codes 65

CHAPTER 6 Report Headings and Subroutines
Program Switches 87

CHAPTER 7 Minor Totals 117

CHAPTER 8 Major, Intermediate, Minor Totals 139

CHAPTER 9 Sequential File Updating Logic 167

CHAPTER 10 Sequential Table Search 197

CHAPTER 11 Binary Search - Table Lookup 215

CHAPTER 12 Internal Sort 231

CHAPTER 1

INTRODUCTION TO PROBLEM SOLVING

INTRODUCTION

The data processing industry has undergone dramatic changes since it first began in the late 1940's. Much of the drama has centered around the changes and improvements in the computer hardware and the related peripheral equipment, such as card readers, magnetic tape units, disk storage devices and other input/output units which are used with the computer.

As computer hardware has gone through an evolutionary period, so too has the art or science of programming. This includes both the jobs done by programmers and the methods available to the programmer for solving problems. Programming may be broken down into two broad categories: systems programming, and applications programming. The systems programmer is concerned with writing programs that make it easier to operate and program computers. Systems programmers turn out control programs that operate input/output equipment, test programs that detect errors and malfunctions within the computer, utility programs, such as sorts, which are available for the applications programmer to use to solve problems.

Applications programming is utilized in two general areas: scientific or engineering applications and business applications. The scientific applications programmer is normally involved in programs requiring a great deal of complex mathematical work. In many cases, scientific applications programmers are also mathematicians or engineers who are using the computer to solve problems with which they are directly concerned.

The business applications programmer writes programs to solve problems relating to the business transactions of a company. Such applications as payroll, billing, and inventory control are the areas of concern to a business applications programmer. Business programming is normally not performed by an accountant or sales manager or production manager. The job of programming most business applications is normally assigned to an individual working in the data processing department. Thus, the ''business programmer'' writes a wide variety of programs for the various business applications within the company.

Regardless of the type of programming which is to be accomplished, however, there are some common procedures which must be followed whenever a program is to be written. These include problem analysis, flowcharting, coding, testing and debugging, and documentation. Although subject to some flexibility, the diagram below illustrates the approximate time spent by a programmer on each of these areas when programming an application.

EXAMPLE

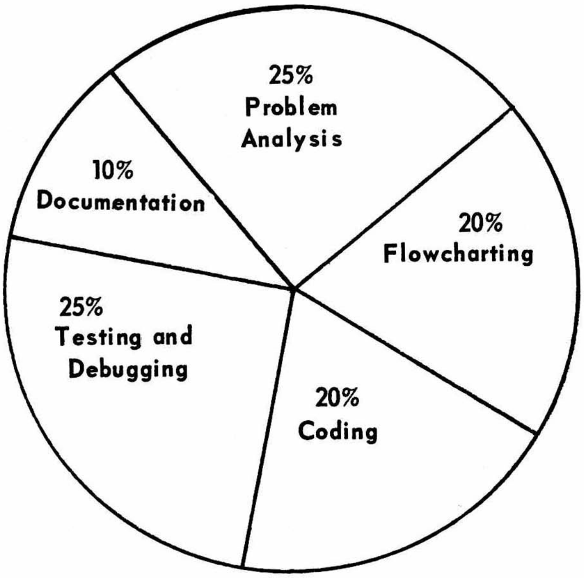


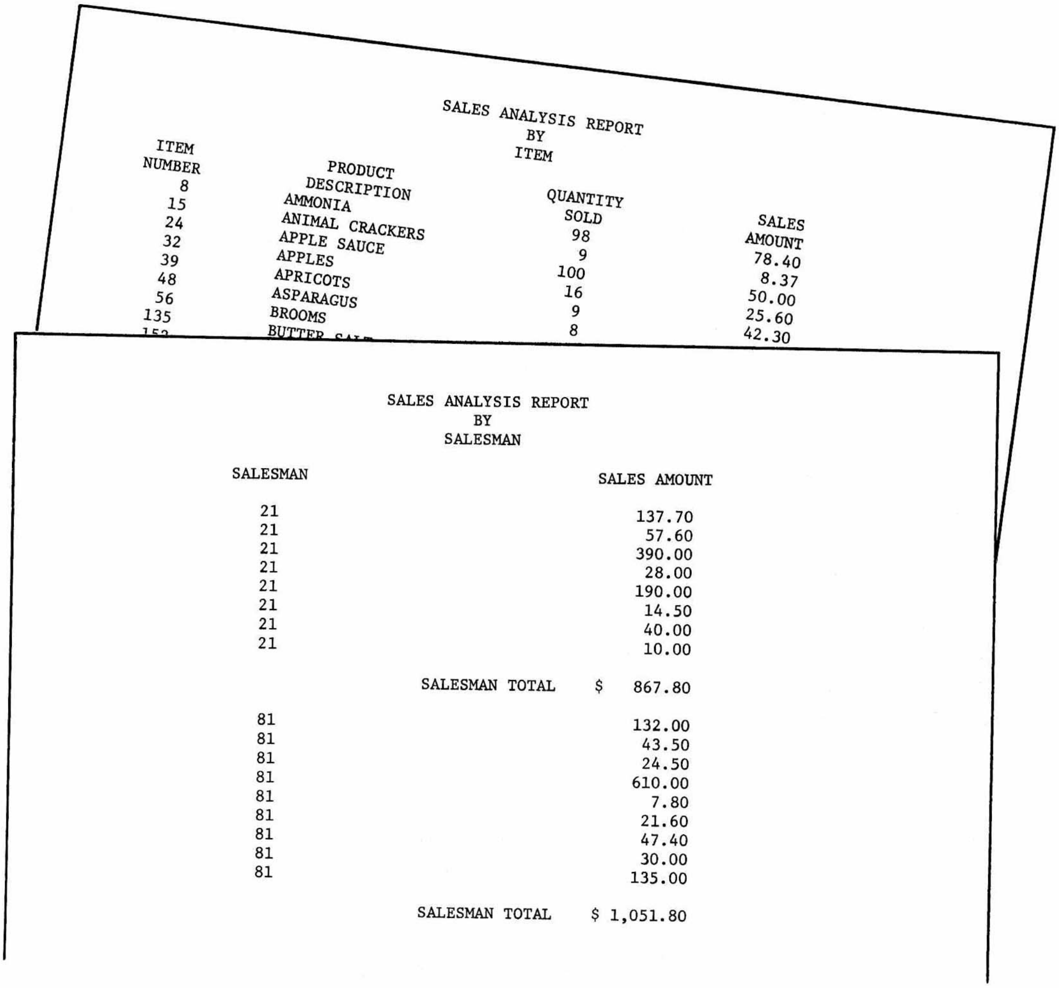
Figure 1-1 Average Time Spent on a Task by an Applications Programmer

Note from the illustration above that, contrary to popular belief, a programmer does not spend all of his time "writing" programs, that is, coding a program in a programming language. A great deal of the programmer's time is spent in analyzing a problem and determining a method of solution. The various job duties of the programmer as described in Figure 1-1 are explained on the following pages.

1. **Problem Analysis** - The normal sequence of a project which is to be implemented on a computer is for a systems analyst to design the overall system, which is composed of one or more programs. The system analyst gathers the information which is necessary to comprise the system from the eventual user of the system, from management, and from past experience which would indicate the type of processing which should be included in the system. He then places this information in a form which may be communicated to the programmer. Typically, the manner of presentation will include record formats, that is, the format of the input and output records which are to be processed by the program, printer spacing charts which are used to illustrate the format of a printed report, and some type of written narrative which will describe in detail the processing which is to take place within the system and within each program included in the system.

For example, the Design phase of a ''typical'' system could involve the following steps:

a. The Marketing Director in consultation with the Systems Analyst has determined that two reports are required by the management of the company to assist in analyzing current sales. These reports include a sales analysis report listing the items that have been sold each day, and a sales analysis report listing the daily sales by each salesman. These reports are currently being prepared manually and it has been determined that they could be prepared efficiently utilizing the computer at a saving of many hours of clerical effort. Figure 1-2 illustrates the reports.

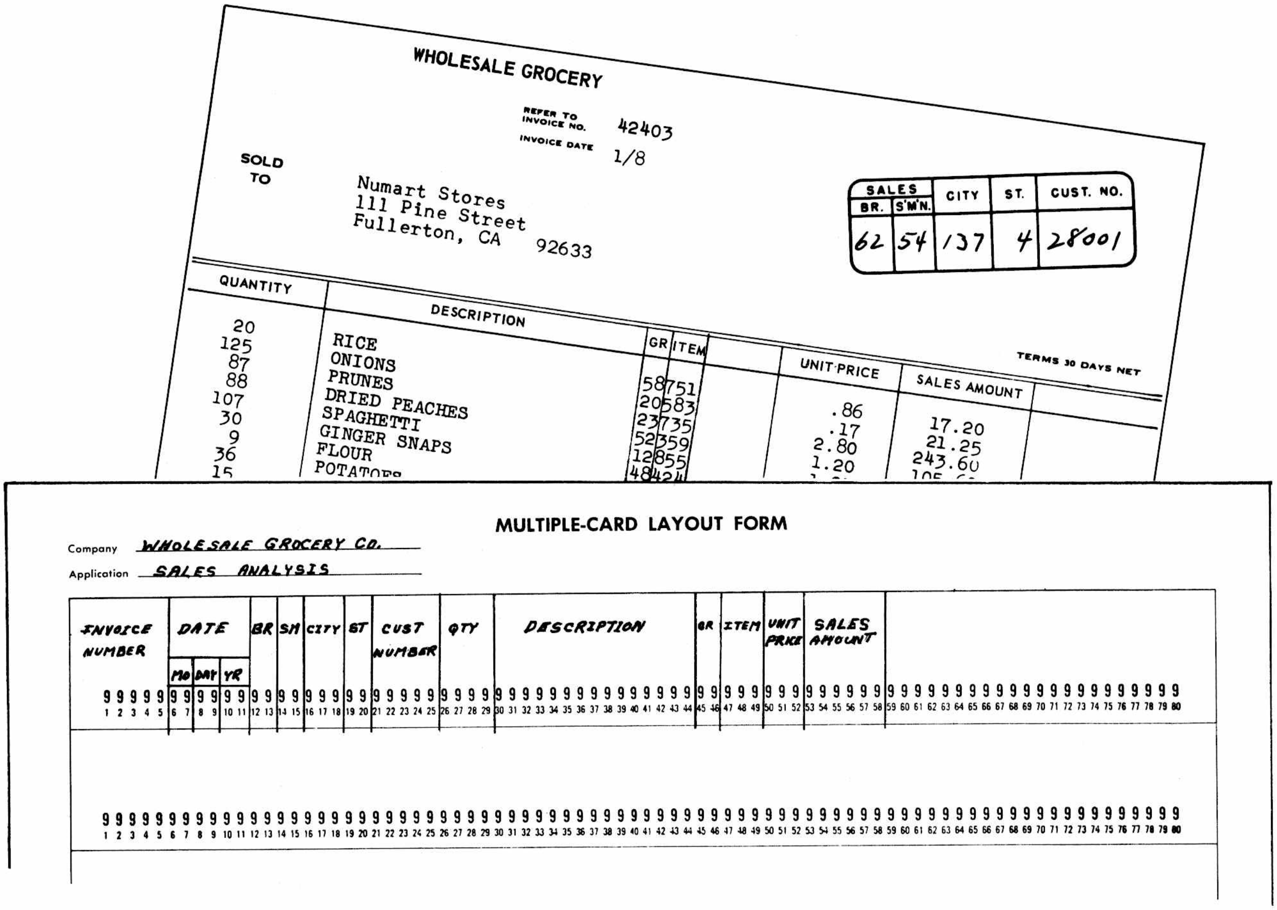


```
                         SALES ANALYSIS REPORT
                                  BY
                                 ITEM
 ITEM
NUMBER         PRODUCT
               DESCRIPTION          QUANTITY
                                      SOLD          SALES
                                                    AMOUNT
   8        AMMONIA                    98            78.40
  15        ANIMAL CRACKERS             9             8.37
  24        APPLE SAUCE               100            50.00
  32        APPLES                     16            25.60
  39        APRICOTS                    9            42.30
  48        ASPARAGUS                   8
  56        BROOMS
 135        BUTTER SA
```

```
                    SALES ANALYSIS REPORT
                             BY
                          SALESMAN

      SALESMAN                          SALES AMOUNT

         21                                137.70
         21                                 57.60
         21                                390.00
         21                                 28.00
         21                                190.00
         21                                 14.50
         21                                 40.00
         21                                 10.00

                     SALESMAN TOTAL   $    867.80

         81                                132.00
         81                                 43.50
         81                                 24.50
         81                                610.00
         81                                  7.80
         81                                 21.60
         81                                 47.40
         81                                 30.00
         81                                135.00

                     SALESMAN TOTAL   $ 1,051.80
```

Figure 1-2 Sales Report

b. After determining the reports that are needed, the systems analyst must then review the original sales orders which are used as the basis for preparing the reports. Selected information on these documents must be converted to punched cards for processing by the computer. The Multiple-Card Layout Form provides a convenient method of planning the format of the punched card. The sequence and the size of the fields which are to be punched are normally recorded on a Multiple-Card Layout Form by drawing a vertical line between the card columns separating the various fields on a card, and labeling these fields with the proper headings. The information from this form will provide the basis for the actual design of the punched card. Figure 1-3 illustrates a Sales Order and a Multiple-Card Layout Form. Note the fields which are recorded on the card. There will be one card punched for each of the items listed on the order.



WHOLESALE GROCERY

REFER TO INVOICE NO. 42403

INVOICE DATE 1/8

SOLD TO

Numart Stores
111 Pine Street
Fullerton, CA 92633

SALES BR.	SALES S'M'N.	CITY	ST.	CUST. NO.
62	54	137	4	28001

TERMS 30 DAYS NET

QUANTITY	DESCRIPTION	GR	ITEM	UNIT PRICE	SALES AMOUNT
20	RICE	58	751	.86	17.20
125	ONIONS	20	583	.17	21.25
87	PRUNES	23	735	2.80	243.60
88	DRIED PEACHES	52	359	1.20	105.6
107	SPAGHETTI	12	855		
30	GINGER SNAPS	48	424		
9	FLOUR				
36	POTATOES				
15					

MULTIPLE-CARD LAYOUT FORM

Company WHOLESALE GROCERY CO.

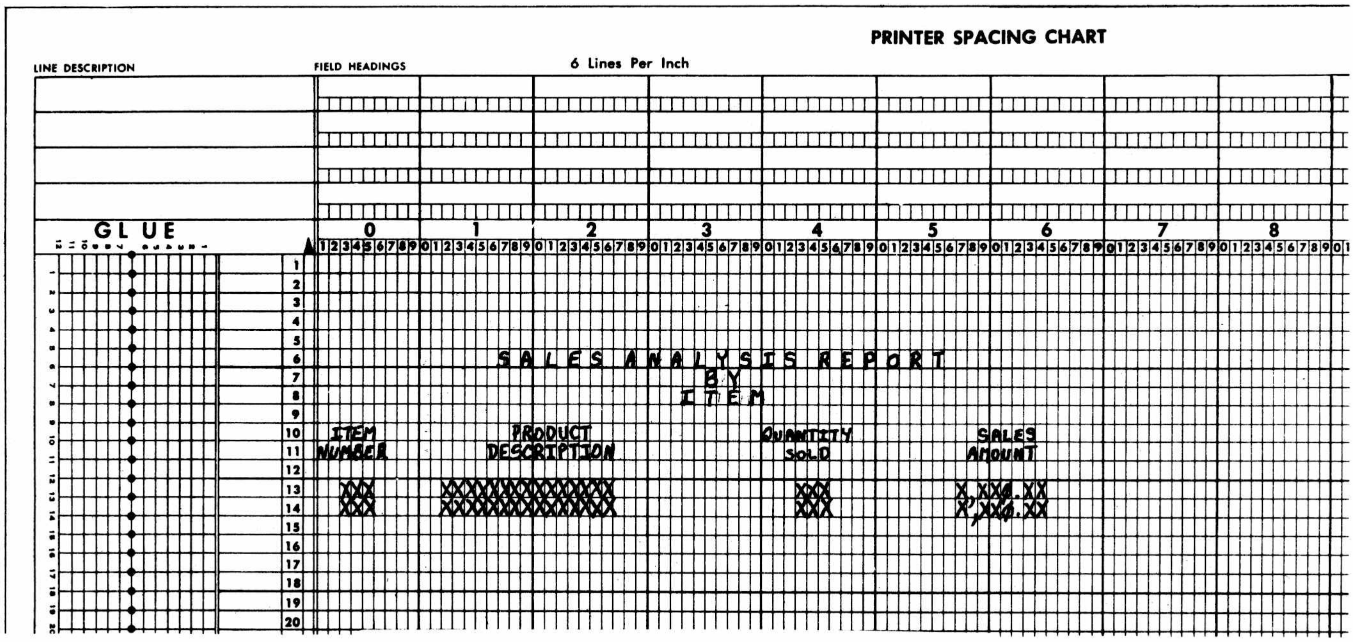
Application SALES ANALYSIS

INVOICE NUMBER	DATE			BR	SM	CITY	ST	CUST NUMBER	QTY	DESCRIPTION	GR	ITEM	UNIT PRICE	SALES AMOUNT	
	MO	DAY	YR												
9 9 9 9 9	9 9	9 9	9 9	9 9	9 9	9 9 9	9 9	9 9 9 9 9	9 9 9 9	9 9 9 9 9 9 9 9 9 9 9 9 9 9 9	9 9	9 9 9	9 9 9	9 9 9 9 9 9	9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5	6 7	8 9	10 11	12 13	14 15	16 17 18	19 20	21 22 23 24 25	26 27 28 29	30 31 32 33 34 35 36 37 38 39 40 41 42 43 44	45 46	47 48 49	50 51 52	53 54 55 56 57 58	59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Figure 1-3 Sales Order and Multiple-Card Layout Form

c. After the card has been designed, the format of the reports, as they are to be prepared by the computer, must be designed. A Printer Spacing Chart is commonly used. Figure 1-4 illustrates a Printer Spacing Chart for the Sales Analysis Report listing the items that have been sold.



PRINTER SPACING CHART

LINE DESCRIPTION

FIELD HEADINGS

6 Lines Per Inch

GLUE

```
                S A L E S  A N A L Y S I S  R E P O R T
                                 BY
                                I T E M

 ITEM            PRODUCT              QUANTITY           SALES
NUMBER        DESCRIPTION               SOLD            AMOUNT

  XXX      XXXXXXXXXXXXXXX               XXX           X,XXØ.XX
  XXX      XXXXXXXXXXXXXXX               XXX           X,XXØ.XX
```

Figure 1-4 Printer Spacing Chart

To lay out a report on the Printer Spacing Chart the programmer selects the print positions for the headings and fields to be printed on the report and makes a notation in the selected positions. The numbers across the top of the spacing chart represent the actual print positions used by the computer printer. The numbers down the left are line numbers. There are six horizontal lines to an inch. Heading or constant information to be printed is written on the printer spacing chart in the same form as it is to be printed on the report. Variable information is represented by placing an "X" in the appropriate position on the spacing chart. It should be noted that the position in a field where zero suppression ends is indicated by a zero rather than an "X" and punctuation is shown as it would appear in edited amounts.

When the programmer writes the program to process the data to produce the report the definition of the input and printed output will normally be in the form of a Multiple-Card Layout Form and a Printer Spacing Chart. Thus, through the use of these forms, the analyst informs the programmer of the formats of the records to be processed by the program.

In addition to the forms which are normally used by the systems analyst, a "system flowchart" is usually included in the specifications presented to the programmer. The System Flowchart indicates the "flow" of data in the system, that is, the sequence of events which is to occur within the system to process the data.

The procedure for preparing the reports in the example system involves converting the information on the Sales Orders into punched cards, sorting the cards by the Item Number field and processing the cards on the computer to prepare the sales analysis report by item. The next step requires that the cards be resorted by Salesman Number and then processing the cards on the computer again to prepare the second report. Figure 1-5 illustrates the System Flowchart. Note that special symbols are utilized to represent specific types of operations.

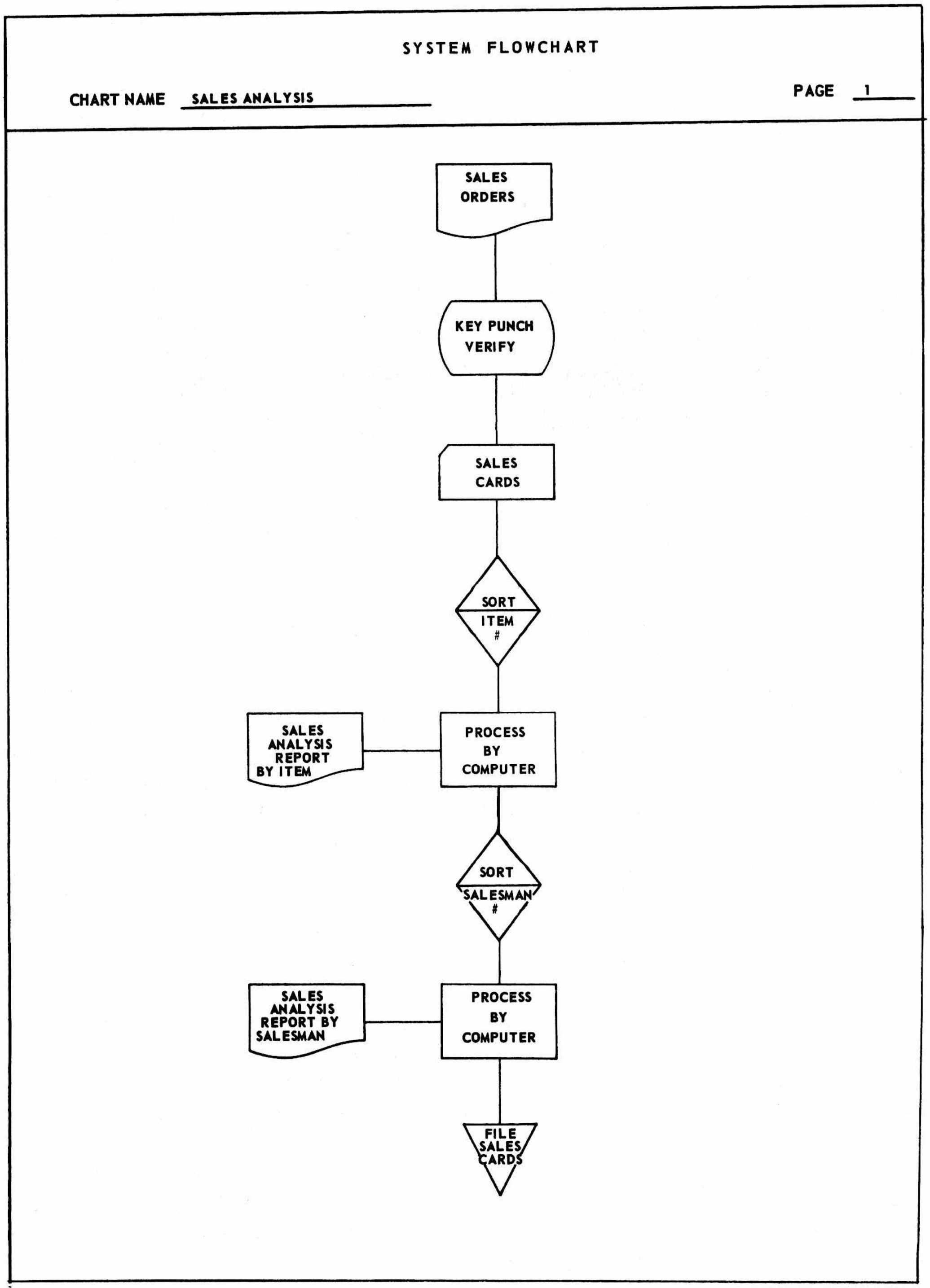


Figure 1-5 Systems Flowchart

In addition to the Multiple-Card Layout Form, the Printer Spacing Chart, and the System Flowchart, the systems analyst will normally provide the programmer with additional written instructions explaining any special processing that is to occur. When the programmer receives this package of specifications from the systems analyst, he must study and understand every aspect of processing which is to occur within the system. For example, the programmer must understand how to handle error cards such as cards containing blanks in the sales amount field; the programmer must know if all item numbers are to be considered valid; the programmer must know the maximum size of all calculations that can develop, etc.

It is absolutely mandatory that the programmer understand all of the processing which is to be performed by the programs which he is to write before attempting to determine the logic and processing which is necessary in order to solve the problem. One of the difficulties encountered in data processing is that the programmer does not understand all of the requirements of a program before beginning the flowchart and coding. Thus, when the program is written, certain aspects of the program are not included. Making corrections to a program and adding routines to process data in a manner not originally planned for is a very time-consuming and difficult task. It has been found that errors in a program are more apt to occur when routines are added and logic is changed after the original program design has been finalized than when all contingencies have been planned for in the original design of the program.

2. **Flowcharting** - When the programmer has determined that he understands all of the processing which is to take place within a program and all questions have been satisfactorily answered, the next step is to determine the logic within the program which is to be used to solve the problem. In most cases, the best way to express this logic is in terms of block diagrams or FLOWCHARTS. A Flowchart is a diagram using pre-defined symbols which represent the logic required to solve a defined problem. Figure 1-6 is an example of a flowchart which could be used to illustrate the logic necessary to read the Sales Order cards and create the Sales Analysis Report listing the items which have been sold.

Flowcharting Worksheet

Programmer: SHELLY Program No.: 1 Date: Page: 1
Chart ID: 1 Chart Name: SALES REPORT – ITEM Program Name:

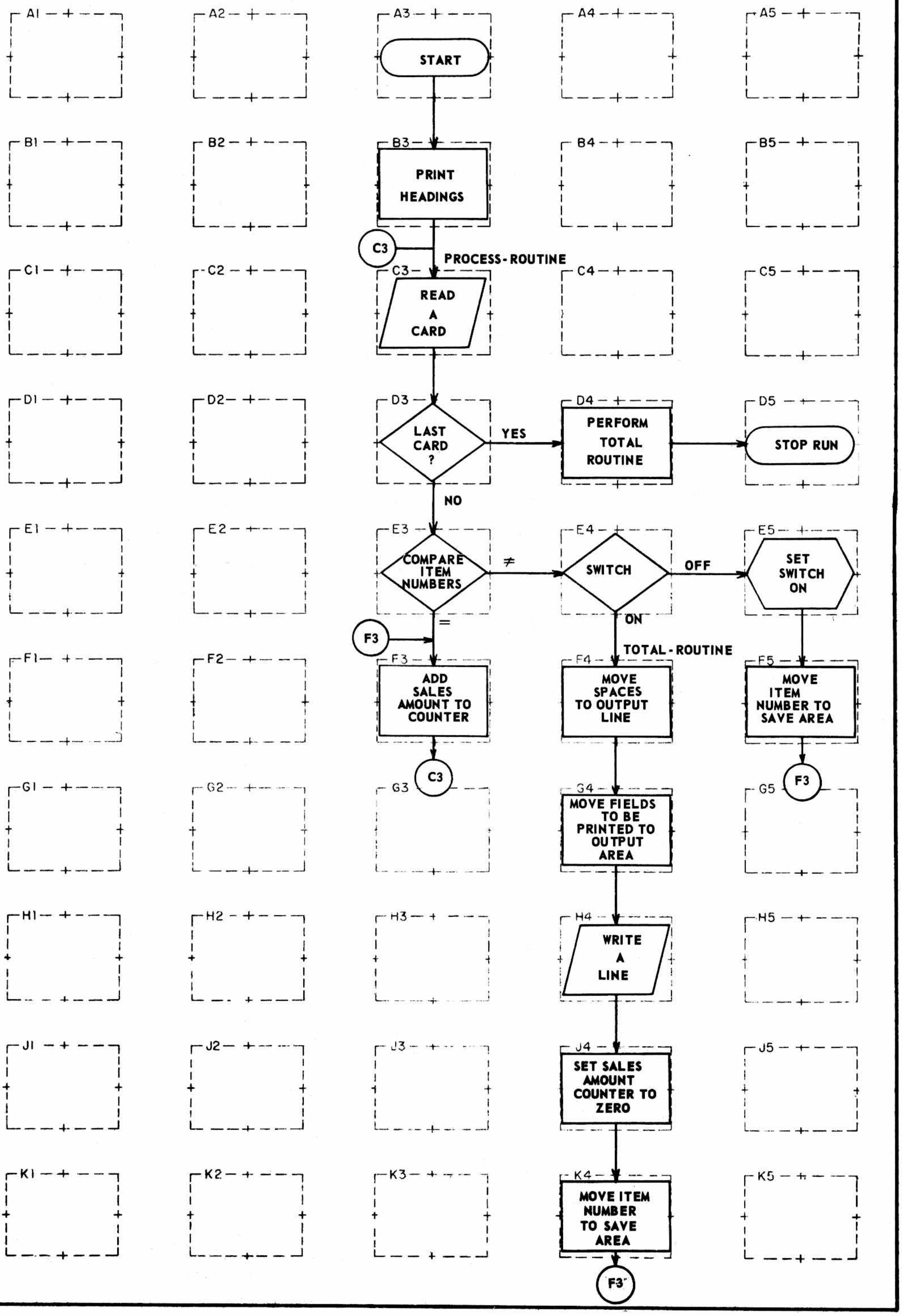
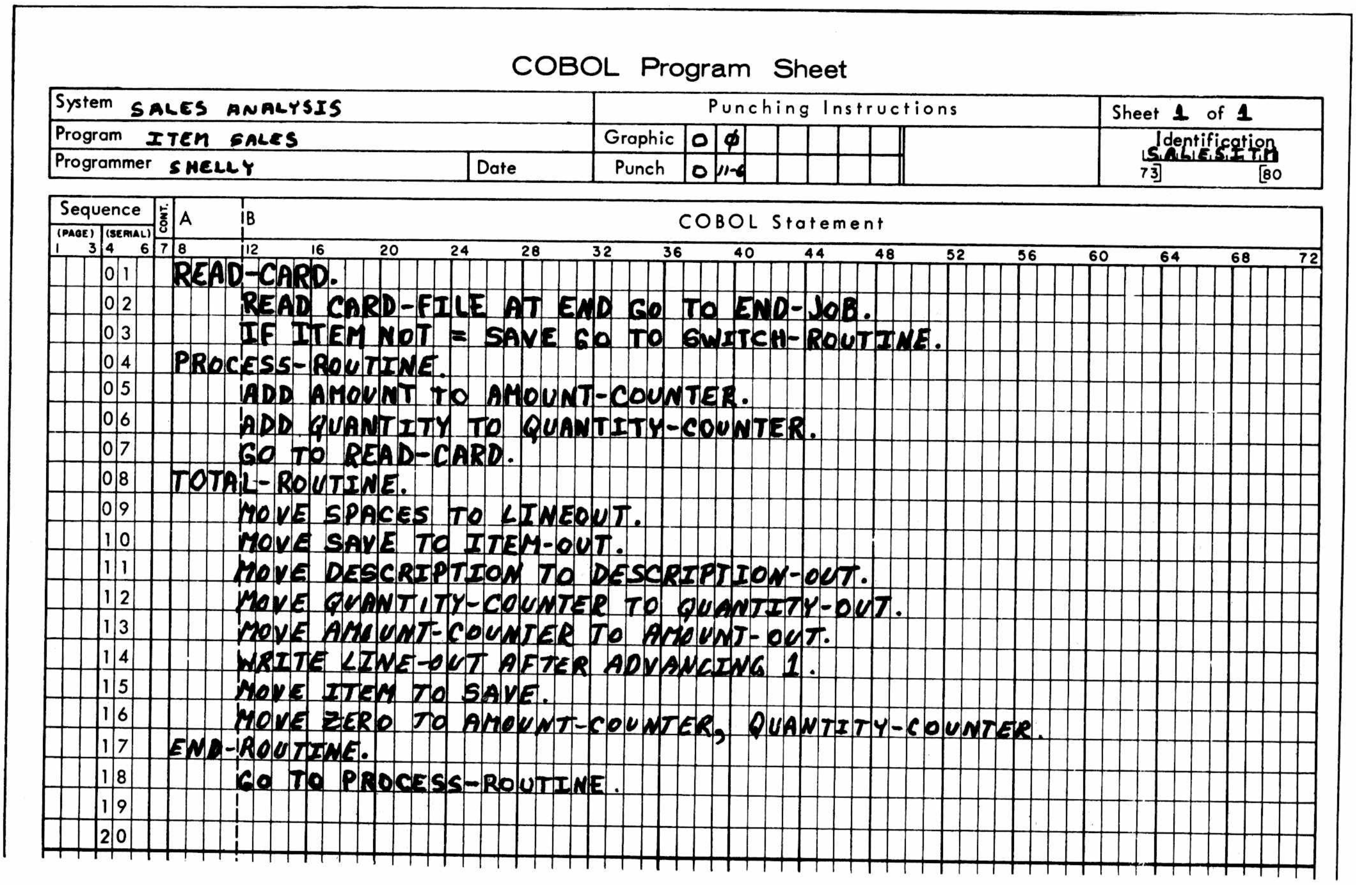


Figure 1-6 Program Flowchart

In the example in Figure 1-6 it can be seen that the flowchart, using specific symbols for each operation, illustrates the detailed logic which must be incorporated into the program to properly process the data. The use of the flowcharting symbols and the logic required to solve many business applications will be illustrated and explained in subsequent chapters.

3. **Coding** - After the logic which is to be incorporated into a program has been symbolically illustrated using a flowchart, the steps in the flowchart are converted into a computer program by the programmer. Commonly used programming languages include COBOL, Assembler Language, RPG, PL/I, and FORTRAN. The following is an illustration of a segment of a COBOL program to prepare the Sales Analysis Report listing the items sold.



COBOL Program Sheet

System SALES ANALYSIS		Punching Instructions	Sheet 1 of 1
Program ITEM SALES		Graphic O Ø	Identification SALESITM
Programmer SHELLY	Date	Punch O 11-6	73 80

Sequence (PAGE) (SERIAL) CONT. A B COBOL Statement

```
   01 READ-CARD.
          READ CARD-FILE AT END GO TO END-JOB.
   02
   03     IF ITEM NOT = SAVE GO TO SWITCH-ROUTINE.
   04 PROCESS-ROUTINE.
   05     ADD AMOUNT TO AMOUNT-COUNTER.
   06     ADD QUANTITY TO QUANTITY-COUNTER.
   07     GO TO READ-CARD.
   08 TOTAL-ROUTINE.
   09     MOVE SPACES TO LINEOUT.
   10     MOVE SAVE TO ITEM-OUT.
   11     MOVE DESCRIPTION TO DESCRIPTION-OUT.
   12     MOVE QUANTITY-COUNTER TO QUANTITY-OUT.
   13     MOVE AMOUNT-COUNTER TO AMOUNT-OUT.
   14     WRITE LINE-OUT AFTER ADVANCING 1.
   15     MOVE ITEM TO SAVE.
   16     MOVE ZERO TO AMOUNT-COUNTER, QUANTITY-COUNTER.
   17 END-ROUTINE.
   18     GO TO PROCESS-ROUTINE.
   19
   20
```

Figure 1-7 COBOL

COBOL (COmmon Business Oriented Language) is the programming language on a well-defined form of English and is especially suited for commercial or business applications.